

Junos® OS

EVPN User Guide

Published
2021-12-16

Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, California 94089
USA
408-745-2000
www.juniper.net

Juniper Networks, the Juniper Networks logo, Juniper, and Junos are registered trademarks of Juniper Networks, Inc. in the United States and other countries. All other trademarks, service marks, registered marks, or registered service marks are the property of their respective owners.

Juniper Networks assumes no responsibility for any inaccuracies in this document. Juniper Networks reserves the right to change, modify, transfer, or otherwise revise this publication without notice.

Junos® OS EVPN User Guide

Copyright © 2021 Juniper Networks, Inc. All rights reserved.

The information in this document is current as of the date on the title page.

YEAR 2000 NOTICE

Juniper Networks hardware and software products are Year 2000 compliant. Junos OS has no known time-related limitations through the year 2038. However, the NTP application is known to have some difficulty in the year 2036.

END USER LICENSE AGREEMENT

The Juniper Networks product that is the subject of this technical documentation consists of (or is intended for use with) Juniper Networks software. Use of such software is subject to the terms and conditions of the End User License Agreement ("EULA") posted at <https://support.juniper.net/support/eula/>. By downloading, installing or using such software, you agree to the terms and conditions of that EULA.

Table of Contents

1

About This Guide | xxiii

Features Common to EVPN-VXLAN and EVPN-MPLS

Configuring Interfaces | 2

VLAN ID Ranges and Lists in an EVPN Environment | 2

Understanding VLAN ID Ranges and Lists in an EVPN Environment | 2

Configuring VLAN ID Lists and Ranges in an EVPN Environment | 5

Configuring Routing Instances for EVPN | 10

Configuring EVPN Routing Instances | 10

Configuring EVPN Routing Instances on EX9200 Switches | 13

Overview of MAC Mobility | 16

Changing Duplicate MAC Address Detection Settings | 19

EVPN Type 5 Route with VXLAN encapsulation for EVPN-VXLAN | 21

EVPN Type 5 Route with MPLS encapsulation for EVPN-MPLS | 22

Understanding EVPN Pure Type 5 Routes | 22

EVPN Type 2 and Type 5 Route Coexistence with EVPN-VXLAN | 27

Ingress Virtual Machine Traffic Optimization | 36

Tracing EVPN Traffic and Operations | 42

Migrating From BGP VPLS to EVPN Overview | 44

Configuring Route Targets | 52

Auto-derived Route targets | 52

Example: Configuring VNI Route Targets Automatically | 54

Requirements | 55

Overview | 55

Configuration | 55

Example: Configuring VNI Route Targets Manually | 58

Requirements | 58

Overview | 58

Configuration | 58

Example: Configuring VNI Route Targets Automatically with Manual Override | 60

Requirements | 60

Overview | 60

Configuration | 61

Routing Policies for EVPN | 64

Routing policies for EVPN | 64

Example: Using Policy Filters to Filter EVPN Routes | 70

Requirements | 70

Overview | 71

Base Configuration | 71

Verification | 94

Configuring EVPN Multihoming | 96

EVPN Multihoming Overview | 96

Configuring EVPN Active-Standby Multihoming to a Single PE Device | 124

Configuring EVPN Active-Standby Multihoming | 127

Example: Configuring Basic EVPN Active-Standby Multihoming | 131

Requirements | 131

Overview and Topology | 132

Configuration | 132

Verification | 143

Example: Configuring EVPN Active-Standby Multihoming | 154

Requirements | 155

Overview and Topology | 155

Configuration | 158

Verification | 175

Example: Configuring Basic EVPN Active-Active Multihoming | 200

Requirements | 201

Overview | 201

Configuration | 202

Example: Configuring EVPN Active-Active Multihoming | 213

Requirements | 213

Overview and Topology | 214

Configuration | 215

Verification | 255

Example: Configuring LACP for EVPN Active-Active Multihoming | 279

Requirements | 279

Overview | 280

Configuration | 283

Verification | 296

Example: Configuring LACP for EVPN VXLAN Active-Active Multihoming | 301

Requirements | 301

Overview | 301

Configuration | 304

Verification | 327

Understanding When to Disable EVPN-VXLAN Core Isolation | 329

Configuring Dynamic List Next Hop | 332

Example: Configuring an ESI on a Logical Interface With EVPN Multihoming | 336

Requirements | 337

Overview and Topology | 337

EVPN Multihoming Active-Standby Configuration | 340

Verification | 349

Understanding Automatically Generated ESIs in EVPN Networks | 352

EVPN Proxy ARP and ARP Suppression, and NDP and NDP Suppression | 361

EVPN Proxy ARP and ARP Suppression, and Proxy NDP and NDP Suppression | 361

ARP and NDP Request with a proxy MAC address | 364

Configuring DHCP Relay Agents | 370

DHCP Relay Agent in EVPN-MPLS Network | 370

DHCP Relay Agent over EVPN-VXLAN | 373

Configuring MAC Pinning | 377

EVPN MAC Pinning Overview | 377

Configuring EVPN MAC Pinning | 379

Creating an Exclusion List for MAC Pinning | 379

High Availability in EVPN | 383

NSR and Unified ISSU Support for EVPN | 383

Preventing Traffic Loss in an EVPN/VXLAN Environment With GRES and NSR | 386

Graceful Restart in EVPN | 387

Monitoring EVPN Networks | 389

Connectivity Fault Management Support for EVPN and Layer 2 VPN Overview | 389

Configuring a MEP to Generate and Respond to CFM Protocol Messages | 391

Configuring a Maintenance Association End Point (MEP) | 392

Configuring a remote Maintenance Association End Point (MEP) | 394

2

EVPN-VXLAN

Overview | 398

Understanding EVPN with VXLAN Data Plane Encapsulation | 398

EVPN-over-VXLAN Supported Functionality | 408

Understanding VXLANs | 414

VXLAN Benefits | 415

How Does VXLAN Work? | 416

VXLAN Implementation Methods | 417

Using QFX5100, QFX5110, QFX5120, QFX5200, QFX5210, EX4300-48MP, and EX4600 Switches with VXLANs | 417

Changing the UDP Port on QFX5100, QFX5110, QFX5200, QFX5210, and EX4600 Switches | 418

Controlling Transit Multicast Traffic on QFX5100, QFX5110, QFX5200, QFX5210, and EX4600 Switches | 419

Using an MX Series Router, EX9200 Switch, or QFX10000 Switch as a VTEP | 420

Manual VXLANs Require PIM | 420

Load Balancing VXLAN Traffic | 421

VLAN IDs for VXLANs | 421

Enabling QFX5120 Switches to Tunnel Traffic on Core-Facing Layer 3 Tagged and IRB Interfaces | 421

Using ping and traceroute with a VXLAN | 422

Supported VXLAN Standards | 422

VXLAN Constraints on QFX Series and EX Series Switches | 423

EVPN Over VXLAN Encapsulation Configuration Overview for QFX Series and EX4600 Switches | 430

Implementing EVPN-VXLAN for Data Centers | 432

PIM NSR and Unified ISSU Support for VXLAN Overview | 436

Routing IPv6 Data Traffic through an EVPN-VXLAN Network with an IPv4 Underlay | 438

Understanding How to Configure VXLANs and Layer 3 Logical Interfaces to Interoperate | 453

Dynamic Load Balancing in an EVPN-VXLAN Network | 455

MAC-VRF Routing Instance Type Overview | 459

Configuring EVPN-VXLAN Interfaces | 465

Understanding Flexible Ethernet Services Support With EVPN-VXLAN | 465

MAC Filtering, Storm Control, and Port Mirroring Support in an EVPN-VXLAN Environment | 468

Example: Micro and Macro Segmentation using Group Based Policy in a VXLAN | 486

Overview | 486

Requirements | 491

Configuration | 491

DHCP Smart Relay in EVPN-VXLAN | 495

Configuring VLAN-Aware Bundle Services, VLAN-Based Services, and Virtual Switch Support | 497

Understanding VLAN-Aware Bundle and VLAN-Based Service for EVPN | 497

Configuring EVPN with VLAN-Based Service | 498

Virtual Switch Support for EVPN Overview | 504

Configuring EVPN with Support for Virtual Switch | 505

Setting Up a Layer 3 VXLAN Gateway | 508

Using a Default Layer 3 Gateway to Route Traffic in an EVPN-VXLAN Overlay Network | 508

Using a RIOT Loopback Port to Route Traffic in an EVPN-VXLAN Network | 514

Loopback Port Solution for Routing in and out of VXLAN Tunnels (RIOT) for Layer 3 VXLAN Gateway Support | 514

Configure a RIOT Loopback Port on a Layer 3 VXLAN Gateway Leaf Device | 518

Understanding the MAC Addresses For a Default Virtual Gateway in an EVPN-VXLAN Overlay Network | 524

Supported Protocols on an IRB Interface in EVPN-VXLAN | 527

Example: Configure an EVPN-VXLAN Centrally-Routed Bridging (CRB) Overlay | 530

Requirements | 531

Overview | 531

Spine 1: Underlay Network Configuration | 535

Spine 1: Overlay Network Configuration | 538

Spine 1: Access Profile Configuration | 540

Spine 2: Full Configuration | 543

Leaf 1: Underlay Network Configuration | 545

Leaf 1: Overlay Network Configuration | 547

Leaf 1: Access Profile Configuration | 549

Leaf 2: Full Configuration | 551

Leaf 3: Full Configuration | 552

Leaf 4: Full Configuration | 553

Verification | 554

Spine 1 and 2: Route Leaking (Optional) | 561

Verification with Route Leaking (Optional) | 563

Example: Configuring a QFX5110 Switch as a Layer 3 VXLAN Gateway in an EVPN-VXLAN Centrally-Routed Bridging Overlay | 566

Requirements | 567

Overview and Topology | 568

Basic Underlay Network Configuration | 571

Basic EVPN-VXLAN Overlay Network Configuration | 572

Basic Customer Profile Configuration | 574

Route Leaking Configuration | 577

Configuring an EVPN-VXLAN Centrally-Routed Bridged Overlay | 580

Example: Configure an EVPN-VXLAN Centrally-Routed Bridging (CRB) Using MX Routers as Spines | 580

Requirements | 580

- Overview | 581
- Topology | 582
- Configuration | 582
- Verification | 601

Configuring an EVPN-VXLAN Edge-Routed Bridging Overlay | 617

Example: Configuring an EVPN-VXLAN Edge-Routed Bridging Overlay Within a Data Center | 617

- Requirements | 618
- Overview and Topology | 619
- Configuration | 623
- Verification | 628

Example: Configuring a QFX5110 Switch as Layer 2 and 3 VXLAN Gateways in an EVPN-VXLAN Edge-Routed Bridging Overlay | 630

- Requirements | 631
- Overview and Topology | 632
- Basic Underlay Network Configuration | 634
- Basic EVPN-VXLAN Overlay Network Configuration | 635
- Basic Customer Profile Configuration | 637
- Route Leaking Configuration | 640

IPv6 Underlay for VXLAN Overlays | 643

EVPN-VXLAN with an IPv6 Underlay | 643

- IPv6 Underlay Support in EVPN-VXLAN Fabrics | 643
- Configure an IPv6 Underlay with EVPN-VXLAN | 648

Example: Configure an IPv6 Underlay for Layer 2 VXLAN Gateway Leaf Devices | 649

- Overview | 650
- Requirements | 652
- Topology | 653
- Configure Leaf 1 | 654
- Configure Leaf 3 | 658
- Verify the IPv6 Underlay on Leaf 3 | 662

Multicast Features with EVPN-VXLAN | 668

Multicast Support in EVPN-VXLAN Overlay Networks | 668

- IPv4 Inter-VLAN Multicast Forwarding Modes for EVPN-VXLAN Overlay Networks | 668

Overview of Multicast Forwarding with IGMP Snooping or MLD Snooping in an EVPN-VXLAN Environment | **675**

Example: Preserving Bandwidth with IGMP Snooping in an EVPN-VXLAN Environment | **694**

Requirements | **695**

Overview | **695**

Configuration | **697**

Verification | **739**

Overview of Selective Multicast Forwarding | **747**

Configuring the number of SMET Nexthops | **750**

Assisted Replication Multicast Optimization in EVPN Networks | **752**

Assisted Replication in EVPN Networks | **752**

Configure Assisted Replication | **766**

Configure an AR Replicator Device | **766**

Configure an AR Leaf Device | **768**

Verify Assisted Replication Setup and Operation | **768**

Optimized Inter-Subnet Multicast in EVPN Networks | **773**

Overview of OISM | **773**

OISM Components | **775**

How OISM Works | **786**

Configure Common OISM Elements on Border Leaf Devices and Server Leaf Devices (Symmetric Bridge Domains) | **798**

Configure Border Leaf Device OISM Elements (Symmetric Bridge Domains) | **801**

Configure Server Leaf Device OISM Elements (Symmetric Bridge Domains) | **804**

CLI Commands to Verify the OISM Configuration | **804**

Configuring the Tunneling of Q-in-Q Traffic | 809

Examples: Tunneling Q-in-Q Traffic in an EVPN-VXLAN Overlay Network | **809**

Requirements | **811**

Overview and Topology | **811**

Configuring Traffic Pattern 1: Popping an S-VLAN Tag | **815**

Requirements | **816**

Introduction | **816**

Ingress VTEP Configuration for Traffic Pattern 1 | **816**

Egress VTEP Configuration for Traffic Pattern 1 | **818**

Configuring Traffic Pattern 2: Mapping a Range of C-VLANs to an S-VLAN, and Pushing an S-VLAN Tag | 820

Requirements | 820

Introduction | 820

Ingress VTEP Configuration for Traffic Pattern 2 | 821

Egress VTEP Configuration for Traffic Pattern 2 | 823

Configuring Traffic Pattern 3: Retaining S-VLAN and C-VLAN Tags | 825

Requirements | 825

Introduction | 825

Ingress and Egress VTEP Configuration for Traffic Pattern 3 | 825

Configuring Traffic Pattern 4: Popping and Later Pushing an S-VLAN Tag | 828

Requirements | 828

Introduction | 828

Configuration for Ingress VTEP for Traffic Pattern 4 | 828

Configuration for Egress VTEP for Traffic Pattern 4 | 830

Tunnel Traffic Inspection on SRX Series Devices | 834

Tunnel Inspection for EVPN-VXLAN by SRX Series Devices | 834

Overview | 834

Example - Configure Security Policies for EVPN-VXLAN Tunnel Inspection | 837

Configuration for Zone-Level Inspection, IDP, UTM and Advanced Anti-Malware for Tunnel Inspection | 850

Complete Device Configurations | 860

3

EVPN-MPLS

Overview | 874

EVPN Overview | 874

EVPN-MPLS Caveats | 877

EVPN Overview for Switches | 878

Supported EVPN Standards | 879

Migrating from FEC128 LDP-VPLS to EVPN Overview | 881

Convergence in an EVPN MPLS Network | 891

Convergence in a Multihomed EVPN MPLS Network | 891

Pseudowire Termination at an EVPN | 893

Overview of Pseudowire Termination at an EVPN | 893

Configuring Pseudowire Termination | 894

Support for Redundant Logical Tunnel | 898

Configuring the Distribution of Routes | 900

Configuring an IGP on the PE and P Routers on EX9200 Switches | 900

Configuring IBGP Sessions Between PE Routers in VPNs on EX9200 Switches | 901

Configuring a Signaling Protocol and LSPs for VPNs on EX9200 Switches | 902

Configuring Entropy Labels | 905

Configuring Control Word for EVPN-MPLS | 906

Understanding P2MPs LSP for the EVPN Inclusive Provider Tunnel | 908

Configuring Bud Node Support | 910

Configuring VLAN Services and Virtual Switch Support | 912

Overview of VLAN Services for EVPN | 912

VLAN-Based Service for EVPN | 915

VLAN Bundle Service for EVPN | 918

Configuring EVPN VLAN Bundle Services | 920

Virtual Switch Support for EVPN Overview | 923

Configuring EVPN with Support for Virtual Switch | 925

Example: Configuring EVPN with Support for Virtual Switch | 929

Example: Configuring EVPN with Support for Virtual Switch | 929

Requirements | 930

Overview | 930

Configuration | 931

Verification | 942

Configuring Integrated Bridging and Routing | 946

EVPN with IRB Solution Overview | 946

An EVPN with IRB Solution on EX9200 Switches Overview | 952

Anycast Gateways | 958

Configuring EVPN with IRB Solution | 962

Configuring an EVPN with IRB Solution on EX9200 Switches | 965

Example: Configuring EVPN with IRB Solution | 968

 EVPN with IRB Solution Overview | 968

 Example: Configuring EVPN with IRB Solution | 974

 Requirements | 974

 Overview | 975

 Configuration | 976

 Verification | 985

Example: Configuring an EVPN with IRB Solution on EX9200 Switches | 995

 Requirements | 996

 Overview | 996

 Configuration | 996

 Verification | 1005

Configuring IGMP or MLD Snooping with EVPN-MPLS | 1013

Overview of Multicast Forwarding with IGMP or MLD Snooping in an EVPN-MPLS Environment | 1013

Configure Multicast Forwarding with IGMP Snooping in an EVPN-MPLS Environment | 1028

 Configure IGMP Snooping for an EVPN or Virtual Switch Routing Instance | 1029

 Configure IGMP Snooping with IGMPv3 on ACX Series Routers to Process Source-Specific Multicast Group Membership Reports Only | 1031

 Configure Multicast Routing Across Bridge Domains or VLANs with PIM in EVPN-MPLS | 1032

 Viewing IGMP Snooping Multicast Information for EVPN-MPLS in the CLI | 1035

Configure Multicast Forwarding with MLD Snooping in an EVPN-MPLS Environment | 1038

 Configure MLD Snooping for Default Any-Source Multicast (ASM) Group Membership Processing with MLDv1 or MLDv2 | 1039

 Configure MLD Snooping with MLDv2 to Process Source-Specific Multicast Group Membership Reports Only | 1042

 Viewing MLD Snooping Multicast Information for EVPN-MPLS in the CLI | 1043

EVPN-VPWS

Configuring VPWS Service with EVPN Mechanisms | 1046

Overview of VPWS with EVPN Signaling Mechanisms | 1046

Control word for EVPN-VPWS | 1050

Overview of Flexible Cross-Connect Support on VPWS with EVPN | 1054

Overview of Headend Termination for EVPN VPWS for Business Services | 1060

Configuring VPWS with EVPN Signaling Mechanisms | 1068

Example: Configuring VPWS with EVPN Signaling Mechanisms | 1070

Requirements | 1070

Overview and Topology | 1071

Configuration | 1073

Verification | 1085

FAT Flow Labels in EVPN-VPWS Routing Instances | 1096

5

EVPN-ETREE

Overview | 1102

EVPN-ETREE Overview | 1102

Configuring EVPN-ETREE | 1106

Example: Configuring EVPN-ETREE SERVICE | 1106

Requirements | 1106

Overview | 1107

Configuration | 1107

Verification | 1116

6

Using EVPN for Interconnection

Interconnecting VXLAN Data Centers With EVPN | 1123

VXLAN Data Center Interconnect Using EVPN Overview | 1123

Example: Configuring VXLAN Data Center Interconnect Using EVPN | 1142

Requirements | 1143

Overview | 1143

Configuration | 1144

Verificatiton | 1156

Interconnecting EVPN-VXLAN Data Centers Through an EVPN-MPLS WAN | 1163

EVPN-VXLAN Data Center Interconnect Through EVPN-MPLS WAN Overview | 1163

Example: Interconnecting EVPN-VXLAN Data Center Networks Through a WAN Running EVPN-based MPLS | 1175

Requirements | 1175

Overview | 1176

Configuration | 1180

Verification | 1275

Overview of EVPN-VXLAN Interconnects through EVPN MPLS-MPLS WAN Using Gateways | 1356

Extending a Junos Fusion Enterprise Using EVPN-MPLS | 1359

Understanding EVPN-MPLS Interworking with Junos Fusion Enterprise and MC-LAG | 1359

Example: EVPN-MPLS Interworking With Junos Fusion Enterprise | 1366

Requirements | 1367

Overview and Topology | 1368

Aggregation Device (PE1 and PE2) Configuration | 1370

PE3 Configuration | 1383

Example: EVPN-MPLS Interworking With an MC-LAG Topology | 1387

Requirements | 1387

Overview and Topology | 1388

PE1 and PE2 Configuration | 1390

PE3 Configuration | 1407

7

PBB-EVPN

Configuring PBB-EVPN Integration | 1414

Provider Backbone Bridging (PBB) and EVPN Integration Overview | 1414

Example: Configuring PBB with Single-Homed EVPN | 1450

Requirements | 1450

Overview and Topology | 1450

Configuration | 1451

Verification | 1470

Example: Configuring PBB with Multihomed EVPN | 1478

Requirements | 1478

Overview and Topology | 1479

Configuration | 1480

Verification | 1512

Configuring MAC Pinning for PBB-EVPNs | 1520

PBB-EVPN MAC Pinning Overview | 1520

Configuring PBB-EVPN MAC Pinning | 1521

8

EVPN Standards

EVPN Standards | 1526

Supported EVPN Standards | 1526

9

VXLAN-Only Features

Flexible VXLAN Tunnels | 1529

Understanding Programmable Flexible VXLAN Tunnels | 1529

Static VXLAN | 1536

Static VXLAN | 1536

Understanding Static VXLAN | 1536

Configuring Static VXLAN | 1539

10

Configuration Statements and Operational Commands

EVPN Configuration Statements | 1544

advertise-ip-prefix | 1546

assisted-replication | 1547

auto-derive | 1549

bgp-peer | 1551

designated-forwarder-election-hold-time (evpn) | 1552

df-election-granularity | 1554

duplicate-mac-detection | 1555

esi | 1558

esi-resolution-per-bridge-domain | 1561

evpn | 1562

evpn-aliasing-optimize | 1567

evpn-ssm-reports-only | 1568

exclusive-mac | 1570

export (Routing Options) | 1572

extended-isid-list | 1574

extended-vlan-list | 1575

flow-label-receive-static | 1577

flow-label-transmit-static | 1579

forwarding-instance identifier | 1580

global-mac-ip-limit | 1582

global-mac-ip-table-aging-time | 1583

global-mac-move | 1585

global-no-control-mac-aging | 1587

import | 1588

instance-type | 1589

instance-type mac-vrf | 1593

interconnect | 1595

interconnected-vni-list | 1597

interface (EVPN Routing Instances) | 1599

interface-mac-ip-limit | 1600

interface-mac-limit (VPLS) | 1602

interface-state | 1604

ip-prefix-routes | 1606

ip-prefix-support | 1610

label-allocation | 1612

leaf | 1613

loop-detect (EVPN) | 1615

mac-ip-table-size | 1619

mac-pinning (EVPN Routing Instances) | 1620

mclag | **1622**

multicast-mode (EVPN) | **1623**

multicast-replication | **1625**

multicast-snooping-options | **1628**

no-arp-suppression | **1630**

no-default-gateway-ext-comm | **1632**

nsr-phantom-holdtime | **1633**

oism | **1634**

oism (Multicast Snooping Options) | **1636**

overlay-ecmp | **1638**

pbb-evpn-core | **1640**

per-esi | **1641**

per-esi-vlan | **1643**

proxy-mac | **1644**

proxy-macip-advertisement | **1646**

remote-ip-host-routes | **1647**

replicator | **1649**

route-distinguisher | **1651**

service-type | **1655**

traceoptions (Protocols EVPN) | **1656**

translation-vni | **1659**

vlan-id (routing instance) | **1662**

vpn-apply-export | **1664**

vpws-service-id | **1665**

vrf-export | **1667**

vrf-import | **1668**

vrf-target | **1670**

VXLAN Configuration Statements | 1673

decapsulate-inner-vlan | **1674**

encapsulate-inner-vlan | **1675**

encapsulation vxlan | **1677**

extended-vni-all | **1678**

extended-vni-list | **1680**

ingress-node-replication (EVPN) | **1682**

interface-num | **1683**

loopback-port | **1685**

next-hop (VXLAN Routing) | **1687**

policy-set | **1689**

riot-loopback | **1691**

static-remote-vtep-list | **1692**

tunnel-inspection | **1696**

virtual-gateway-address | **1698**

virtual-gateway-v4-mac | **1700**

virtual-gateway-v6-mac | **1702**

vni | **1704**

vni-options | **1706**

vnid (EVPN) | **1707**

vtep-source-interface | **1708**

vxlan | **1710**

vxlan-disable-copy-tos-decap | **1712**

vxlan-disable-copy-tos-encap | **1714**

vxlan-gbp-profile | **1716**

[vxlan-routing | 1718](#)

[EVPN Operational Commands | 1720](#)

[clear bridge mac-ip-table | 1722](#)

[clear ethernet-switching mac-ip-table | 1723](#)

[clear ethernet-switching evpn arp-table | 1724](#)

[clear ethernet-switching evpn nd-statistics | 1726](#)

[clear ethernet-switching evpn nd-table | 1727](#)

[clear evpn duplicate-mac-suppression | 1729](#)

[clear evpn mac-ip-table | 1730](#)

[clear evpn nd-table | 1731](#)

[clear evpn statistics | 1733](#)

[clear loop-detect enhanced interface | 1734](#)

[clear mac-vrf forwarding mac-ip-table | 1736](#)

[clear mac-vrf forwarding mac-learning-log | 1738](#)

[clear mac-vrf forwarding mac-table | 1739](#)

[clear mac-vrf forwarding recovery-timeout | 1741](#)

[clear mac-vrf forwarding redundancy-group | 1742](#)

[show bridge mac-ip-table | 1743](#)

[show ethernet-switching mac-ip-table | 1748](#)

[show ethernet-switching evpn nd-statistics | 1755](#)

[show ethernet-switching evpn nd-table | 1758](#)

[show ethernet-switching flood | 1762](#)

[show ethernet-switching mgrp-policy | 1767](#)

[show ethernet-switching table | 1773](#)

[show ethernet-switching-vxlan-tunnel-end-point esi | 1806](#)

[show-ethernet-switching-vxlan-tunnel-end-point source | 1809](#)

show ethernet-switching vxlan-tunnel-end-point svlnh | **1811**

show evpn arp-table | **1815**

show evpn database | **1819**

show evpn flood | **1825**

show evpn igmp-snooping database | **1827**

show evpn igmp-snooping proxy | **1831**

show evpn instance | **1834**

show evpn ip-prefix-database | **1852**

show evpn l3-context | **1862**

show evpn mac-ip-table | **1866**

show evpn mac-table | **1870**

show evpn mld-snooping database | **1872**

show evpn multicast-snooping assisted-replication multihomed-peers | **1876**

show evpn multicast-snooping assisted-replication next-hops | **1880**

show evpn multicast-snooping assisted-replication replicators | **1883**

show evpn multicast-snooping status | **1888**

show evpn nd-table | **1890**

show evpn oism | **1894**

show evpn p2mp | **1896**

show evpn peer-gateway-macs | **1900**

show evpn prefix | **1901**

show evpn vpws-instance | **1903**

show loop-detect enhanced interface | **1911**

show mac-vrf forwarding flood | **1917**

show mac-vrf forwarding flood-group | **1921**

show mac-vrf forwarding global-information | **1926**

show mac-vrf forwarding global-mac-count | 1929

show mac-vrf forwarding global-mac-ip-count | 1931

show mac-vrf forwarding instance | 1932

show mac-vrf forwarding instance-mapping | 1938

show mac-vrf forwarding interface | 1941

show mac-vrf forwarding mac-ip-table | 1946

show mac-vrf forwarding mac-table | 1951

show mac-vrf forwarding mgrp-policy | 1955

show mac-vrf forwarding statistics | 1961

show mac-vrf forwarding vlans | 1966

show mac-vrf forwarding vxlan-tunnel-end-point esi | 1969

show mac-vrf forwarding vxlan-tunnel-end-point remote | 1973

show mac-vrf forwarding vxlan-tunnel-end-point svlbnh | 1982

show mld snooping evpn database | 1985

show route forwarding-table | 1989

show route table | 2003

show vlans evpn nd-table | 2028

VXLAN Operational Commands | 2033

show flexible-tunnels profiles | 2033

show security flow tunnel inspection statistics | 2039

show security policies policy set | 2042

show security tunnel inspection | 2045

About This Guide

Use this guide to learn more about, configure, and monitor EVPN-VXLAN, EVPN-MPLS, EVPN-VPWS, EVPN-ETREE, and PBB-EVPN on Juniper Network devices.

1

PART

Features Common to EVPN-VXLAN and EVPN-MPLS

[Configuring Interfaces | 2](#)

[Configuring Routing Instances for EVPN | 10](#)

[Configuring Route Targets | 52](#)

[Routing Policies for EVPN | 64](#)

[Configuring EVPN Multihoming | 96](#)

[EVPN Proxy ARP and ARP Suppression, and NDP and NDP Suppression | 361](#)

[Configuring DHCP Relay Agents | 370](#)

[Configuring MAC Pinning | 377](#)

[High Availability in EVPN | 383](#)

[Monitoring EVPN Networks | 389](#)

CHAPTER 1

Configuring Interfaces

IN THIS CHAPTER

- [VLAN ID Ranges and Lists in an EVPN Environment | 2](#)

VLAN ID Ranges and Lists in an EVPN Environment

IN THIS SECTION

- [Understanding VLAN ID Ranges and Lists in an EVPN Environment | 2](#)
- [Configuring VLAN ID Lists and Ranges in an EVPN Environment | 5](#)

You can specify VLAN ID lists and ranges in a service provider style of interface configuration that is referenced in an Ethernet VPN (EVPN) routing instance (a routing instance of type evpn). For more information, see the following topics:

Understanding VLAN ID Ranges and Lists in an EVPN Environment

IN THIS SECTION

- [Benefits of VLAN ID Range and List Support | 3](#)
- [VLAN Bundle Service | 4](#)
- [Sample VLAN ID Range and List Configuration | 4](#)
- [Caveats and Limitations | 5](#)

The service provider style of interface configuration enables you to customize Ethernet-based services at the logical interface level. Service providers typically have multiple customers connected to the same physical interface or aggregated Ethernet interface. Using the service provider style, you can configure multiple logical interfaces on the physical interface or aggregated Ethernet interface and associate each unit with a different VLAN.

Starting in Junos OS Release 19.2R1, you can specify VLAN ID lists and ranges in a service provider style interface configuration that is referenced in an Ethernet VPN (EVPN) routing instance (a routing instance of type evpn). This configuration is supported with the following EVPN environments, services, and features:

- Environments:
 - EVPN with Virtual Extensible LAN (VXLAN) encapsulation
 - EVPN with MPLS encapsulation
- VLAN bundle service:
 - E-LAN
 - E-Tree
 - E-Line
- Features:
 - EVPN multihoming
 - All-active
 - Single-active
 - Single homing

Benefits of VLAN ID Range and List Support

Without the support of VLAN ID ranges and lists, you must configure a dedicated logical interface for each VLAN. VLAN ID range and list support enables you to associate multiple VLANs with a single logical interface, which reduces the overall number of logical interfaces needed. Using fewer logical interfaces provides these benefits:

- Reduces the amount of configuration time
- Reduces the amount of memory consumed
- Reduces the impact to system performance

VLAN Bundle Service

The VLAN bundle service supports the mapping of multiple broadcast domains (VLANs) to a single bridge domain (MAC learning domain). You can associate multiple VLANs with a single EVPN routing instance. As a result, these broadcast domains (VLANs) share the same MAC table in the EVPN routing instance, thereby reducing the utilization of resources—for example, the number of MAC tables, MAC routes, and labels.

Sample VLAN ID Range and List Configuration

The following sample configuration shows a service provider style interface (interface xe-1/0/0 and logical interfaces xe-1/0/0.0 and xe-1/0/0.1) that is configured on a Juniper Networks device in an EVPN-VXLAN topology. The sample configuration also shows the EVPN routing instance (EVPN-VXLAN-3) in which logical interfaces xe-1/0/0.0 and xe-1/0/0.1 are referenced.

```

interfaces {
  xe-1/0/0 {
    unit 0 {
      encapsulation vlan-bridge;
      vlan-id-range 100-102;
      family bridge;
    }
    unit 1 {
      encapsulation vlan-bridge;
      vlan-id-list [ 200-203 213 248 ];
      family bridge;
    }
  }
}

routing-instances {
  EVPN-VXLAN-3 {
    description "EVPN-VXLAN Vlan Bundle service";
    instance-type evpn;
    vtep-source-instance lo0.0;
    interface xe-1/0/0.0;
    interface xe-1/0/0.1;
    route-distinguisher 10.255.235.35:200;
    vrf-target target:123:123;
    protocols {
      evpn {
        encapsulation vxlan;
        extended-vni-list 551;
      }
    }
  }
}

```

```

    }
  }
  vxlan {
    vni 551;
    encapsulate-inner-vlan;
    decapsulate-accept-inner-vlan;
  }
}
}

```

In this configuration, logical interface xe-1/0/0.0 includes a VLAN ID range and logical interface xe-1/0/0.1 includes a VLAN ID list, which is comprised of a VLAN ID range and individual VLAN IDs. EVPN routing instance EVPN-VXLAN-3 references both logical interfaces.

Caveats and Limitations

When specifying VLAN ID ranges and lists in a service provider style interface configuration in an EVPN environment, keep these caveats and limitations in mind:

- When specifying a range in either a VLAN ID range or list, you must use an ascending range—for example, 100-102. If you specify a descending range—for example, 102-100—the system considers the range to be invalid, and a commit error occurs.

Configuring VLAN ID Lists and Ranges in an EVPN Environment

Starting in Junos OS Release 19.2R1, you can specify VLAN ID lists and ranges in a service provider style of interface configuration that is referenced in an Ethernet VPN (EVPN) routing instance (a routing instance of type evpn).

This feature enables you to associate multiple VLANs with a single logical interface, thereby freeing you from having to configure a dedicated logical interface for each VLAN.

This feature works with the VLAN bundle service.

This procedure shows you how to specify multiple VLANs using VLAN ID ranges and lists in a service provider style interface configuration and to associate the interface with an EVPN routing instance.

The sample configurations that follow the procedure provide more comprehensive configurations of service provider style interfaces in an EVPN environment.

1. Configure the physical interface or aggregated Ethernet interface with the encapsulation type of flexible Ethernet services, which enables you to specify Ethernet encapsulations at the logical interface level.

```
[edit]user@switch# set interfaces interface-name encapsulation flexible-ethernet-services
```

2. Configure a service provider style logical interface.

- a. Specify the encapsulation type of `vlan-bridge` to enable bridging on the logical interface:

```
[edit]user@switch# set interfaces interface-name unit logical-unit-number encapsulation
vlan-bridge
```

- b. Associate the logical interface with multiple VLANs using either a VLAN ID range or list:

```
[edit]user@switch# set interfaces interface-name unit logical-unit-number vlan-id-range
vlan-id-vlan-id
```

OR

```
[edit]user@switch# set interfaces interface-name unit logical-unit-number vlan-id-list
[ vlan-id vlan-id vlan-id-vlan-id ]
```

3. Repeat Step 2 for each additional service provider style logical interface that you need to configure.
4. Create an EVPN routing instance.
 - a. Specify that the routing instance is of type `evpn`.

```
[edit]user@switch# set routing-instances routing-instance-name instance-type evpn
```

- b. Associate the logical interfaces that you configured earlier with the EVPN routing instance.

```
[edit]user@switch# set routing-instances routing-instance-name interface interface-
name.logical-unit-number
```

Sample Configuration: Multiple Logical Interfaces

This sample configuration shows aggregated Ethernet interface `ae0`, which is divided into logical interfaces `ae0.100` and `ae0.150`. Logical interface `ae0.100` is associated with VLANs ranging from 100

through 102. Logical interface ae0.150 is associated with a list of VLANs, which includes 150 through 152, 200, 213, and 248. EVPN routing instance EVPN-1 references both logical interfaces.

```

interfaces {
  ae0 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 100 {
      encapsulation vlan-bridge;
      vlan-id-range 100-102;
      family bridge;
    }
    unit 150 {
      encapsulation vlan-bridge;
      vlan-id-list [ 150-152 200 213 248 ];
      family bridge;
    }
  }
}

routing-instances {
  EVPN-1 {
    instance-type evpn;
    interface ae0.100;
    interface ae0.150;
    route-distinguisher 192.160.0.1:111;
    vrf-target target:65000:111;
    protocols {
      evpn;
    }
  }
}

```

Sample Configuration: Single Logical Interface

This sample configuration is similar to the multiple logical interface sample configuration except that aggregated Ethernet interface ae0 includes only one logical interface (ae0.150) with which all VLANs (100 through 102, 150 through 152, 200, 213, and 248) are associated. EVPN routing instance EVPN-1 references logical interface ae0.150.

```

interfaces {
  ae0 {
    flexible-vlan-tagging;

```

```

        encapsulation flexible-ethernet-services;
        unit 150 {
            encapsulation vlan-bridge;
            vlan-id-list [ 100-102 150-152 200 213 248 ];
            family bridge;
        }
    }

    routing-instances {
        EVPN-1 {
            instance-type evpn;
            interface ae0.150;
            route-distinguisher 192.160.0.1:111;
            vrf-target target:65000:111;
            protocols {
                evpn;
            }
        }
    }
}

```

Sample Configuration: E-Tree

This sample E-Tree configuration is similar to the other sample configurations except for some information specific to E-Tree use (for example, specifying each logical interface as either root or leaf, and enabling the EVPN-ETREE service).

```

interfaces {
    ae0 {
        flexible-vlan-tagging;
        encapsulation flexible-ethernet-services;
        unit 100 {
            encapsulation vlan-bridge;
            vlan-id-range 100-102;
            family bridge;
            etree-ac-role leaf;
        }
        unit 200 {
            encapsulation vlan-bridge;
            vlan-id-list [ 200 213 248 ];
            family bridge;
            etree-ac-role root;
        }
    }
}

```

```

}

routing-instances {
  ETREE-1 {
    instance-type evpn;
    interface ae0.100;
    interface ae0.200;
    route-distinguisher 192.160.0.1:111;
    vrf-target target:65000:111;
    protocols {
      evpn {
        evpn-etree;
      }
    }
  }
}

```

Release History Table

Release	Description
19.2R1	Starting in Junos OS Release 19.2R1, you can specify VLAN ID lists and ranges in a service provider style interface configuration that is referenced in an Ethernet VPN (EVPN) routing instance (a routing instance of type evpn).

RELATED DOCUMENTATION

[Flexible Ethernet Services Encapsulation](#)

Configuring Routing Instances for EVPN

IN THIS CHAPTER

- [Configuring EVPN Routing Instances | 10](#)
- [Configuring EVPN Routing Instances on EX9200 Switches | 13](#)
- [Overview of MAC Mobility | 16](#)
- [Changing Duplicate MAC Address Detection Settings | 19](#)
- [EVPN Type 5 Route with VXLAN encapsulation for EVPN-VXLAN | 21](#)
- [EVPN Type 5 Route with MPLS encapsulation for EVPN-MPLS | 22](#)
- [Understanding EVPN Pure Type 5 Routes | 22](#)
- [EVPN Type 2 and Type 5 Route Coexistence with EVPN-VXLAN | 27](#)
- [Ingress Virtual Machine Traffic Optimization | 36](#)
- [Tracing EVPN Traffic and Operations | 42](#)
- [Migrating From BGP VPLS to EVPN Overview | 44](#)

Configuring EVPN Routing Instances

To configure an EVPN routing instance, complete the following configuration on the PE router (or on the MPLS edge switch or QFX Series switch) within the EVPN service provider's network:

1. Configure the EVPN routing instance name using the `routing-instances` statement at the [edit] hierarchy level:

```
routing-instances routing-instance-name {...}
```

2. Configure the `evpn` option for the `instance-type` statement at the [edit `routing-instances routing-instance-name`] hierarchy level:

```
instance-type evpn;
```

NOTE: For MX Series devices, EX Series, and QFX Series switches, you can include multiple IFLs of an Ethernet segment identifier (ESI) across different bridge-domains or VLANs of an EVPN routing instance in all-active mode. However, you cannot include multiple IFLs of the same ESI within the same bridge-domain or VLAN.

3. Configure the interfaces for handling EVPN traffic between the MES or PEs and the CE device using the interface statement at the [edit routing-instances *routing-instance-name*] hierarchy level:

```
interface interface-name;
```

4. Configure a VLAN identifier for the EVPN routing instance using the vlan-id statement at the [edit routing-instances *routing-instance-name*] hierarchy level:

NOTE: For QFX Series, set the VLAN ID to none.

```
vlan-id (vlan-id | all | none);
```

5. Configure a route distinguisher on a PE router by including the route-distinguisher statement:

```
route-distinguisher (as-number:number | ip-address:number);
```

Each routing instance that you configure on a PE router must have a unique route distinguisher associated with it. VPN routing instances need a route distinguisher to help BGP to distinguish between potentially identical network layer reachability information (NLRI) messages received from different VPNs. If you configure different VPN routing instances with the same route distinguisher, the commit fails.

For a list of the hierarchy levels at which you can include this statement, see the statement summary for this statement.

The route distinguisher is a 6-byte value that you can specify in one of the following formats:

- *as-number:number*, where *as-number* is an autonomous system (AS) number (a 2-byte value) and *number* is any 4-byte value. The AS number can be in the range 1 through 65,535. We recommend that you use an Internet Assigned Numbers Authority (IANA)-assigned, nonprivate AS number, preferably the Internet service provider's (ISP's) own or the customer's own AS number.

NOTE: The automatic derivation of the BGP route target (auto-RT) for advertised prefixes is supported on a 2-byte AS number only.

- *ip-address:number*, where *ip-address* is an IP address (a 4-byte value) and *number* is any 2-byte value. The IP address can be any globally unique unicast address. We recommend that you use the address that you configure in the *router-id* statement, which is a nonprivate address in your assigned prefix range.
6. Configure either import and export policies for the EVPN routing table, or configure the default policies using the *vrf-target* statement configured at the [edit routing-instances *routing-instance-name*] hierarchy level.
See *Configuring Policies for the VRF Table on PE Routers in VPNs*.
 7. Configure each EVPN interface for the EVPN routing instance:
 - Configure each interface using the ["interface" on page 1599](#) statement at the [edit routing-instances *routing-instance-name* protocols *evpn*] hierarchy level.
 - Configure interface encapsulation for the CE facing interfaces at the [edit interfaces *interface-name* encapsulation] hierarchy level. Supported encapsulations, except for EX9200 switches and QFX Series switches, are: (ethernet-bridge | vlan-bridge | extended-vlan-bridge). Supported encapsulations for EX9200 switches are: (extended-vlan-bridge | flexible-ethernet-services). Supported encapsulation for QFX Series switches is vxlan.
 - (Optional) Allow the EVPN to establish a connection to the CE device even if the CE device interface encapsulation and the EVPN interface encapsulations do not match by including the *ignore-encapsulation-mismatch* statement at the [edit routing-instances *routing-instance-name* protocols *evpn* interface *interface-name*] hierarchy level.
 - (Optional) (Not available on EX9200 switches) Specify a static MAC address for a logical interface in a bridge domain using the *static-mac* statement at the [edit routing-instances *routing-instance-name* protocols *evpn* interface *interface-name*] hierarchy level.
 8. Specify the maximum number of media access control (MAC) addresses that can be learned by the EVPN routing instance by including the ["interface-mac-limit" on page 1602](#) statement.

You can configure the same limit for all interfaces configured for a routing instance by including this statement at the [edit routing-instances *routing-instance-name* protocols *evpn*] hierarchy level. You can also configure a limit for a specific interface by including this statement at the [edit routing-instances *routing-instance-name* protocols *evpn* interface *interface-name*] hierarchy level.

By default, packets with new source MAC addresses are forwarded after the MAC address limit is reached. You can alter this behavior by including the *packet-action* drop statement at either the [edit routing-instances *routing-instance-name* protocols *evpn* interface-mac-limit] or the [edit routing-instances *routing-instance-name* protocols *evpn* interface *interface-name*] hierarchy level. If you configure this

statement, packets from new source MAC addresses are dropped once the configured MAC address limit is reached.

9. Specify the MPLS label allocation setting for the EVPN by including the *label-allocation* statement with the per-instance option at the [edit routing-instances *routing-instance-name* protocols evpn] hierarchy level.

If you configure this statement, one MPLS label is allocated for the specified EVPN routing instance.

10. Enable MAC accounting for the EVPN by including the *mac-statistics* statement at the [edit routing-instances *routing-instance-name* protocols evpn] hierarchy level.
11. Specify the number of addresses that can be stored in the MAC routing table using the *mac-table-size* statement at the [edit routing-instances *routing-instance-name* protocols evpn] hierarchy level.

You can optionally configure the *packet-action* drop option to specify that packets for new source MAC addresses be dropped once the MAC address limit is reached. If you do not configure this option, packets for new source MAC addresses are forwarded.

12. Disable MAC learning by including the *no-mac-learning* statement at either the [edit routing-instances *routing-instance-name* protocols evpn] hierarchy level to apply this behavior to all of the devices configured for an EVPN routing instance or at the [edit routing-instances *routing-instance-name* protocols evpn interface *interface-name*] hierarchy level to apply this behavior to just one of the CE devices.

RELATED DOCUMENTATION

Configuring Policies for the VRF Table on PE Routers in VPNs

Configuring Routing Instances on PE Routers in VPNs

[Tracing EVPN Traffic and Operations | 42](#)

Configuring EVPN Routing Instances on EX9200 Switches

To configure an EVPN routing instance, complete the following configuration on the PE router (or on the MPLS edge switch) within the EVPN service provider's network:

1. Configure the EVPN routing instance name using the routing-instances statement at the [edit] hierarchy level:

```
routing-instances routing-instance-name {...}
```

- 2.

Configure the `evpn` option for the `instance-type` statement at the `[edit routing-instances routing-instance-name]` hierarchy level:

```
instance-type evpn;
```

3. Configure the interfaces for handling EVPN traffic between the MES and the CE device using the `interface` statement at the `[edit routing-instances routing-instance-name]` hierarchy level:

```
interface interface-name;
```

4. Configure a VLAN identifier for the EVPN routing instance using the `vlan-id` statement at the `[edit routing-instances routing-instance-name]` hierarchy level:

```
vlan-id (vlan-id | all | none);
```

5. Configure a route distinguisher on a PE router by including the `route-distinguisher` statement:

```
route-distinguisher (as-number:number | ip-address:number);
```

Each routing instance that you configure on a PE router must have a unique route distinguisher associated with it. VPN routing instances need a route distinguisher to help BGP to distinguish between potentially identical network layer reachability information (NLRI) messages received from different VPNs. If you configure different VPN routing instances with the same route distinguisher, the commit fails.

For a list of the hierarchy levels at which you can include this statement, see the statement summary for this statement.

The route distinguisher is a 6-byte value that you can specify in one of the following formats:

- *as-number:number*, where *as-number* is an autonomous system (AS) number (a 2-byte value) and *number* is any 4-byte value. The AS number can be in the range 1 through 65,535. We recommend that you use an Internet Assigned Numbers Authority (IANA)-assigned, nonprivate AS number, preferably the Internet service provider's (ISP's) own or the customer's own AS number.
 - *ip-address:number*, where *ip-address* is an IP address (a 4-byte value) and *number* is any 2-byte value. The IP address can be any globally unique unicast address. We recommend that you use the address that you configure in the `router-id` statement, which is a nonprivate address in your assigned prefix range.
6. Configure either import and export policies for the EVPN routing table, or configure the default policies using the `vrf-target` statement configured at the `[edit routing-instances routing-instance-name]` hierarchy level.

See *Configuring Policies for the VRF Table on PE Routers in VPNs*.

7. Configure each EVPN interface for the EVPN routing instance:

- Configure interface encapsulation for the CE facing interfaces at the [edit interfaces *interface-name* encapsulation] hierarchy level. . Supported encapsulations for EX9200 switches are: (extended-vlan-bridge | flexible-ethernet-services | vlan-bridge).
- Configure vlan-bridge encapsulation on the logical interface at the [edit interfaces *interface-name* flexible-vlan-tagging encapsulation flexible-ethernet-services unit 0 encapsulation] hierarchy level.
- (Optional) Allow the EVPN to establish a connection to the CE device even if the CE device interface encapsulation and the EVPN interface encapsulations do not match by including the *ignore-encapsulation-mismatch* statement at the [edit routing-instances *routing-instance-name* protocols evpn interface *interface-name*] hierarchy level.

8. Specify the maximum number of media access control (MAC) addresses that can be learned by the EVPN routing instance by including the ["interface-mac-limit" on page 1602](#) statement.

You can configure the same limit for all interfaces configured for a routing instance by including this statement at the [edit routing-instances *routing-instance-name* protocols evpn] hierarchy level. You can also configure a limit for a specific interface by including this statement at the [edit routing-instances *routing-instance-name* protocols evpn interface *interface-name*] hierarchy level.

By default, packets with new source MAC addresses are forwarded after the MAC address limit is reached. You can alter this behavior by including the *packet-action* drop statement at either the [edit routing-instances *routing-instance-name* protocols evpn interface-mac-limit] or the [edit routing-instances *routing-instance-name* protocols evpn interface *interface-name*] hierarchy level. If you configure this statement, packets from new source MAC addresses are dropped once the configured MAC address limit is reached.

9. Enable MAC accounting for the EVPN by including the *mac-statistics* statement at the [edit routing-instances *routing-instance-name* protocols evpn] hierarchy level.
10. Specify the number of addresses that can be stored in the MAC routing table using the *mac-table-size* statement at the [edit routing-instances *routing-instance-name* protocols evpn] hierarchy level.

You can optionally configure the *packet-action* drop option to specify that packets for new source MAC addresses be dropped once the MAC address limit is reached. If you do not configure this option, packets for new source MAC addresses are forwarded.

11. Disable MAC learning by including the *no-mac-learning* statement at either the [edit routing-instances *routing-instance-name* protocols evpn] hierarchy level to apply this behavior to all of the devices configured for an EVPN routing instance or at the [edit routing-instances *routing-instance-name* protocols evpn interface *interface-name*] hierarchy level to apply this behavior to just one of the CE devices.

RELATED DOCUMENTATION

Configuring Policies for the VRF Table on PE Routers in VPNs

Configuring Routing Instances on PE Routers in VPNs

[Tracing EVPN Traffic and Operations | 42](#)

Overview of MAC Mobility

MAC mobility describes the scenario where a host moves from one Ethernet segment to another segment in the EVPN network. Provider Edge (PE) devices discover the host MAC address from its local interfaces or from remote PE devices. When a PE device learns of a new local MAC address, it sends a MAC advertisement route message to other devices in the network. During this time, there are two advertised routes and the PE devices in the EVPN network must decide which of the MAC advertisement messages to use.

To determine the correct MAC address location, PE devices use the MAC mobility extended community field, as defined in RFC 7432, in the MAC advertisement route message. The MAC mobility extended community includes a static flag and a sequence number. The static flag identifies pinned MAC addresses that should not be relocated. The sequence number identifies newer MAC advertisement messages. Starting at 0, the sequence number is incremented for every MAC address mobility event. PE devices running Junos OS apply the following precedence order in determining the MAC advertisement route to use:

1. Advertisement routes with a local pinned MAC address (static MAC address).
2. Advertisement routes with a remote pinned MAC address (static MAC address).
3. Advertisement routes with a higher sequence number.

NOTE: When there are two advertisement route messages for pinned MAC addresses with different routes or two advertisement route messages with the same sequence number, the local device chooses the advertisement route message from the PE device with the lower IP address.

[Figure 1 on page 17](#) illustrates a network where a MAC address is relocated from PE1 to PE2. Before the move, a MAC advertisement route message sent by PE1 has the active route for all PE devices in the network. After the relocation, PE2 learns of the new local MAC address and sends an updated MAC advertisement route message. [Table 1 on page 17](#) lists the action taken by each PE device based on the two MAC advertisements. The PE device generates a syslog message when it encounters conflicts with a pinned MAC address.

NOTE: Table 1 on page 17 includes use cases with pinned MAC addresses. These use cases do not apply to PE devices that do not support MAC pinning. To determine whether or not MAC pinning is supported by a particular Juniper Networks device or Junos OS release, see [Feature Explorer](#).

Figure 1: MAC Mobility in an EVPN Network

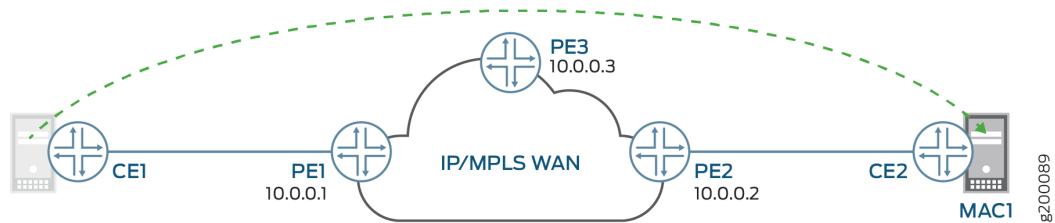


Table 1: MAC Advertisement Routes on PE Devices

MAC Advertisement	PE1	PE2	PE3
PE1: MAC address with a sequence number (n). PE2: MAC address with the sequence number incremented by one (n+1).	Install the remote MAC advertisement route from PE2 because it has a higher sequence number (n+1).	Advertise the local MAC route because it has a higher sequence number (n+1).	Install the remote MAC advertisement route from PE2 because it has a higher sequence number (n+1).
PE1: MAC address with a sequence number (n). PE2: MAC address with the same sequence number (n).	Advertise the local MAC route because PE1 has the lower IP address (10.0.0.1).	Install the remote MAC advertisement route from PE1 because PE1 has the lower IP address (10.0.0.1).	Use the MAC advertisement route from PE1 because PE1 has the lower IP address (10.0.0.1).

Table 1: MAC Advertisement Routes on PE Devices *(Continued)*

MAC Advertisement	PE1	PE2	PE3
PE1: Pinned MAC address with the static bit set. PE2: MAC address and a sequence number (n).	Advertise the local MAC route because it is a pinned MAC address. Generate a syslog message.	Install the remote MAC advertisement route from PE1 because it is a pinned MAC address.	Use the MAC advertisement route from PE1 because it is a pinned MAC address. Generate a syslog message.
PE1: MAC address with a sequence number (n). PE2: Pinned MAC address with the static bit set.	Install the remote the MAC advertisement route from PE2 because it is a pinned MAC address.	Advertise local MAC route because it is a pinned MAC address. Generate a syslog message.	Install the remote MAC advertisement route from PE2 because it is a pinned MAC address.
PE1: Pinned MAC address with static bit set. PE2: Pinned MAC address with static bit set.	Advertise the local MAC route because it is a local pinned MAC address. Generate a syslog message.	Advertise the local MAC route because it is a local pinned MAC address. Generate a syslog message.	Use the MAC advertisement route from PE1 because PE1 has the lower IP address (10.0.0.1). Generate a syslog message.

Junos supports MAC mobility automatically by default. To disable MAC mobility, use the `set protocols evpn mac-mobility no-sequence-numbers` statement.

RELATED DOCUMENTATION

[EVPN MAC Pinning Overview](#) | 377

[clear evpn duplicate-mac-suppression](#) | 1729

[duplicate-mac-detection](#) | 1555

Changing Duplicate MAC Address Detection Settings

When a host is physically moved or when a host is reconfigured on a different Ethernet segment, the PE device sends an updated MAC advertisement route to other PE devices to update their route table. If there is a misconfiguration in the network, MAC advertisement messages oscillate between the different routes causing MAC address flapping. This makes the network more vulnerable and wastes network resources. Junos supports MAC mobility automatically by default. To disable MAC mobility, use the `set protocols evpn mac-mobility no-sequence-numbers` statement.

Junos OS also automatically detects and suppresses duplicate MAC addresses. Optionally, you can also configure the length of time that the duplicate MAC address is suppressed. When the PE device encounters duplicate MAC addresses, Junos OS generates a syslog message.

To change the duplicate MAC address detection settings, include the `duplicate-mac-detection` statement at either the `[edit routing-instances routing-instance-name protocols]` hierarchy level or the `[edit logical-systems logical-system-name routing-instances routing-instance-name protocols]` hierarchy level:

```
evpn
  duplicate-mac-detection {
    detection-threshold detection-threshold;
    detection-window seconds;
    auto-recovery-time minutes;
  }
```

You can modify the following options under the `duplicate-mac-detection` statement:

- `detection-window`—The time interval used in detecting a duplicate MAC address. The value can be from 5 through 600 seconds. The default is 180 seconds
- `detection-threshold`—The number of MAC mobility events that are detected for a given MAC address within the `detection-window` before it is identified as a duplicate MAC address. Once the detection threshold is reached, updates for the MAC address are suppressed. The value can be from 2 through 20. The default is 5.
- `auto-recovery-time`—(Optional) The length of time a device suppresses a duplicate MAC address. At the end of this duration, MAC address updates will resume. The value can be from 1 through 360 minutes. If a value is not specified, then the MAC address continues to be suppressed.

NOTE: To ensure that the mobility advertisements have sufficient time to age out, set an `auto-recovery-time` greater than the `detection-window`.

To manually clear the suppression of duplicate MAC addresses, use the `clear evpn duplicate-mac-suppression` command.

To view MAC duplicate addresses in the EVPN MAC database, use the `show evpn database` command. The following example displays a sample output. The output fields related to duplicate MAC detections are State, Mobility history, and MAC advertisement route status:

```
user@PE1> show evpn database mac-address 00:00:00:00:00:02
extensive

Instance: ALPHA

VLAN ID: 100, MAC address: 00:00:00:00:00:02
State: 0x1 <Duplicate-Detected>
Mobility history
  Mobility event time      Type      Source                               Seq num
  Aug 03 17:22:28.585619  Local    ge-0/0/2.0                         31
  Aug 03 17:22:30.307198  Remote   10.255.0.3                         32
  Aug 03 17:22:37.611786  Local    ge-0/0/2.0                         33
  Aug 03 17:22:39.289357  Remote   10.255.0.3                         34
  Aug 03 17:22:45.609449  Local    ge-0/0/2.0                         35
Source: ge-0/0/2.0, Rank: 1, Status: Active
  Mobility sequence number: 35 (minimum origin address 10.255.0.2)
  Timestamp: Aug 03 17:22:44 (0x5983be54)
  State: <Local-MAC-Only Local-To-Remote-Adv-Allowed>
  MAC advertisement route status: Not created (duplicate MAC suppression)
  IP address: 10.0.0.2
Source: 10.255.0.3, Rank: 2, Status: Inactive
  MAC label: 300176
  Mobility sequence number: 34 (minimum origin address 10.255.0.3)
  Timestamp: Aug 03 17:22:39 (0x5983be4f)
  State: <>
  MAC advertisement route status: Not created (inactive source)
  IP address: 10.0.0.3
```

RELATED DOCUMENTATION

[clear evpn duplicate-mac-suppression](#) | 1729

EVPN Type 5 Route with VXLAN encapsulation for EVPN-VXLAN

EVPN is a flexible solution that uses Layer 2 overlays to interconnect multiple edges (virtual machines) within a data center. Traditionally, the data center is built as a flat Layer 2 network with issues such as flooding, limitations in redundancy and provisioning, and high volumes of MAC addresses learned, which cause churn at node failures. EVPN is designed to address these issues without disturbing flat MAC connectivity.

VXLAN is an overlay technology that encapsulates MAC frames into a UDP header at Layer 2. Communication is established between two virtual tunnel endpoints (VTEPs). VTEPs encapsulate the virtual machine traffic into a VXLAN header, as well as strip off the encapsulation. Virtual machines can only communicate with each other when they belong to the same VXLAN segment. A 24-bit virtual network identifier (VNID) uniquely identifies the VXLAN segment. This enables having the same MAC frames across multiple VXLAN segments without traffic crossover. Multicast in VXLAN is implemented as Layer 3 multicast, in which endpoints subscribe to groups.

When a Bridge Domain (BD) is not L2 extended across Data Centers (DC), the IP subnet belonging to the BD is confined within a single DC. If all BDs within each DC network satisfy this requirement, there is no longer a need to advertise MAC+IP route for each tenant between data centers as host routes for the tenants can be aggregated. Thus the L2 inter-DC connectivity issue can be simply transformed to an inter-DC L3 IP prefix reachability issue.

Starting with Junos OS Release 17.1, the EVPN Type 5 IP prefix route advertises the IP prefixes between the DCs. Unlike the type-2 EVPN MAC advertisement route, the EVPN Type 5 IP prefix route separates the host MAC address from its IP address and provides a clean advertisement of an IP prefix for the bridge domain.

Junos OS Release 17.1 also supports:

- Inter-DC connectivity with VXLAN encapsulation for EVPN/VXLAN by using the EVPN type 5 IP prefix route. Each BD within a DC is not L2 extended. If EVPN/VXLAN is enabled between DC GW (Data Center Gateway) router and ToR while providing inter-DC connectivity, the spine, which acts as a DC GW router, is capable of performing L3 routing and IRB functions.
- Inter-pod connectivity with VXLAN encapsulation by using the EVPN Type 5 IP prefix route. The solution provided does not address the L2 extension problem when a BD is stretched across different pods. The spines that provide the inter-pod connectivity is able to perform L3 routing and IRB functions.

RELATED DOCUMENTATION

[EVPN Type 5 Route with MPLS encapsulation for EVPN-MPLS](#) | 22

EVPN Type 5 Route with MPLS encapsulation for EVPN-MPLS

EVPN is a flexible solution that uses Layer 2 overlays to interconnect multiple edges (virtual machines) within a data center. Traditionally, the data center is built as a flat Layer 2 network with issues such as flooding, limitations in redundancy and provisioning, and high volumes of MAC addresses learned, which cause churn at node failures. EVPN is designed to address these issues without disturbing flat MAC connectivity.

The MPLS infrastructure allows you to take advantage of the MPLS functionality provided by the Junos operating system (Junos OS), including fast reroute, node and link protection, and standby secondary paths.

Starting with Junos OS Release 17.1, to support the interconnection through EVPN Type 5 route in the metro service application using EVPN/MPLS, an MPLS tunnel is required instead of a VXLAN tunnel.

NOTE: L3VPN forwarding based on pure Type 5 without overlay next-hop is only supported.

In the control plane EVPN Type 5 is used to advertise IP prefix for inter-subnet connectivity across metro peering points. To reach the end host through the connectivity provided by the EVPN Type 5 prefix route, data packets are sent out as an IP packet encapsulated in the MPLS across the metro peering points.

RELATED DOCUMENTATION

[EVPN Type 5 Route with VXLAN encapsulation for EVPN-VXLAN](#) | 21

Understanding EVPN Pure Type 5 Routes

IN THIS SECTION

● [Defining EVPN-VXLAN Route Types](#) | 23

- Implementing Pure Type 5 Routes in an EVPN-VXLAN Environment | 24
- Best Practices and Caveats | 26

Ethernet VPN (EVPN) offers an end-to-end solution for data center Virtual Extensible LAN (VXLAN) networks. A main application of EVPN is Data Center Interconnect (DCI), which provides the ability to extend Layer 2 connectivity between different data centers. EVPN uses the concept of route types to establish sessions between the provider edge and the customer edge. There are many route types. A Type 5 route, also called the IP prefix route, is used to communicate between data centers (DC) when the Layer 2 connection does not extend across DCs and the IP subnet in a Layer 2 domain is confined within a single DC. In this scenario, the Type 5 route enables connectivity across DCs by advertising the IP prefixes assigned to the VXLANs confined within a single DC. Data packets are sent as Layer 2 Ethernet frames encapsulated in the VXLAN header. Additionally, the gateway device for the DC must be able to perform Layer 3 routing and provide IRB functionality.

A pure Type 5 route operates without an overlay next hop or a Type 2 route for recursive route resolution. With pure Type 5 routing, the Type 5 route is advertised with the MAC extended community so that the Type 5 route provides all necessary forwarding information required for sending VXLAN packets in the data plane to the egress network virtual endpoint. There is no need to use an IP address as an overlay next hop to interconnect Layer 3 virtual routing and forwarding (VRF) routes sitting in different data centers. Because no Type 2 routes are used for route recursive resolution, this provisioning model is also called the IP-VRF-to-IP-VRF model without a core-facing IRB interface.

Defining EVPN-VXLAN Route Types

The EVPN-VXLAN route types are:

- Type 1 route, Ethernet autodiscovery route—Type 1 routes are for networkwide messages. Ethernet autodiscovery routes are advertised on a per end virtual identifier (EVI) and per Ethernet segment identifier (ESI) basis. The Ethernet autodiscovery routes are required when a customer edge (CE) device is multihomed. When a CE device is single-homed, the ESI is zero. This route type is supported by all EVPN switches and routers.

An ESI can participate in more than one broadcast domain; for example, when a port is trunked. An ingress provider edge (PE) device that reaches the MAC on that ESI must have Type 1 routes to perform split horizon and fast withdraw. Therefore, a Type 1 route for an ESI must reach all ingress PE devices importing a virtual network identifier (VNI) or tag (broadcast domains) in which that ESI is a member. The Junos OS supports this by exporting a separate route target for the Type 1 route.

- Type 2 route, MAC with IP advertisement route—Type 2 routes are per-VLAN routes, so only PEs that are part of a VNI need these routes. EVPN allows an end host's IP and MAC addresses to be

advertised within the EVPN Network Layer reachability information (NLRI). This allows for control plane learning of ESI MAC addresses. Because there are many Type 2 routes, a separate route-target auto-derived per VNI helps to confine their propagation. This route type is supported by all EVPN switches and routers.

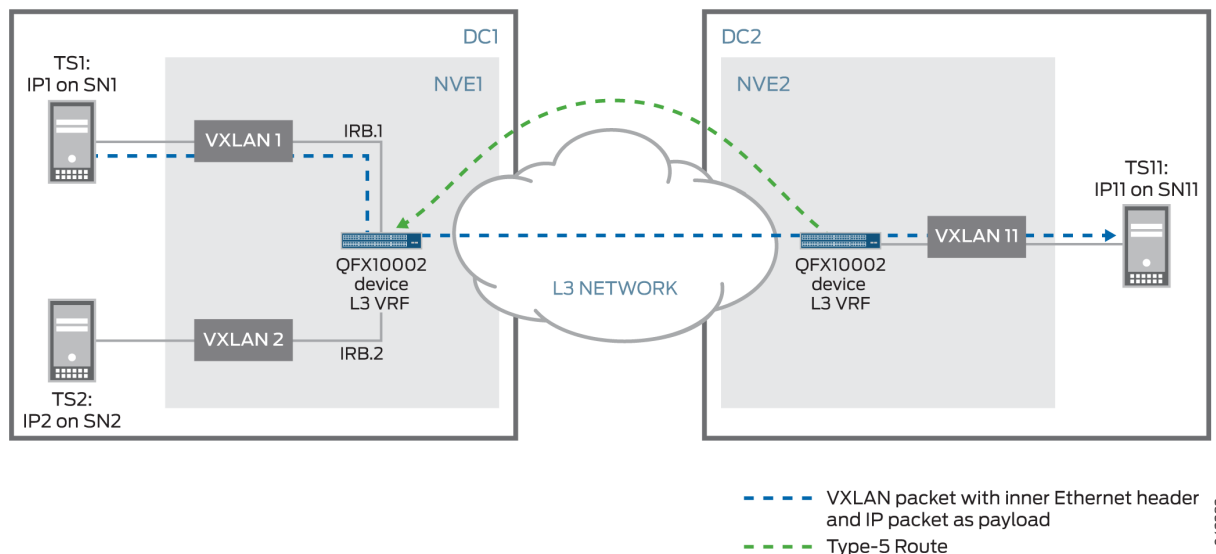
- Type 3 route, inclusive multicast Ethernet tag route—Type 3 routes are per-VLAN routes; therefore, only PE devices that are part of a VNI need these routes. An inclusive multicast Ethernet tag route sets up a path for broadcast, unknown unicast, and multicast (BUM) traffic from a PE device to the remote PE device on a per-VLAN, per-EVI basis. Because there are many Type 3 routes, a separate route-target auto-derived per VNI helps in confining their propagation. This route type is supported by all EVPN switches and routers.
- Type 4 route, Ethernet segment Route—An Ethernet segment identifier (ESI) allows a CE device to be multihomed to two or more PE devices—in single/active or active/active mode. PE devices that are connected to the same Ethernet segment discover each other through the ESI. This route type is supported by all EVPN switches and routers.
- Type 5 route, IP prefix Route—An IP prefix route provides encoding for inter-subnet forwarding. In the control plane, EVPN Type 5 routes are used to advertise IP prefixes for inter-subnet connectivity across data centers. To reach a tenant using connectivity provided by the EVPN Type 5 IP prefix route, data packets are sent as Layer 2 Ethernet frames encapsulated in the VXLAN header over the IP network across the data centers.
- Type 6 route, selective multicast Ethernet tag routes.
- Type 7 route, network layer reachability information (NLRI) to sync IGMP joins.
- Type 8 route, NLRI to sync IGMP leaves.

Implementing Pure Type 5 Routes in an EVPN-VXLAN Environment

You can use EVPN pure Type 5 routes on QFX10000 switches to communicate between data centers through a Layer 3 network. See [Figure 2 on page 25](#). A unified EVPN control plane accomplishes L3 route advertisement between multiple data center locations so that you do not have to rely on an additional L3 VPN protocol family. On the customer edge (CE), hosts such as servers, storage devices, or

any bare-metal devices are attached to leaf switches on the provider edge. Between those leaf devices, an MP-BGP session is established for EVPN routes to be used in the overlay control protocol. .

Figure 2: EVPN-VXLAN Connection with Pure Type 5 Route Between Two Data Centers



A global unique virtual network identifier (VNI) is provisioned for each customer L3 VRF and identifies the customer L3 VRF at the egress. A chassis MAC is used as the inner destination MAC (DMAC) for the VXLAN packet. The chassis MAC is shared among different customer L3 VRF instances

NOTE: When a virtual machine (VM) moves from one QFX10000 data center to another, a Type 5 route no longer works. This is because both the VXLAN and IP subnet that belong to the VM are no longer confined within a single data center.

NOTE: For an example of communicating within a single data center without Type 5 routing, see ["Example: Configure an EVPN-VXLAN Centrally-Routed Bridging \(CRB\) Overlay" on page 530.](#)

Understanding Pure Type 5-Route Forwarding

Pure Type 5 route forwarding is also called the IP-VRF-to-IP-VRF (virtual routing and forwarding) model. In IP-based computer networks, Layer 3 VRF allows multiple instances of a routing table to coexist within the same router at the same time. Because the routing instances are independent, the same or overlapping IP addresses can be used without conflicting with each other. In this scenario, for a given tenant, such as an IP VPN service, a network virtualization edge (NVE) has one MAC VRF, which

consists of multiple VXLANs (one VXLAN per VLAN). The MAC VRFs on an NVE for a given tenant are associated with an IP VRF corresponding to that tenant (or IP VPN service) through their IRB interfaces. A global unique VNI is provisioned for each customer Layer 3 VRF. The VNI is used to identify the Layer 3 VRF for the customer on each data center.

Understanding EVPN Pure Type 5 Routes and Local Preferences

On QFX10000 switches running Junos OS Release 15.1X53-D65 or later, the local preference setting for an Ethernet VPN (EVPN) pure Type 5 route is inherited by IP routes that are derived from the EVPN type 5 route. Further, when selecting an IP route for incoming traffic, the QFX10000 switches consider the local preference of the route. A benefit of the QFX10000 switches including local preference in their route selection criteria is that you can set up a policy to manipulate the local preference, thereby controlling which route the switch selects.

Advantages of Using EVPN Pure Type 5 Routing

There are two main advantages to using EVPN pure Type 5 routing:

- There is no need to exchange all host routes between data center locations. This results in smaller requirements for the routing information base (RIB), also known as the routing table, and the forwarding information base (FIB), also known as the forwarding table, on DCI equipment.
- There is no need to use multiple protocol families, such as both EVPN and an L3 VPN, to advertise L2 and L3 reachability information.

Best Practices and Caveats

BEST PRACTICE: You can use pure Type 5 route within a single data center to interconnect points of delivery (pods) as long as the IP prefix can be confined within the pod.

BEST PRACTICE: Note that there are differences between EVPN VXLAN and EVPN MPLS. EVPN VXLAN exports a separate route target for Type 1 routes. EVPN-MPLS exports the Type 1 route with the collective set of route-targets of the VNI or tags (broadcast domains) in which the Ethernet segment identifier is participating.

NOTE: You cannot use Contrail with pure Type 5 route.

Release History Table

Release	Description
17.4R1	Starting with Junos OS Release 17.4R1, pure Type 5 routes are supported on standalone QFX5110 switches only.
15.1X53D60	Starting with Junos OS Release 15.1X53-D60, pure Type 5 routes are also supported on QFX10008 and QFX10016 switches.
15.1X53-D30	Only pure Type 5 routes are supported. Support was added in Junos OS Release 15.1X53-D30 for QFX10002 switches only.

RELATED DOCUMENTATION

[Understanding EVPN with VXLAN Data Plane Encapsulation | 398](#)

[ip-prefix-routes | 1606](#)

[ip-prefix-support | 1610](#)

EVPN Type 2 and Type 5 Route Coexistence with EVPN-VXLAN

SUMMARY

Learn how a device in an EVPN-VXLAN fabric gives preference to either an EVPN Type 2 route or an EVPN Type 5 route when the device learns and advertises both types of routes.

IN THIS SECTION

- [Benefits | 28](#)
- [EVPN Type 2 and Type 5 Coexistence Preference Algorithm | 28](#)
- [Type 5 Route Options and Policy Options | 29](#)
- [CLI Commands to Verify Preferred Route Type | 30](#)

A device in an EVPN-VXLAN edge-routed bridging fabric imports and advertises EVPN Type 2 MAC+IP routes by default. You can also configure the device to import and to advertise EVPN Type 5 IP prefix routes. The device treats either type of route as a unique route, even when it corresponds to the same destination host. Each unique host route uses a dedicated next hop in the Packet Forwarding Engine (PFE). As a result, having both types of routes enabled together puts a strain on the PFE's next-hop

resources. Also, traffic generally flows more efficiently if the device uses one type of route over the other in certain cases.

Starting in Junos OS Release 21.2R1, the device can maintain both types of routes together in a routing instance. If the device has both route types for the same destination, it uses a preference algorithm to choose and to store the preferred route.

Benefits

- Supports higher scaling in edge-routed bridging fabrics because the preference algorithm reduces next-hop resource requirements in the PFE
- Improves bridging performance by choosing the more efficient path for locally learned (Type 2) routes in an Ethernet segment

EVPN Type 2 and Type 5 Coexistence Preference Algorithm

In an EVPN-VXLAN fabric, devices use Type 2 routes by default. You must explicitly enable the device to also import and advertise EVPN Type 5 routes in a virtual routing and forwarding (VRF) instance. To enable Type 5 routes and the coexistence preference algorithm, configure the `ip-prefix-routes` statement at the `[edit routing-instances name protocols evpn]` hierarchy level.

With Type 5 routes enabled, the device might learn a Type 2 route for which it also has a Type 5 route for the same prefix. In that case, the device uses the coexistence preference algorithm to choose only one of the two routes to store as the preferred route.

The preference algorithm works as follows:

- For any destinations for which the device has no Type 5 route, the device uses Type 2 routes.
- If the device has a Type 5 route with a matching prefix for a local ESI Type 2 route, it installs the Type 2 route.
- Otherwise the device prefers the Type 5 route for all other destinations.

With this algorithm, EVPN-VXLAN devices generally prefer Type 5 routes for traffic in or between data centers and in or between VLANs (bridge domains). Type 5 routes are ideal for the purpose of VXLAN overlay routing. However, a device can forward packets more efficiently using Type 2 routes in some cases. One example is when the device learns Type 2 MAC+IP routes for destinations that it only needs to bridge locally. The preference algorithm accounts for that case.

The device does not use MBB actions to effect preference changes due to configuration updates. When making a route preference change, the device first deletes the non-preferred route, then adds the new preferred route. Examples of configuration changes that can cause preference changes include:

- If you didn't enable Type 5 routes, and then you enable Type 5 routes.

- When you configure an ESI locally, in which case the device prefers the MAC+IP Type 2 route.

Type 5 Route Options and Policy Options

In your EVPN-VXLAN configuration, use one of these advertising mode options with the `ip-prefix-routes` statement when you enable Type 5 routes:

- `advertise direct-nexthop`—With this option, the device advertises all non-host routes for a VRF as Type 5 advertisements.
- `advertise gateway-address`—With this option, the device only advertises prefixes for the IRB interfaces for the extended EVPN instances and bridge domains.

You can also include policy options in your EVPN-VXLAN Type 5 route configuration to refine the routes the device actually imports or advertises.

For example, you can define:

- A policy to export all local routes and EVPN routes, such as the following

```
set policy-options policy-statement type5_policy term 1 from protocol local
set policy-options policy-statement type5_export_policy term 1 then accept
set policy-options policy-statement type5_export_policy term 2 from protocol evpn
set policy-options policy-statement type5_export_policy term 2 from route-filter 0.0.0.0/0
prefix-length-range /32-/32
set policy-options policy-statement type5_export_policy term 2 then accept
set policy-options policy-statement type5_export_policy term 3 from protocol evpn
set policy-options policy-statement type5_export_policy term 3 from route-filter
00:00:00:00:00:00:00:00/0 prefix-length-range /128-/128
set policy-options policy-statement type5_export_policy term 3 then accept
```

- A policy that only advertises routes for specific host addresses or prefixes, such as the following:

```
set policy-options policy-statement type5_policy term 1 from protocol local
set policy-options policy-statement type5_policy term 1 from route-filter 10.0.2.0/24 orlonger
set policy-options policy-statement type5_policy term 1 then accept
set policy-options policy-statement type5_policy term 2 from protocol local
set policy-options policy-statement type5_policy term 2 from route-filter
2001::192:101:1:100/48 orlonger
set policy-options policy-statement type5_policy term 2 then accept
```


- A policy that filters and doesn't import routes for specific host addresses or prefixes, such as the following:

```
set policy-options policy-statement type5_import_policy term 1 from protocol bgp
set policy-options policy-statement type5_import_policy term 1 from route-filter 10.0.2.0/24
orlonger
set policy-options policy-statement type5_import_policy term 1 then reject
set policy-options policy-statement type5_import_policy term 2 from protocol bgp
set policy-options policy-statement type5_import_policy term 2 from route-filter
2001::192:101:1:100/48 orlonger
set policy-options policy-statement type5_import_policy term 2 then reject
```

Include the desired policy options in your ip-prefix-routes configuration. For example:

```
set routing-instances vrf1 protocols evpn ip-prefix-routes advertise direct-nexthop
set routing-instances vrf1 protocols evpn ip-prefix-routes encapsulation vxlan
set routing-instances vrf1 protocols evpn ip-prefix-routes vni 100
set routing-instances vrf1 protocols evpn ip-prefix-routes import type5_import_policy
set routing-instances vrf1 protocols evpn ip-prefix-routes export type5_export_policy
set routing-instances vrf1 instance-type vrf
set routing-instances vrf1 interface irb.1
set routing-instances vrf1 interface irb.2
set routing-instances vrf1 route-distinguisher 192.168.0.1:100
set routing-instances vrf1 vrf-target target:100:200
```

CLI Commands to Verify Preferred Route Type

You can use the CLI commands in this section to see the preferred route type.

show ethernet-switching mac-ip-table

The ["show ethernet-switching mac-ip-table" on page 1748](#) CLI command displays the value RTS in the MAC IP flags column when the switch "skips" adding a learned Type 2 route. This value means the device prefers a Type 5 route with a matching prefix. The command displays the definition Dest Route Skipped for the value RTS at the top of the output.

The command displays results for all VLANs (bridge domains) by default. You can also specify a particular VLAN (bridge domain). By default, the command displays information for the default-switch instance. You can also use other command line options to display results for a specific routing instance if the device has multiple routing instances.

The following sample output shows MAC+IP table information for the default-switch instance where the device preferred Type 5 routes for some destinations:

```
user@device> show ethernet-switching mac-ip-table
```

MAC IP flags (S - Static, D - Dynamic, L - Local , R - Remote, Lp - Local Proxy,
Rp - Remote Proxy, K - Kernel, RT - Dest Route, (N)AD - (Not) Advt to remote,
RE - Re-ARP/ND, RO - Router, OV - Override, Ur - Unresolved,
RTS - Dest Route Skipped, RGW - Remote Gateway)

Routing instance : default-switch

Bridging domain : v1

IP address	MAC address	Flags	Logical Interface	Active source
10.1.1.254	10:10:10:00:00:01	S,K	irb.1	
2001:db8::a:1:1:fffe	20:20:20:00:00:01	S,K, RTS	irb.1	
10.1.1.2	c8:e7:f0:4b:d1:00	DR,K, RTS	vtep.32769	
192.168.22.22				
2001:db8::a:1:1:2	c8:e7:f0:4b:d1:00	DR,K, RTS	vtep.32769	
192.168.22.22				
fe80::cae7:f000:14b:d100	c8:e7:f0:4b:d1:00	DR,K,RT	vtep.32769	
192.168.22.22				
10.1.1.1	dc:38:e1:e0:30:c0	S,K	irb.1	
2001:db8::a:1:1:1	dc:38:e1:e0:30:c0	S,K	irb.1	
fe80::de38:e100:1e0:30c0	dc:38:e1:e0:30:c0	S,K	irb.1	

MAC IP flags (S - Static, D - Dynamic, L - Local , R - Remote, Lp - Local Proxy,
Rp - Remote Proxy, K - Kernel, RT - Dest Route, (N)AD - (Not) Advt to remote,
RE - Re-ARP/ND, RO - Router, OV - Override, Ur - Unresolved,
RTS - Dest Route Skipped, RGW - Remote Gateway)

Routing instance : default-switch

Bridging domain : v2

IP address	MAC address	Flags	Logical Interface	Active source
10.1.2.254	10:10:10:00:00:02	S,K, RTS	irb.2	
2001:db8::a:1:2:fffe	20:20:20:00:00:02	S,K, RTS	irb.2	
10.1.2.2	c8:e7:f0:4b:d1:00	DR,K, RTS	vtep.32769	
192.168.22.22				
2001:db8::a:1:2:2	c8:e7:f0:4b:d1:00	DR,K, RTS	vtep.32769	
192.168.22.22				
fe80::cae7:f000:24b:d100	c8:e7:f0:4b:d1:00	DR,K,RT	vtep.32769	
192.168.22.22				
10.1.2.1	dc:38:e1:e0:30:c0	S,K	irb.2	

2001:db8::a:1:2:1	dc:38:e1:e0:30:c0	S,K	irb.2
fe80::de38:e100:2e0:30c0	dc:38:e1:e0:30:c0	S,K	irb.2

With the extensive option, the show ethernet-switching mac-ip-table command includes the value dest_route_skipped in the MAC IP flags row. The following command shows the MAC+IP table information for MAC address c8:e7:f0:4b:d1:00:

```
user@device> show ethernet-switching mac-ip-table extensive c8:e7:f0:4b:d1:00
```

```
IP address: 10.1.1.2
```

```
MAC address: c8:e7:f0:4b:d1:00
```

```
Routing instance: default-switch
```

```
Bridging domain: v1
```

```
Logical interface: vtep.32769
```

IP address	MAC address	Flags	Logical Interface	Active source
---------------	----------------	-------	----------------------	------------------

```
192.168.22.22
```

```
MAC-IP local mask: 0x0000000000000000
```

```
MAC-IP remote mask: 0x0000000000000000
```

```
MAC-IP remote-proxy mask: 0x0000000000000000
```

```
MAC-IP RPD flags: 0x08000081
```

```
MAC-IP flags: remote, kernel, dest_route_skipped
```

```
IP address: 2001:db8::a:1:1:2
```

```
MAC address: c8:e7:f0:4b:d1:00
```

```
Routing instance: default-switch
```

```
Bridging domain: v1
```

```
Logical interface: vtep.32769
```

IP address	MAC address	Flags	Logical Interface	Active source
---------------	----------------	-------	----------------------	------------------

```
192.168.22.22
```

```
MAC-IP local mask: 0x0000000000000000
```

```
MAC-IP remote mask: 0x0000000000000000
```

```
MAC-IP remote-proxy mask: 0x0000000000000000
```

```
MAC-IP RPD flags: 0x08000101
```

```
MAC-IP flags: remote, kernel, dest_route_skipped
```

```
IP address: fe80::cae7:f000:14b:d100
```

```
MAC address: c8:e7:f0:4b:d1:00
```

```
Routing instance: default-switch
```

Bridging domain: v1

Logical interface: vtep.32769

IP address	MAC address	Flags	Logical Interface	Active source
---------------	----------------	-------	----------------------	------------------

192.168.22.22

MAC-IP local mask: 0x0000000000000000

MAC-IP remote mask: 0x0000000000000000

MAC-IP remote-proxy mask: 0x0000000000000000

MAC-IP RPD flags: 0x08000101

MAC-IP flags: remote, kernel, dest_route

IP address: 10.1.2.2

MAC address: c8:e7:f0:4b:d1:00

Routing instance: default-switch

Bridging domain: v2

Logical interface: vtep.32769

IP address	MAC address	Flags	Logical Interface	Active source
---------------	----------------	-------	----------------------	------------------

192.168.22.22

MAC-IP local mask: 0x0000000000000000

MAC-IP remote mask: 0x0000000000000000

MAC-IP remote-proxy mask: 0x0000000000000000

MAC-IP RPD flags: 0x08000081

MAC-IP flags: remote, kernel, **dest_route_skipped**

IP address: 2001:db8::a:1:2:2

MAC address: c8:e7:f0:4b:d1:00

Routing instance: default-switch

Bridging domain: v2

Logical interface: vtep.32769

IP address	MAC address	Flags	Logical Interface	Active source
---------------	----------------	-------	----------------------	------------------

192.168.22.22

MAC-IP local mask: 0x0000000000000000

MAC-IP remote mask: 0x0000000000000000

MAC-IP remote-proxy mask: 0x0000000000000000

MAC-IP RPD flags: 0x08000101

MAC-IP flags: remote, kernel, **dest_route_skipped**

IP address: fe80::cae7:f000:24b:d100

```

MAC address: c8:e7:f0:4b:d1:00
Routing instance: default-switch
Bridging domain: v2
Logical interface: vtep.32769

```

IP address	MAC address	Flags	Logical Interface	Active source
192.168.22.22				

```

MAC-IP local mask: 0x0000000000000000
MAC-IP remote mask: 0x0000000000000000
MAC-IP remote-proxy mask: 0x0000000000000000
MAC-IP RPD flags: 0x08000101
MAC-IP flags: remote,kernel,dest_route

```

show route forwarding-table

The ["show route forwarding-table" on page 1989](#) CLI command shows when the device uses a local Type 5 route over a learned Type 2 route for a destination. In this case, when you include the `extensive` option, the output includes the keywords `VxLAN Local` in the `Flags` field. This flag means the route is a Type 5 route.

For example:

```

user@device> show route forwarding-table destination 10.1.1.2 table vrf1 extensive
Routing table: vrf1.inet [Index 8]
Internet:

Destination: 10.1.1.2/32
Route type: user
Route reference: 0                      Route interface-index: 0
Multicast RPF nh index: 0
P2mpidx: 0
Flags: sent to PFE, prefix load balance , VxLAN Local
Nexthop:
Next-hop type: composite                Index: 1799      Reference: 3
Next-hop type: indirect                  Index: 524291   Reference: 2
Nexthop: 10.0.0.2
Next-hop type: unicast                   Index: 1796     Reference: 3
Next-hop interface: xe-0/0/11.0

```

show route table

The ["show route table" on page 2003](#) CLI command with the extensive option includes the keyword `VxlanLocalRT` in the State output field for a Type 5 route.

Here's an example using the syntax `show route destination-prefix table table-name extensive`. This command shows that the device installed a Type 5 route for the destination IP prefix 10.1.1.2 in the routing table for routing instance vrf1:

```
user@device> show route 10.1.1.2 table vrf1.inet.0 extensive

vrf1.inet.0: 14 destinations, 18 routes (14 active, 0 holddown, 0 hidden)
10.1.1.2/32 (1 entry, 1 announced)
TSI:
KRT in-kernel 10.1.1.2/32 -> {composite(1799)}
    *EVPN   Preference: 170/-101
           Next hop type: Indirect, Next hop index: 0
           Address: 0x6d06abc
           Next-hop reference count: 6
           Next hop type: Router, Next hop index: 1796
           Next hop: 10.0.0.2 via xe-0/0/11.0, selected
           Label element ptr: 0x83f82d8
           Label parent element ptr: 0x0
           Label element references: 1
           Label element child references: 0
           Label element lsp id: 0
           Session Id: 0x0
           Protocol next hop: 192.168.22.22
           Composite next hop: 0x66fc1b0 1799 INH Session ID: 0x0
           VXLAN tunnel rewrite:
             MTU: 0, Flags: 0x0
             Encap table ID: 0, Decap table ID: 8
             Encap VNI: 100, Decap VNI: 100
             Source VTEP: 192.168.11.11, Destination VTEP: 192.168.22.22
             SMAC: dc:38:e1:e0:30:c0, DMAC: c8:e7:f0:4b:d1:00
           Indirect next hop: 0x6df0e84 524291 INH Session ID: 0x0
           State: <Active Int Ext VxlanLocalRT>
           Age: 5:10:00   Metric2: 1
           Validation State: unverified
           Task: vrf1-EVPN-L3-context
           Announcement bits (1): 2-KRT
           AS path: I
```

```

Communities: target:100:200 encapsulation:vxlan(0x8) router-mac:c8:e7:f0:4b:d1:00
Thread: junos-main
Composite next hops: 1
  Protocol next hop: 192.168.22.22 Metric: 1
  Composite next hop: 0x66fc1b0 1799 INH Session ID: 0x0
  VXLAN tunnel rewrite:
    MTU: 0, Flags: 0x0
    Encap table ID: 0, Decap table ID: 8
    Encap VNI: 100, Decap VNI: 100
    Source VTEP: 192.168.11.11, Destination VTEP: 192.168.22.22
    SMAC: dc:38:e1:e0:30:c0, DMAC: c8:e7:f0:4b:d1:00
  Indirect next hop: 0x6df0e84 524291 INH Session ID: 0x0
  Indirect path forwarding next hops: 1
    Next hop type: Router
    Next hop: 10.0.0.2 via xe-0/0/11.0
    Session Id: 0x0
    192.168.22.22/32 Originating RIB: inet.3
    Metric: 1 Node path count: 1
    Forwarding nexthops: 1
      Next hop type: Router
      Next hop: 10.0.0.2 via xe-0/0/11.0
      Session Id: 0x0

```

RELATED DOCUMENTATION

| [ip-prefix-routes](#) | [1606](#)

Ingress Virtual Machine Traffic Optimization

IN THIS SECTION

- [Benefits of Ingress Virtual Machine Traffic Optimization](#) | [42](#)

When virtual machines and hosts are moved within a data center or from one data center to another, network traffic can become inefficient when the traffic is not routed to the optimal gateway. This can happen when the host is relocated. The arp table does not always get flushed and data flow to the host

is sent to the configured gateway even when there is a more optimal gateway. The traffic is “tromboned” and routed unnecessarily to the configured gateway.

Starting in Junos OS Release 18.4R1, Junos supports ingress Virtual machine traffic optimization (VMTO). When ingress VMTO feature is enabled, the remote IP host routes are stored in L3 Virtual Routing and Forwarding (VRF) table and the device routes inbound traffic directly back to host that has been relocated.

[Figure 3 on page 38](#) illustrates tromboned traffic without ingress VMTO and optimized traffic with ingress VMTO enabled. Without ingress VMTO, Spine 1 and 2 from DC1 and DC2 all advertise the remote IP host route 10.0.0.1 when the origin route is from DC2. The ingress traffic can be directed to either Spine 1 and 2 in DC1. It would then be routed to Spine 1 and 2 in DC2 where route 10.0.0.1 has been moved. This causes the tromboning effect. With ingress VMTO, we can achieve optimal forwarding

path by configuring a policy for IP host route (10.0.0.1) to only be advertised by Spine 1 and 2 from DC2, and not from DC1 when the IP host is moved to DC2.

Figure 3: Traffic with and without Ingress VMTO

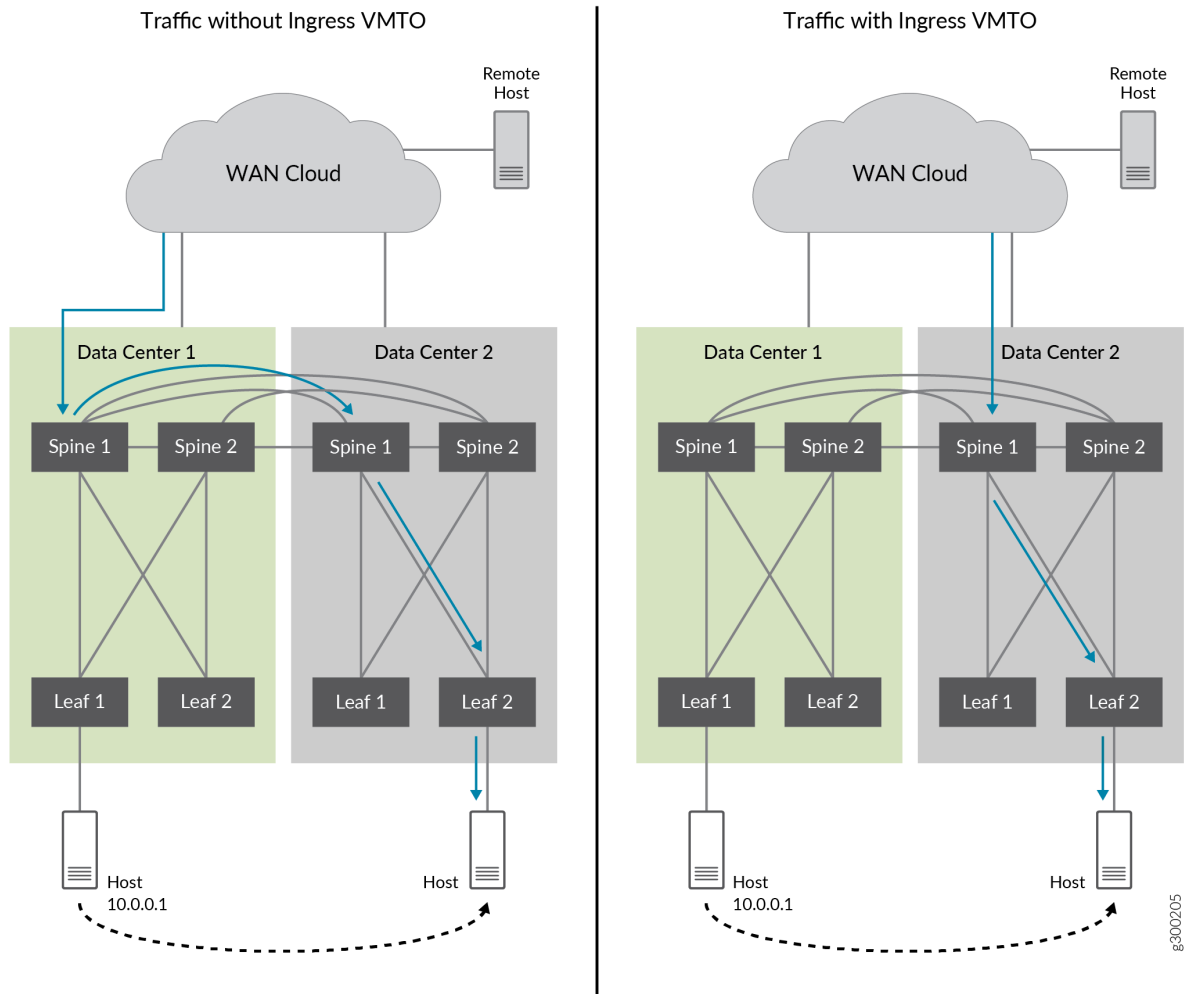
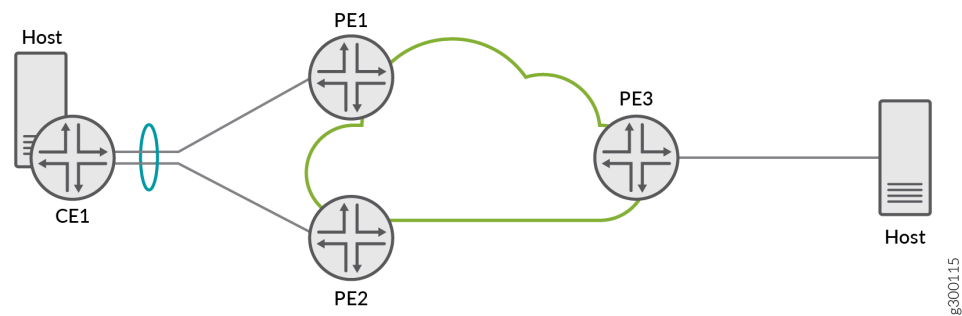


Figure 4 on page 39 illustrates an EVPN network with different elements. PE1 shares an Ethernet Segment with PE2. PE3 is on a separate segment. When PE1 learns of new IP host routes from CE1, PE1 adds the route into the VRF table since it is a locally learned route. If PE2 learns the route from remote peer PE1 (and not from CE1), PE2 will also add the route into VRF table as if the route is also locally learned since PE1 and PE2 are in the same Ethernet Segment. Table 2 on page 39 summarizes the activity taken in the VRF table when a PE device learns of new IP host routes from remote PE devices under EVPN-VXLAN and there are no routing policies configured on the device. Table 3 on page 40 summarizes the activity taken in the VRF table when a PE device learns of new IP host routes from remote PE devices under EVPN-MPLS and there are no routing policies configured on the device.

NOTE: You can also configure policies to selectively add the routes that you want to the VRF table.

Figure 4: EVPN with multihomed devices



NOTE: The IP host routes that PE1 and PE2 learned from each other are described as “From remote device that is connected to a shared Ethernet Segment” and the IP host routes that PE3 learned from PE1 or PE2 are described as “From a remote device that is not connected to a shared Ethernet Segment.”

Table 2: Activity in the VRF table for EVPN-VXLAN

Ingress VMTO Configuration Status	From a remote device that is connected to a shared Ethernet Segment	From a remote device that is not connected to a shared Ethernet Segment
Prior to Junos OS Release 18.4R1.	The IP host route is created with the IRB interface as the next hop and the route is added to the VRF instance table.	The IP host route is not added to the VRF instance table.
Ingress VMTO not configured	The IP host route is created with the IRB interface as the next hop and the route is added to the VRF instance table.	The IP host route is not added to the VRF instance table.

Table 2: Activity in the VRF table for EVPN-VXLAN (Continued)

Ingress VMTO Configuration Status	From a remote device that is connected to a shared Ethernet Segment	From a remote device that is not connected to a shared Ethernet Segment
Ingress VMTO configured	The IP host route is created with the IRB interface as the next hop and the route is added to the VRF instance table.	The IP host route is created with the IRB interface as the next hop and the route is added to the VRF instance table.

Table 3: Activity in the VRF table for EVPN-MPLS

Ingress VMTO Configuration Status	From a remote device with a locally shared Ethernet Segment	From a remote device without a locally shared Ethernet Segment
Prior to Junos OS Release 18.4 R1 with chained composite next hop configured.	The IP host route is created with the composite next hop. The route is not advertise to its peer.	The IP host route is created with the composite next hop. The route is not advertise to its peer.
Prior to Junos OS Release 18.4R1 without chained composite next hop.	The IP host route is not added to the VRF instance table.	The IP host route is not added to the VRF instance table.
Ingress VMTO not configured	The IP host route is created with the IRB interface as the next hop and the route is added to the VRF instance table.	The IP host route is not added to the VRF instance table.
Ingress VMTO configured	The IP host route is created with the IRB interface as the next hop and the route is added to the VRF instance table.	The IP host route is created with the IRB interface as the next hop and the route is added to the VRF instance table.
Ingress VMTO configured with composite next hop	The IP host route is created with the IRB interface as the next hop and the route is added to the VRF instance table.	The IP host route is created with the composite next hop and the route is added to the VRF instance table.

To enable ingress VMTO, configure `remote-ip-host-routes` in the `[edit routing-instances routing-instance-name protocols evpn]` hierarchy level. You can specify the remote IP host routes to be added to the VRF table

through policies by including an import policy to under `remote-ip-host` routes to filter out the unwanted routes.

NOTE: Juniper recommends that you only enable ingress VMTO on the spine devices in a centrally-routed bridging (CRB) underlay in an EVPN network. This allows devices to advertise learned routes to other routing protocols and to advertise EVPN type 5 routes across different data centers.

To address tomboning in [Figure 3 on page 38](#), you would define the communities for Data Center 1 and Data Center 2 and configure a policy in the spine devices to only import the routes learned from members in its own community. Before the move, the spine devices in Data Center 1 advertise the IP host route to the host. After the move, the spine devices in Data Center 2 will advertise the IP host route to the host. As a result, the next-hop table on the remote host will have the updated route to Data Center 2 after the move.

The following output shows the sample policy and sample configuration with an import policy configured with `remote-ip-host`.

```
user@router1# show policy-options
policy-statement vmto-DC1-import {
  term in-DC1 {
    from community [DC1_devices];
    then accept;
  }
  term not-in-DC1 {
    then reject;
  }
}
```

```
user@router1# show routing-instances
blue {
  instance-type virtual-switch;
  route-distinguisher 10.255.0.3:100;
  vrf-import vmto-DC1-import;
  vrf-target target:100:100;
  protocols {
    evpn {
      remote-ip-host-routes {
        import vmto-DC1-import;
        mpls-use-cnh;
      }
    }
  }
}
```

```

    }
    extended-vlan-list 100;
    default-gateway do-not-advertise;
  }
}
.

```

Benefits of Ingress Virtual Machine Traffic Optimization

Ingress VMTO provides greater network efficiency and optimizes ingress traffic and can eliminate the trombone effect between VLANs.

Tracing EVPN Traffic and Operations

To configure the EVPN routing instance to trace a variety of different parameters related to EVPN operation:

1. Specify the name of one or more EVPN trace files using the `file` option for the `traceoptions` statement at the `[edit routing-instances routing-instance-name protocols evpn]` hierarchy level:

```

traceoptions {
  file filename <files number> <size size> <world-readable | no-world-readable>;
}

```

The `file` option includes the following sub-options:

- `filename`—Specify the name of the file to receive the output of the tracing operation. Enclose the name within quotation marks. All files are placed in the directory `/var/log`.
- `files number`—(Optional) Maximum number of trace files. When a trace file named **trace-file** reaches its maximum `size`, it is renamed **trace-file.0**, then **trace-file.1**, and so on, until the specified maximum `number` of trace files specified is reached. Then the oldest trace file is overwritten.
- `size size`—(Optional) Maximum size of each trace file. When a trace file named `trace-file` reaches its maximum size, it is renamed `trace-file.0`, then `trace-file.1`, and so on, until the maximum number of trace files is reached. Then the oldest trace file is overwritten.
- `world-readable | no-world-readable`—(Optional) Enable unrestricted file access or restrict file access to the user who created the file.

2. Specify the flag option for the traceoptions statement:

```
traceoptions {
    flag flag <flag-modifier> <disable>;
}
```

The flag option allows you to specify the scope of the trace by including one of the following sub-options:

- all—All EVPN tracing options
- error—Error conditions
- general—General events
- mac-database—MAC route database in the EVPN routing instance
- nlri—EVPN advertisements received or sent by means of the BGP
- normal—Normal events
- oam—OAM messages
- policy—Policy processing
- route—Routing information
- state—State transitions
- task—Routing protocol task processing
- timer—Routing protocol timer processing
- topology—EVPN topology changes caused by reconfiguration or advertisements received from other PE routers using BGP

You can also specify one of the following modifiers for any of the traceoptions flags:

- detail—Provide detailed trace information.
- disable—Disable this trace flag.
- receive—Trace received packets.
- send—Trace sent packets.

RELATED DOCUMENTATION

[Configuring EVPN Routing Instances | 10](#)

[traceoptions \(Protocols EVPN\) | 1656](#)

Migrating From BGP VPLS to EVPN Overview

IN THIS SECTION

- [Technology Overview and Benefits | 44](#)
- [BGP VPLS to EVPN Migration | 45](#)
- [EVPN Migration Configuration | 47](#)
- [Reverting to VPLS | 49](#)
- [BGP VPLS to EVPN Migration and Other Features | 49](#)

For service providers using both BGP VPLS and EVPN networks, there is a need to interconnect these networks. Prior to Junos OS Release 18.1, a logical tunnel interfaces on the interconnection point of the VPLS and EVPN routing instances was used for this purpose. In this case, the PE devices in each network are unaware of the PE devices in the other technology network. Starting in Junos OS Release 18.1, a solution is introduced for enabling staged migration from BGP VPLS toward EVPN on a site-by-site basis for every VPN routing instance. In this solution, the PE devices running EVPN and VPLS for the same VPN routing instance and single-homed segments can coexist. The solution supports single-active redundancy of multi-homed networks and multi-homed devices for EVPN PEs. With single-active redundancy, the participant VPN instances may span across both EVPN PEs and VPLS PEs as long as single-active redundancy is employed by EVPN PEs.

NOTE: For migration from BGP VPLS, you should expect some traffic loss when family EVPN is enabled to carry EVPN NLRIs. Routing-instance migration to EVPN should see minimal loss.

The following sections describe the migration from BGP VPLS to EVPN:

Technology Overview and Benefits

VPLS is an Ethernet-based point-to-multipoint Layer 2 VPN. This technology allows you to connect geographically dispersed data center LANs to each other across an MPLS backbone while maintaining

Layer 2 connectivity. The high availability features defined in VPLS standards (such as LER dual homing) and topology autodiscovery features using BGP signaling make VPLS scalable and easy to deploy. Because VPLS uses MPLS as its core, it provides low latency variation and statistically bound low convergence times within the MPLS network.

EVPN, on the other hand, is a combined Layer 2 and Layer 3 VPN solution that is more scalable, resilient, and efficient than current technologies. It provides several benefits including greater network efficiency, reliability, scalability, virtual machine (VM) mobility, and policy control for service providers and enterprises.

Although VPLS is a widely deployed Layer 2 VPN technology, service provider networks migrate to EVPN on account of the scaling benefits and ease of deployment. Some of the benefits of EVPN include:

- Control plane traffic is distributed with BGP and the broadcast and multicast traffic is sent using a shared multicast tree or with ingress replication.
- Control plane learning is used for MAC and IP addresses instead of data plane learning. MAC address learning requires the flooding of unknown unicast and ARP frames; whereas, IP address learning does not require any flooding.
- Route reflector is used to reduce a full mesh of BGP sessions among PE devices to a single BGP session between a PE and the route reflector.
- Autodiscovery with BGP is used to discover PE devices participating in a given VPN, PE devices participating in a given redundancy group, tunnel encapsulation types, multicast tunnel type, multicast members, etc.
- All-Active multihoming is used. This allows a given CE device to have multiple links to multiple PE devices, and traffic traversing to-and-from that CE fully utilizes all of these links (Ethernet segment).
- When a link between a CE device and a PE device fails, the PE devices for that EVPN instance (EVI) are notified of the failure with the withdrawal of a single EVPN route. This allows those PE devices to remove the withdrawing PE device as a next hop for every MAC address associated with the failed link (mass withdrawal).

BGP VPLS to EVPN Migration

Some service providers want to preserve their investments in VPLS. This leads to the need to connect the old VPLS networks to new networks that run EVPN. For this purpose, logical tunnel interfaces on the interconnection point of the VPLS and EVPN routing-instances was used. However, all the other PE devices either belonged to the VPLS network or the EVPN network and were unaware of the other technology.

Starting in Junos OS Release 18.1, EVPN can be introduced into an existing BGP VPLS network in a staged manner, with minimal impact to VPLS services. On a BGP VPLS PE device, some customers could

be moved to EVPN, while other customers continue to use VPLS pseudowires. Other PE devices could be entirely VPLS and switching customers on other PEs to EVPN.

The seamless migration from BGP VPLS to EVPN solution supports the following functionality:

- Allow for staged migration toward EVPN on a site-by-site basis per VPN instance. For instance, new EVPN sites to be provisioned on EVPN PE devices.
- Allow for the coexistence of PE devices running both EVPN and VPLS for the same VPN instance and single-homed segments.

In the BGP VPLS to EVPN migration, the PE device where some customers have been migrated to EVPN while other customers are being served using VPLS, is called a super PE. As super PE devices discover other super PE devices within a routing instance, they use the EVPN forwarding to communicate with other super PE devices and VPLS pseudowires to PE devices running VPLS. The PE device with no EVPN awareness, and running only VPLS for all the customers, is called a VPLS PE.

The CE device connected to a super PE can reach both the CE devices connected to EVPN-only PE devices and the CE devices connected to the VPLS-only PE device. The CE devices connected to EVPN-only PE devices cannot reach the CE devices that are connected to VPLS-only PE devices.

Because the migration from BGP VPLS to EVPN is supported at a per routing instance basis, and if the routing instance is serving multiple customers on a PE device, all are migrated together. EVPN is responsible for setting up data forwarding between the PE devices upgraded to EVPN, while VPLS continues to setup data forwarding to PE devices that run VPLS.

NOTE: The following features are not supported with the BGP VPLS to EVPN migration:

- Migration from FEC129 VPLS to EVPN.
- Migration of VPLS virtual switch to EVPN virtual switch.
- Migration of VPLS routing instance to EVPN virtual switch.
- Migration of VPLS routing instance or PBB-VPLS to PBB-EVPN.
- Seamless migration from EVPN back to VPLS.
- Enhancing EVPN to support the set of tools or statements and commands that VPLS supports.
- Spanning all-active across EVPN and VPLS PE devices does not work, as all-active multihoming feature is not supported on VPLS.
- Connecting EVPN-only PE devices with VPLS-only PE devices through super PE devices.

- IPv6, logical systems, multi-chassis support, and SNMP, as they are currently not supported on EVPN.

EVPN Migration Configuration

To perform the BGP VPLS to EVPN migration, do the following:

1. On the backup Routing Engine, load Junos OS Release 18.1R1.
2. Perform ISSU to acquire primary role. Ensure that the VPLS ISSU does not have any impact on the VPLS forwarding.
3. Identify routing instances (customers) that need to be migrated to EVPN.
4. Enable family EVPN signaling in BGP by adding family evpn and signaling at the [edit protocols bgp group-session] hierarchy level.

NOTE: Adding family evpn to the BGP protocol will cause the CE IFL to be deleted and recreated, so you should expect some traffic loss after this step.

```
protocols {
  bgp {
    group session {
      family evpn {
        signaling;
      }
    }
  }
}
```

5. Configure the ESI interface using a preference value that reflects the configured VPLS preference.

NOTE: Creating the ESI interface will cause the CE IFL to be deleted and recreated, so you should expect some traffic loss after this step.

```
[edit interfaces interface-name]
esi {
```

```

00:01:02:03:04:05:06:00:00:01;
    single-active;
    df-election-type {
        preference {
            value match vpls preference;
        }
    }
}

```

6. Enable EVPN in a single routing instance.

- Change routing instance type from `vpls` to `evpn`, at the `[edit routing-instances routing-instance-name]` hierarchy level in your existing BGP VPLS configuratoin.

```

[edit routing-instances routing-instance-name]
instance-type evpn;

```

7. Include the `evpn` and `vpls` statements at the `[edit routing-instances routing-instance-name protocols]` hierarchy level in order to support EVPN and VPLS commands.

```

[edit routing-instances routing-instance-name]
protocols {
    evpn
    vpls
}

```

8. Once all nodes in the VPLS network have been migrated to EVPN, you can optionally decommission the VPLS protocol. This will allow EVPN to setup its single-homing and multi-homing state cleanly, but may result in traffic loss. To decommission the VPLS protocol, delete the `protocols vpls` statement under `[edit routing-instances routing-instance-name]` hierarchy.

```

[edit routing-instances routing-instance-name]
user@host# delete protocols vpls

```

After the configuration for the EVPN migration is committed, the routing protocol process and the Layer 2 address learning process start building the EVPN state to reflect interfaces, bridge domains, peers and routes. The locally learnt MAC addresses are synchronized by the Layer 2 address learning process in the `instance.vpls.0` to the routing protocol process. When a local MAC ages out in the `instance.vpls.0`, the routing protocol process is informed by the Layer 2 address learning process.

When an EVPN peer is learnt, the routing protocol process sends a new message to the Layer 2 address learning process to remove the peer's label-switched interface or virtual tunnel logical interface from the VE mesh group and disables MAC-learning on it. The EVPN IM next-hop is then added to the VE mesh group. The EVPN behavior in the routing protocol process of learning MAC addresses over BGP and informing Layer 2 address learning process of the MPLS next hop is maintained.

The VPLS statements and commands continue to apply to the VPLS pseudowires between the PE devices and the MAC addresses learnt over them. The EVPN statements and commands apply to PE devices running EVPN.

Reverting to VPLS

If the EVPN migration runs into issues, you can revert back to VPLS until the issue is understood. The routing instance is reverted from a super PE to a VPLS PE in a non-catastrophic manner by enabling the following configuration:

```
[edit routing-instances routing-instance-name]
user@host# set instance-type vpls
user@host# delete protocols evpn
```

On reverting the EVPN migration to VPLS, the following happens:

1. The EVPN state information is deleted.
2. There is a trigger for withdrawal of EVPN control plane routes.
3. The routing protocol process sends a new message to the Layer 2 address learning process with the label-switched interface or the virtual tunnel logical interface for the routing instance and peer.
4. The label-switched or virtual tunnel interface adds the new message to the flood group and MAC learning is enabled.
5. The egress IM next hop is deleted by the routing protocols process, prompting the Layer 2 address learning process to remove it from the flood group.
6. Remote MAC addresses are learnt again over the label-switched interface or virtual tunnel logical interface.

BGP VPLS to EVPN Migration and Other Features

[Table 4 on page 50](#) describes the functionality of some of the related features, such as multihoming and integrated routing and bridging (IRB) with the BGP VPLS to EVPN migration.

Table 4: EVPN Migration and Other Features Support

Feature	Supported Functionality in EVPN Migration
MAC move	<p>MAC moves are supported between VPLS-only PE and super PE devices.</p> <p>When a MAC address moves from a VPLS-only PE to a super PE, it is learnt over BGP, and the routing protocol process informs the Layer 2 address learning process of the EVPN next hop to be updated in the foo.vpls.0 routing table.</p> <p>When a MAC address moves from a super PE to a VPLS-only PE, it is learnt in the Packet Forwarding Engine on the label-switched interface or virtual tunnel interface. The Layer 3 address learning process updates it to VPLS or the label-switched interface next hop.</p> <p>When the type 2 route is withdrawn by EVPN BGP, the MAC address is not deleted from the forwarding table, so there is no loss of data.</p> <p>The forwarding MAC table is shared by VPLS and EVPN. Some attributes, such as mac-table-size and mac-table-aging-time could be configured under both EVPN and VPLS. When there is a conflict, the values under EVPN take precedence.</p>
IRB	<p>No changes needed in IRB.</p> <p>On super PE, EVPN populates the /32 host routes learnt over MAC+IP type 2 routes from EVPN peers in a Layer 3 virtual routing and forwarding, while VPLS IRB forwarding using subnet routes work on sites still running VPLS.</p>
Hierarchical VPLS	<p>In a H-VPLS network with hub and spoke PE devices, when the hub PE is migrated to EVPN, local MAC addresses learnt over the access label-switched or virtual tunnel interface need to be advertised into BGP, so that the other EVPN-only PE devices or super PE devices can reach them.</p> <p>Take the following into consideration when migrating a H-VPLS network to EVPN:</p> <ul style="list-style-type: none"> • Hubs typically have local-switching enabled as inter-spoke traffic is forwarded through the hub. If spoke(s) alone is migrated to EVPN and spokes have Layer 3 or MPLS reachability to each other, the label-switched or virtual tunnel interface to the hub and EVPN next hop (remote spoke) is present in the VE floodgroup and this results in two copies of broadcast, unknown unicast, and multicast (BUM) traffic received by the remote spoke. An option to avoid this is to migrate the hub(s) to EVPN too. • EVPN is not aware of hierarchy. All peers are considered core-facing. Once hubs and spokes are migrated to EVPN, split horizon prevents the BUM traffic from being forwarded to other core-facing PE devices.

Table 4: EVPN Migration and Other Features Support *(Continued)*

Feature	Supported Functionality in EVPN Migration
ESI configuration	ESI is configured at the physical interface or port level.

RELATED DOCUMENTATION

| [EVPN Overview](#) | 874

Configuring Route Targets

IN THIS CHAPTER

- [Auto-derived Route targets | 52](#)
- [Example: Configuring VNI Route Targets Automatically | 54](#)
- [Example: Configuring VNI Route Targets Manually | 58](#)
- [Example: Configuring VNI Route Targets Automatically with Manual Override | 60](#)

Auto-derived Route targets

IN THIS SECTION

- [Benefits of Auto-Derived Route Targets | 54](#)

Route targets are used to identify the different routes that are imported and exported into the VRF tables. When you enable the auto-derived route targets option, Junos derives the route targets based on the EVPN encapsulation for EVPN route type 2 (MAC/IP Advertisement Route) and EVPN route type 3 (Inclusive Multicast Ethernet Tag route). The route targets for remaining EVPN route types are not auto-derived. For EVPN-MPLS, the route target is automatically derived from the VLAN ID (VID). For EVPN-VXLAN, the route target is automatically derived from the VXLAN network identifier (VNI). For PBB-EVPN, the route target is derived from instance service identifier (ISID). The auto-derived route targets have higher precedence over manually configured route targets in vrf-targets, vrf-export policies, and vrf-import policies.

As defined in RFC 8365, the auto-derived route target field includes the following fields:

- Global Administrator—A 2-octet field containing an AS number assigned by Internet Assigned Numbers Authority (IANA).
- Local Administrator—A 4-Octet field that includes the following:

- A single bit field with a value of zero indicating that the RT is auto-derived.
- Type—A 3-bit field identifying the service.
- D-ID—A 4-bit field identifying the domain ID.
- Service ID—A 3-octet field set to the VNI, VSID, I-SID, or VID.

NOTE: Auto-derived route targets are not supported for inter-AS routing.

To enable auto-derived route targets, include the `auto` statement at the [edit routing-instances routing-instance-name vrf-target]. Auto-derived route targets are supported on virtual switch and EVPN routing instances.

The following is a sample configuration for auto-derived route targets for an EVPN and virtual switch routing instance:

```
routing-instances {
  VS-1 {
    instance-type virtual-switch;
    interface ae0.110;
    interface ae1.120;
    interface ae2.130;
    route-distinguisher 100.100.100.2:101;
    vrf-target {
      target:100:101;
      auto;
    }
    protocols {
      evpn {
        extended-vlan-list [ 110 120 130 ];
      }
    }
    bridge-domains {
      bd-110 {
        vlan-id 110;
      }
      bd-120 {
        vlan-id 120;
      }
      bd-130 {
        vlan-id 130;
      }
    }
  }
}
```



```

    }
  }
}
EVPN-1 {
  instance-type evpn;
  vlan-id 10;
  interface ae0.0;
  interface ae1.0;
  interface ae2.0;
  route-distinguisher 100.100.100.2:1;
  vrf-target {
    target:100:1
    auto;
  }
  protocols {
    evpn;
  }
}

```

Benefits of Auto-Derived Route Targets

Auto-derived route targets simplify the configuration of VLAN services for EVPN, especially in VLAN-aware bundle services where you can have multiple VLANs, multiple bridge domains and the VLANs for a given service are not present on all PE devices. Without auto-derived target option enabled, EVPN type 2 and type 3 routes are imported into the EVIs on all receiving PE devices and the routes subsequently dropped for non-existing VLANs (bridge-domains). To minimize the number of routes that are distributed, different auto-derived route targets can be used within each bridge-domain. Together with constrained route distribution as described in RFC 4684, you can limit the distribution of bridge-domain specific EVPN route types (Type 2 and Type 3) to only the interested PE devices.

Example: Configuring VNI Route Targets Automatically

IN THIS SECTION

- [Requirements | 55](#)
- [Overview | 55](#)

This example shows how to automatically derive route targets for multiple VNIs in an EVPN-VXLAN topology.

Requirements

This example uses the following hardware and software components:

- A QFX Series switch.
- Junos OS version 15.1X53-D30

Overview

The `vrf-target` statement can be used to configure specific route targets for each VNI under `vni-options`. You can configure the `vrf-target` statement with the `auto` option to automatically derive route targets for each VNI.

Configuration

IN THIS SECTION

- [Configuring VNI Route Target Automatic Derivation | 55](#)
- [Results | 56](#)

To configure the `vrf-target` statement with the `auto` option, perform these tasks:

Configuring VNI Route Target Automatic Derivation

Step-by-Step Procedure

To configure the automatic derivation of VNI route targets:

1. At the `[switch-options]` hierarchy level, configure the `vtep-source-interface`, and `route-distinguisher` statements. Then configure the `vrf-import` statement with a policy that will be configured in a later step. Next, configure the `vrf-target` statement with a target and the `auto` option. The route target

configured under `vrf-target` will be used by type 1 EVPN routes. Type 2 and type 3 EVPN routes will use the auto-derived per-VNI route target for export and import.

```
[edit switch-options]
user@switch# set vtep-source-interface lo0.0
user@switch# set route-distinguisher 192.0.2.11:1
user@switch# set vrf-import import-policy
user@switch# set vrf-target target:1111:11
user@switch# set vrf-target auto
```

2. At the `[evpn]` hierarchy level, configure the `encapsulation` and `extended-vni-list` statements. For the purposes of this example, the `extended-vni-list` statement will be configured with `all`, to apply automatic route targets to all VNIs.

```
[edit protocols evpn]
user@switch# set encapsulation vxlan
user@switch# set extended-vni-list all
```

3. Configure two policies at the `[policy-statement]` hierarchy level. The first policy will be named `comglobal` and the next policy will be named `import-policy`. These policies function as an import filter that accepts routes with the auto-derived route target.

```
[edit policy-options]
user@switch# set community comglobal members target:1111:11
```

```
[edit policy-options policy-statement import-policy]
user@switch# set term 1 from community comglobal
user@switch# set term 1 then accept
user@switch# set term 100 then reject
```

Results

Use the `show` command to verify the results of your configuration.

```
user@switch> show configuration switch-options
vtep-source-interface lo0.0;
route-distinguisher 192.0.2.11:1;
```

```
vrf-import imp;  
vrf-target {  
    target:1111:11;  
    auto;  
}
```

```
user@switch> show configuration protocols evpn  
encapsulation vxlan;  
extended-vni-list all;
```

```
user@switch> show configuration policy-options community comglobal  
members target:1111:11;
```

```
user@switch> show configuration policy-options policy-statement import-policy  
term 1{  
    from community [ comglobal ];  
    then accept;  
}  
term 100 {  
    then reject;  
}
```

RELATED DOCUMENTATION

[vrf-target | 1670](#)

[Example: Configuring VNI Route Targets Manually | 58](#)

[Example: Configuring VNI Route Targets Automatically with Manual Override | 60](#)

Example: Configuring VNI Route Targets Manually

IN THIS SECTION

- [Requirements | 58](#)
- [Overview | 58](#)
- [Configuration | 58](#)

This example shows how to manually set route targets for multiple VNIs in an EVPN-VXLAN topology.

Requirements

This example uses the following hardware and software components:

- A QFX Series switch.
- Junos OS version 15.1X53-D30

Overview

The `vrf-target` statement can be used to configure specific route targets for each VNI under `vni-options`. You can configure the `vrf-target` statement to automatically derive route targets for each VNI or you can configure route targets manually. This example explains how to configure route targets manually.

Configuration

IN THIS SECTION

- [Configuring VNI Route Targets Manually | 59](#)
- [Results | 59](#)

To manually configure VNI route targets, perform these tasks:

Configuring VNI Route Targets Manually

Step-by-Step Procedure

To manually configure VNI route targets:

1. At the [switch-options] hierarchy level, configure the vtep-source-interface, and route-distiguisher statements. Next, configure the vrf-target statement with a target. All EVPN routes for all VLANs will use the vrf-target address configured in this step.

```
[edit switch-options]
user@switch# set vtep-source-interface lo0.0
user@switch# set route-distinguisher 192.0.2.11:1
user@switch# set vrf-import import-policy
user@switch# set vrf-target target:1111:11
```

2. At the [evpn] hierarchy level, configure the encapsulation and extended-vni-list statements. For the purposes of this example, the extended-vni-list statement will be configured with all, to apply automatic route targets to all VNIs.

```
[edit protocols evpn]
user@switch# set encapsulation vxlan
user@switch# set extended-vni-list all
```

Results

After following the steps above, use the show command to verify the results of your configuration.

```
user@switch> show configuration switch-options
vtep-source-interface lo0.0;
route-distinguisher 192.0.2.11:1;
vrf-target {
    target:1111:11;
}
```

RELATED DOCUMENTATION

| [vrf-target](#) | 1670

[Example: Configuring VNI Route Targets Automatically | 54](#)

[Example: Configuring VNI Route Targets Automatically with Manual Override | 60](#)

Example: Configuring VNI Route Targets Automatically with Manual Override

IN THIS SECTION

- [Requirements | 60](#)
- [Overview | 60](#)
- [Configuration | 61](#)

This example shows how to automatically set route targets for multiple VNIs, and manually override the route target for a single VNI in an EVPN-VXLAN topology.

Requirements

This example uses the following hardware and software components:

- A QFX Series switch.
- Junos OS version 15.1X53-D30

Overview

The `vrf-target` statement can be used to configure specific route targets for each VNI under `vni-options`. You can configure the `vrf-target` statement to automatically derive route targets for each VNI or you can configure route targets manually. It's also possible to automatically derive route targets for VNIs, then manually override the route target for one or more VNIs. This example explains how to manually override the automatically assigned route targets for a specific VNI.

Configuration

IN THIS SECTION

- [Configuring Automatic VNI Route Targets with Manual Override | 61](#)
- [Results | 62](#)

To manually override an automatically configured VNI route targets, perform these tasks:

Configuring Automatic VNI Route Targets with Manual Override

Step-by-Step Procedure

To configure automatic VNI route targets with manual override:

1. At the `[switch-options]` hierarchy level, configure the `vtep-source-interface`, and `route-distinguisher` statements. Then configure the `vrf-import` statement with a policy that will be configured in a later step. Next, configure the `vrf-target` statement with a target and the `auto` option. The route target configured under `vrf-target` will be used by type 1 EVPN routes and all type 2 and 3 EVPN routes for all VLANs except the ones that do not match the VNI under `vni-options` in the next step.

```
[edit switch-options]
user@switch# set vtep-source-interface lo0.0
user@switch# set route-distinguisher 192.0.2.11:1
user@switch# set vrf-import import-policy
user@switch# set vrf-target target:1111:11
user@switch# set vrf-target auto
```

2. The `[evpn]` hierarchy level is where you can override the automatic assignment of VNI route targets. Configure the `vni-options` statement for VNI 100 with an export target of 1234:11. This route target will be used by type 2 and 3 EVPN routes for all VLANs that match VNI 100. Next, configure the `encapsulation` and `extended-vni-list` statements. For the purposes of this example, the `extended-vni-list` statement will be configured with only 2 VNIs.

```
[edit protocols evpn]
user@switch# vni-options vni 100 vrf-target export target:1234:11
```



```

user@switch# set encapsulation vxlan
user@switch# set extended-vni-list 100 101

```

3. Configure three policies at the [policy-statement] hierarchy level. The first policy will be named `comglobal`, the next policy will be named `com1234`, and the final policy will be named `import-policy`. These policies function as an import filter that accepts routes using the auto-derived route target and the manual override route target.

```

[edit policy-options policy-statement comglobal]
user@switch# set members target:1111:11

```

```

[edit policy-options policy-statement com1234]
user@switch# set members target:1234:11

```

```

[edit policy-options policy-statement import-policy]
user@switch# set term 1 from community comglobal com1234
user@switch# set term 1 then accept
user@switch# set term 100 then reject

```

Results

After following the steps above, use the `show` command to verify the results of your configuration.

```

user@switch> show configuration switch-options
vtep-source-interface lo0.0;
route-distinguisher 192.0.2.11:1;
vrf-import imp;
vrf-target {
    target:1111:11;
    auto;
}

```

```

user@switch> show configuration protocols evpn
vni-options {
    vni 100 {
        vrf-target export target:1234:11;
    }
}

```

```
    }  
}  
encapsulation vxlan;  
extended-vni-list [ 100 101 ];
```

```
user@switch> show configuration policy-options community comglobal  
members target:1111:11;
```

```
user@switch> show configuration policy-options community com1234  
members target:1234:11;
```

```
user@switch> show configuration policy-options policy-statement import-policy  
term 1{  
    from community [ com1234 comglobal ];  
    then accept;  
}  
term 100 {  
    then reject;  
}
```

RELATED DOCUMENTATION

[vrf-target | 1670](#)

[Example: Configuring VNI Route Targets Automatically | 54](#)

[Example: Configuring VNI Route Targets Manually | 58](#)

Routing Policies for EVPN

IN THIS CHAPTER

- Routing policies for EVPN | 64
- Example: Using Policy Filters to Filter EVPN Routes | 70

Routing policies for EVPN

SUMMARY

Create routing policies to control the EVPN routing information that will be imported and exported to the different routing tables.

Routing policies allow you to control the routing information that are imported and exported to the routing and forwarding tables. Starting in Junos OS 19.4R1, Junos has expanded routing policy support to include the creation and application of policy filters specific to EVPN routes.

Policies can be applied at the routing-instance level or at a BGP level. When you apply policies at the BGP level, they effect all EVPN routing instances. When applied at the routing-instance level, they effect the specified EVPN routing instance only. To apply the policy at the BGP level, include the `vpn-apply-export` statement at the `[edit protocols bgp]` hierarchy level and import or export the policy. To apply the policy at the routing-instance level, use `vrf-export` or `vrf-import` statement to apply the policy for that particular routing instance.

Policies are composed of match conditions, actions, and terms. For more information on policies, see [Policy Framework Overview](#).

[Table 5 on page 65](#) lists the match conditions supported for use in filtering EVPN routes.

Table 5: List of Match Conditions for Filtering EVPN Routes

Match Condition	Description																																			
<code>community [names]</code>	<p>BGP EVPN routes can have a set of EVPN extended communities carried in the BGP update message path attribute, and as such, you can use these extended communities for filtering BGP EVPN routes. The EVPN specific information available in extended communities includes, for example, encapsulation type, MAC-mobility information, EVPN split-horizon label information, EVPN ESI split-horizon label, ESI mode, E-tree leaf label, and more.</p> <p>Use the following syntax to specify BGP EVPN extended communities:</p> <ul style="list-style-type: none"><code>set policy-options community <i>name</i> members type:val1:val2</code> <p>All values (including type) are in decimal; type is 2 octets, with the higher-order octet defining the type of extended community, and the low-order octet defining the community sub-type. <i>val1</i> and <i>val2</i> can be specified as [2 + 4] octets, or as [4 + 2] octets.</p> <p>The extended communities most commonly used with BGP EVPN routes are provided here.</p> <table><tr><th>High-order</th><th>Low-order (Sub-type)</th><th>Type (Hex)</th><th>Type (Dec)</th><th>Name</th></tr><tr><td>0x03</td><td>0x0c</td><td>0x030c</td><td>780</td><td>BGP Encapsulation</td></tr><tr><td>0x03</td><td>0x0d</td><td>0x030d</td><td>781</td><td>Default Gateway</td></tr><tr><td>0x06</td><td>0x00</td><td>0x0600</td><td>1536</td><td>EVPN MAC Mobility</td></tr><tr><td>0x06</td><td>0x01</td><td>0x0601</td><td>1537</td><td>EVPN ESI Label</td></tr><tr><td>0x06</td><td>0x02</td><td>0x0602</td><td>1538</td><td>EVPN ES-Import Route Target</td></tr><tr><td>0x06</td><td>0x04</td><td>0x0604</td><td>1540</td><td>EVPN Layer 2 Attributes</td></tr></table>	High-order	Low-order (Sub-type)	Type (Hex)	Type (Dec)	Name	0x03	0x0c	0x030c	780	BGP Encapsulation	0x03	0x0d	0x030d	781	Default Gateway	0x06	0x00	0x0600	1536	EVPN MAC Mobility	0x06	0x01	0x0601	1537	EVPN ESI Label	0x06	0x02	0x0602	1538	EVPN ES-Import Route Target	0x06	0x04	0x0604	1540	EVPN Layer 2 Attributes
High-order	Low-order (Sub-type)	Type (Hex)	Type (Dec)	Name																																
0x03	0x0c	0x030c	780	BGP Encapsulation																																
0x03	0x0d	0x030d	781	Default Gateway																																
0x06	0x00	0x0600	1536	EVPN MAC Mobility																																
0x06	0x01	0x0601	1537	EVPN ESI Label																																
0x06	0x02	0x0602	1538	EVPN ES-Import Route Target																																
0x06	0x04	0x0604	1540	EVPN Layer 2 Attributes																																

Table 5: List of Match Conditions for Filtering EVPN Routes *(Continued)*

Match Condition	Description					
	<table><tr><td>0x06</td><td>0x05</td><td>0x0605</td><td>1541</td><td>EVPN E-Tree</td></tr></table>	0x06	0x05	0x0605	1541	EVPN E-Tree
0x06	0x05	0x0605	1541	EVPN E-Tree		
	<p>For full list of Extended Communities please refer to Border Gateway Protocol (BGP) Extended Communities .</p>					
evpn-esi	<p>You can filter BGP EVPN routes on the basis of Ethernet Segment Identifiers (ESIs) information for routes types 1, 2, 4, 7, and 8, which are the only types to include the ESI attribute in their prefix. (ESI values are encoded as 10-byte integers and are used to identify a multihomed segment.) Note that the evpn-esi matching statement is valid only together with “family evpn” matching statement.</p>					
evpn-etag	<p>You can filter BGP EVPN routes on the basis of EVPN Ethernet Tag information, which is part of the prefix of the EVPN route. This matching statement is valid only together with family evpn match statement.</p>					
evpn-mac-route	<p>Filtering BGP EVPN Type 2 routes based on if it has any IP address.</p> <p>EVPN Type 2 MAC/IP Advertisement routes can have IP address in the prefix along with MAC address. The IP address carried in the MAC-IP Advertisement route can be either IPv4 or IPv6 address. It is possible to filter out Type 2 routes based on MAC address only, MAC+IPv4 address, or MAC+IPv6 address. To do so requires the following CLI statement be set:</p> <ul style="list-style-type: none">from evpn-mac-route [mac-ipv4 mac-ipv6 mac-only] <p>Note that this match statement is valid only together with the family evpn match statement.</p>					
local-preference	<p>Set the local preference (LOCAL_PREF) attribute. The preference value can be a number in the range from 0 through 4,294,967,295.</p>					
mac-filter-list	<p>(BGP only) Named MAC filter list. EVPN Type 2 routes have MAC address as part of the prefix, which you can use to create a list of MAC addresses.</p>					

Table 5: List of Match Conditions for Filtering EVPN Routes *(Continued)*

Match Condition	Description
metric	Metric corresponds to the MED, and metric2 corresponds to the IGP metric if the BGP next hop loops through another router. You can specify up to four metric values, metric, metric2, metric3, and metric4.
next-hop (<i>address / discard / next-table table-name / peer-address / reject / self</i>)	<p>Requires IBGP or EBGp confederations (third-party next hop must be advertised).</p> <ul style="list-style-type: none"> • <i>discard</i>—The next-hop address is replaced by a discard next hop. • <i>next-table</i>—The routing device performs a forwarding lookup in the specified table. • <i>self</i>—The next-hop address is replaced by one of the local routing device's addresses, as determined by the advertising protocol, typically the local IP address used for the BGP adjacency. • <i>specify peer-address</i>—The next-hop address is replaced by the peer's IP address (import only), typically an advertising routing device or another directly connected routing device.
nlri-route-type	<p>For EVPN, NLRI route types range from 1 to 8 (the first octet of the route prefix in the BGP update message is the EVPN route type).</p> <p>Multiple route types can be specified in a single policy.</p>
origin	<p>Set the BGP path origin attribute to one of the following values:</p> <ul style="list-style-type: none"> • <i>egp</i>—Path information originated in another AS. • <i>igp</i>—Path information originated within the local AS. • <i>incomplete</i>—Path information learned by some other means.

Table 5: List of Match Conditions for Filtering EVPN Routes (Continued)

Match Condition	Description
<code>prefix-list-filter</code> <i>prefix-list-name match-type</i>	<p>Both <code>prefix-list</code> and <code>prefix-list-filter</code> match conditions are supported. <code>prefix-list</code> is similar to <code>prefix-list-filter</code>, with the exception that a <code>match-type</code> can be specified only with <code>prefix-list-filter</code>. You can specify prefix length qualifiers for the list of prefixes in the prefix list.</p> <p>When used with EVPN NRLI route Types 2 and 5, the following are supported:</p> <ul style="list-style-type: none"> from <code>prefix-list-filter</code> [<code>exact</code> <code>longer</code> <code>orlonger</code>]
<code>route-distinguisher</code>	<p>Value of the route-distinguisher (RD).</p> <p>Filtering BGP EVPN routes based on RD is supported. The RD information is carried in the prefix of the EVPN route.</p>
<code>route-filter</code> <code>route-filter-list</code>	<p>Named route filter or route filter list. You can specify prefix length qualifiers for the list of routes in the route filter list.</p> <p>When used with EVPN NRLI route types 2 and 5, the following are supported:</p> <ul style="list-style-type: none"> from <code>route-filter</code> [<code>address-mask</code> <code>exact</code> <code>longer</code> <code>orlonger</code> <code>prefix-length-range</code> <code>through</code> <code>upto</code>]

When using policy filters to filter EVPN routes, in Junos OS Release 19.4R1 and later, the following policy actions are supported (that is, they can be specified as the **then** qualifier in the policy).

[Table 6 on page 68](#) lists actions that can be used when filtering EVPN routes.

Table 6: List of Actions for Filtering EVPN Routes

Action	Description
<code>accept</code>	Accept a route.
<code>apply-groups</code> <i>group-name</i>	Apply a configuration group to a policy. If you specify more than one group name, the first group listed takes priority over the next, and so on.

Table 6: List of Actions for Filtering EVPN Routes *(Continued)*

Action	Description
apply-groups-except <i>group-name</i>	Disable inheritance of a configuration group. This action is useful when you use the apply-group statement in a policy but also want to override the values inherited from the configuration group for a specific parameter.
as-path-prepend	<p>Appends one or more AS numbers at the beginning of the AS path. If you are specifying more than one AS number, include the numbers in quotation marks.</p> <p>The AS numbers are added after the local AS number has been added to the path. This action adds AS numbers to AS sequences only, not to AS sets. If the existing AS path begins with a confederation sequence or set, the appended AS numbers are placed within a confederation sequence. Otherwise, the appended AS numbers are placed with a non-confederation sequence.</p>
default-action	Accept or Reject any action log protocol by overriding them. This is a non-terminating policy action.
next	Skip to next policy or term.
preference	Sets the BGP local preference attribute for the route. The preference can be a number from 0 through 4,294,967,295), with lower numbers being more preferred. Selected routes are installed into the forwarding table.
priority	Set the priority for route installation: high, low, or medium. High priority routes are updated first in the in the RIB (routing table) and the FIB (forwarding table), before medium and low priority routes. Routes are placed in different priority queues according to the priority.
reject	Rejects the route and does not propagate it. After a route is rejected, no other terms in the routing policy and no other routing policies are evaluated.
tag (add subtract) tag2 (add subtract) <i>number</i>	Change the tag value by the specified amount.

RELATED DOCUMENTATION

[Policy Framework Overview](#)[Example: Using Policy Filters to Filter EVPN Routes | 70](#)[export \(Routing Options\) | 1572](#)[import | 1588](#)[vpn-apply-export | 1664](#)[vrf-export | 1667](#)[vrf-import | 1668](#)

Example: Using Policy Filters to Filter EVPN Routes

IN THIS SECTION

- [Requirements | 70](#)
- [Overview | 71](#)
- [Base Configuration | 71](#)

In Junos, routing policies can be used to control Border Gateway Protocol (BGP) route advertisements and to filter routes using different address families. But, although Ethernet VPN (EVPN) uses BGP to exchange MAC-IP addresses between different PE routers, differences such as the EVPN route prefix format and extended community information that is encoded in the BGP update message, mean that special match conditions are needed to be able to filter EVPN routes.

The examples in this topic show the various router configurations available in Junos for filtering EVPN routes.

Requirements

EVPN route filtering is supported on MX, VMX, EX, ACX, and QFX devices running Junos Release 19.4R1 or later. It is available at the routing-instance level of the hierarchy (where it is configured with **vrf-export** or **vrf-import** policy), and at the protocols bgp level (in which case you also need to configure vpn-apply-export for the policy to take effect).

Overview

IN THIS SECTION

- [Topology | 71](#)

You can use policy filters to filter EVPN routes, for example to specify particular extended community attributes. Routes are filtered according to the match conditions you specify in the **from** qualifier of the policy. Supported match criteria for EVPN routes include EVPN NLRI type, BGP path attributes, route distinguishers, EVPN Ethernet Tag, Ethernet Segment Identifier (ESI), and MAC addresses in EVPN Type 2 routes.

The following route filters are also supported: local-preference, as-path, community, next-hop, metric, and origin.

Actions are taken according to the criteria you specify in the **then** qualifier specified in the policy.

See "[Routing policies for EVPN](#)" on page 64 for a complete list and description of supported match conditions and actions.

Topology

The following network scenarios show the configuration used for setting up various EVPN match conditions.

Base Configuration

IN THIS SECTION

- [Verification | 94](#)

CLI Quick Configuration

For EVPN routes, a policy can be applied at the routing-instance level of the hierarchy, or at the protocols bgp level. The configuration for both is shown below. At the routing-instance level, the policy is applied as an vrf-export or vrf-import policy. When an *export* policy is applied at the BGP group level, you must configure vpn-apply-export for the policy to work properly.

Case 1 shows the mandatory use of the statement `vpn-apply-export` when a policy is applied at the BGP level of the hierarchy.

To use the example, you need to navigate to various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To quickly configure the examples, copy the list of commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

Case 1: Applying the policy at the protocol BGP level of the hierarchy.

```
set protocols bgp group evpn-sessions type internal
set protocols bgp group evpn-sessions local-address 10.255.255.4
set protocols bgp group evpn-sessions import bgp-evpn-exp
set protocols bgp group evpn-sessions family evpn signaling
set protocols bgp group evpn-sessions neighbor 10.255.255.1
set protocols bgp group evpn-sessions neighbor 10.255.255.6
set protocols bgp group evpn-sessions neighbor 10.255.255.8
set protocols bgp group evpn-sessions vpn-apply-export
set policy-options policy-statement bgp-evpn-exp term 1 from family evpn
set policy-options policy-statement bgp-evpn-exp term 1 from nlri-route-type 2
set policy-options policy-statement bgp-evpn-exp term 1 from nlri-route-type 3
set policy-options policy-statement bgp-evpn-exp term 1 then community add COM5
set policy-options policy-statement bgp-evpn-exp term 1 then as-path-prepend 999
```

Case 2 shows the mandatory use of the statements `vrf-export` and `vrf-import` when match conditions are being applied at the routing instances level of the hierarchy.

EVPN uses 8 different route types to extend Layer 2 connectivity. The EVPN NLRI route type is defined in the first octet of the route prefix field in the BGP update message.

NOTE: In Junos, the following EVPN route types, **Type 1 AD per ESI**, **Type 4 ES**, **Type 7 IGMP join**, and **Type 8 IGMP leave**, routes are not specific to a given routing-instance. Instead, they are automatically added to the default routing-instance table when exported. As a result no routing-instance **vrf-export** or **vrf-import** policies are applied to these route types. If you want to apply an export policy to these routes, you need to do it at the BGP export level of the hierarchy. The same is true for importing Type 1 per ESI, Type 4, Type 7, and Type 8 routes (they are automatically imported into the default-routing instance table). So, to apply an import policy to these route types, you need to do so at the BGP import level of the hierarchy rather than at the routing-instance level.

Case 2: Applying the policy at the routing-instance level of the hierarchy.

```

set routing-instances evpa protocols evpn
set routing-instances evpa instance-type evpn
set routing-instances evpa vlan-id none
set routing-instances evpa routing-interface irb.600
set routing-instances evpa interface ge-0/0/1.600
set routing-instances evpa route-distinguisher 2:3
set routing-instances evpa vrf-export vrf-exp-pol
set routing-instances evpa vrf-target target:1:1
set policy-options policy-statement vrf-exp-pol term 1 from family evpn
set policy-options policy-statement vrf-exp-pol term 1 from nlri-route-type 1
set policy-options policy-statement vrf-exp-pol term 1 then community add COM11
set policy-options policy-statement vrf-exp-pol term 1 then accept

```

Filtering BGP EVPN routes based on EVPN NLRI type

CLI Quick Configuration

A complete list of set commands used in the example are presented first, followed by the same commands in step-by-step format, as well as instructions for confirming your configuration. Verification commands that you can use to see relevant output from a properly configured system are shown at the end of this topic.

Filtering BGP EVPN routes based on EVPN NLRI type

```

set policy-options policy-statement bgp-evpn-exp term 1 from family evpn
set policy-options policy-statement bgp-evpn-exp term 1 from nlri-route-type 2
set policy-options policy-statement bgp-evpn-exp term 1 from nlri-route-type 3
set policy-options policy-statement bgp-evpn-exp term 1 then community add COM5
set policy-options policy-statement bgp-evpn-exp term 1 then as-path-prepend 999
set protocols bgp group evpn-session type internal
set protocols bgp group evpn-session local address 10.255.255.4
set protocols bgp group evpn-session family evpn signaling
set protocols bgp group evpn-session export bgp-evpn-exp
set protocols bgp group evpn-session neighbor 10.255.255.1
set protocols bgp group evpn-session neighbor 10.255.255.6
set protocols bgp group evpn-session neighbor 10.255.255.8
set protocols bgp group evpn-session vpn-apply-export

```

Step-by-Step Procedure

To set up the filtering of BGP EVPN routes based on BGP path attributes:

1. Configure the BGP path attributes you want to filter on (enclose multiple types in brackets and separate with a space) and the action to take on the matching routes.

```
[edit policy-options policy-statement bgp-evpn-exp]
user@PE1# set term 1 from family evpn
user@PE1# set term 1 from nlri-route-type [2 3]
user@PE1# set term 1 then community add COM5
user@PE1# set term 1 then as-path-prepend 999
```

2. Configure the BGP group protocol session.

```
[edit protocols bgp group evpn-session ]
user@PE1# set type internal
user@PE1# set local address 10.255.255.4
user@PE1# set family evpn signaling
user@PE1# set import bgp-evpn-exp
user@PE1# set neighbor 10.255.255.1
user@PE1# set neighbor 10.255.255.6
user@PE1# set neighbor 10.255.255.8
user@PE1# set vpn-apply-export
```

Results

To see your configuration results, from configuration mode at the top of the CLI hierarchy, confirm your configuration by entering the `show policy-options policy-statement bgp-evpn-exp`, , and `show protocols bgp group evpn-sessions` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show policy-options policy-statement bgp-evpn-exp
term 1 {
  from {
    family evpn;
    nlri-route-type [ 2 3 ];
  }
  then {
    community add COM5;
```

```

        as-path-prepend 999;
    }
}

```

```

user@PE1# show protocols bgp group evpn-sessions
group evpn-sessions {
    type internal;
    local-address 10.255.255.4;
    export bgp-evpn-exp;
    family evpn {
        signaling;
    }
    local-address 10.255.255.1;
    local-address 10.255.255.6;
    local-address 10.255.255.8;
    vpn-apply-export;
}

```

Filtering BGP EVPN routes based on route distinguisher

CLI Quick Configuration

Route distinguisher (RD) information is encoded in the EVPN route prefix. This example shows how to filter EVPN routes on the basis of the route distinguisher.

A complete list of set commands used in the example are presented first, followed by the same commands in step-by-step format, as well as instructions for confirming your configuration. Verification commands that you can use to see relevant output from a properly configured system are shown at the end of this topic.

Filtering BGP EVPN routes based on route distinguisher

```

set policy-options policy-statement bgp-evpn-exp term 1 from family evpn
set policy-options policy-statement bgp-evpn-exp term 1 from route-distinguisher 100:200
set policy-options policy-statement bgp-evpn-exp term 1 then community add COM5
set policy-options policy-statement bgp-evpn-exp term 1 then as-path-prepend 999
set protocols bgp group evpn-session type internal
set protocols bgp group evpn-session local address 10.255.255.4
set protocols bgp group evpn-session family evpn signaling
set protocols bgp group evpn-session export bgp-evpn-exp
set protocols bgp group evpn-session neighbor 10.255.255.1
set protocols bgp group evpn-session neighbor 10.255.255.6

```

```
set protocols bgp group evpn-session neighbor 10.255.255.8
set protocols bgp group evpn-session vpn-apply-export
```

Step-by-Step Procedure

To set up the filtering of BGP EVPN routes based on route distinguisher:

1. Configure the route distinguisher you want to filter on and the action to take on the matching routes.

```
[edit policy-options policy-statement bgp-evpn-exp]
user@PE1# set term 1 from family evpn
user@PE1# set term 1 from route-distinguisher 100:200
user@PE1# set term 1 then community add COM5
user@PE1# set term 1 then as-path-prepend 999
```

2. Configure the BGP group protocol session.

```
[edit protocols bgp group evpn-session ]
user@PE1# set type internal
user@PE1# set local address 10.255.255.4
user@PE1# set family evpn signaling
user@PE1# set export bgp-evpn-exp
user@PE1# set neighbor 10.255.255.1
user@PE1# set neighbor 10.255.255.6
user@PE1# set neighbor 10.255.255.8
user@PE1# set vpn-apply-export
```

Results

To see your configuration results, from configuration mode at the top of the CLI hierarchy, confirm your configuration by entering the `show policy-options policy-statement bgp-evpn-exp`, `show policy-options route-distinguisher RD1`, and `show protocols bgp group evpn-sessions` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show policy-options policy-statement bgp-evpn-exp
term 1 {
  from {
    family evpn;
    route-distinguisher 100:200;
```

```

    }
    then {
        community add COM5;
        as-path-prepend 999;
    }
}

```

```

user@PE1# show protocols bgp group evpn-sessions
group evpn-sessions {
    type internal;
    local-address 10.255.255.4;
    import bgp-evpn-exp;
    family evpn {
        signaling;
        local-address 10.255.255.1;
        local-address 10.255.255.6;
        local-address 10.255.255.8;
        vpn-apply-export;
    }
}

```

Filtering BGP EVPN routes based on EVPN Ethernet Tags

CLI Quick Configuration

EVPN Ethernet Tag information (or vlan-id information) is carried in the prefix of the EVPN route. This example shows how to filter EVPN routes based on the Ethernet Tag carried in the prefix of the route. Note that you must include the `family evpn` qualifier when configuring this filtering option.

A complete list of set commands used in the example are presented first, followed by the same commands in step-by-step format, as well as instructions for confirming your configuration. Verification commands that you can use to see relevant output from a properly configured system are shown at the end of this topic.

Filtering BGP EVPN routes based on EVPN Ethernet Tags

```

set policy-options policy-statement bgp-evpn-exp term 1 from family evpn
set policy-options policy-statement bgp-evpn-exp term 1 from evpn-tag [ 10 12 13 ]
set policy-options policy-statement bgp-evpn-exp term 1 then community add COM5
set policy-options policy-statement bgp-evpn-exp term 1 then as-path-prepend 999
set protocols bgp group evpn-session type internal
set protocols bgp group evpn-session local address 10.255.255.4

```



```

set protocols bgp group evpn-session family evpn signaling
set protocols bgp group evpn-session export bgp-evpn-exp
set protocols bgp group evpn-session neighbor 10.255.255.1
set protocols bgp group evpn-session neighbor 10.255.255.6
set protocols bgp group evpn-session neighbor 10.255.255.8
set protocols bgp group evpn-session vpn-apply-export

```

Step-by-Step Procedure

To set up the filtering of BGP EVPN routes based on the EVPN Ethernet Tag:

1. Configure the EVPN Ethernet Tag you want to filter on and the action to take on the matching routes.

```

[edit policy-options policy-statement bgp-evpn-exp]
user@PE1# set term 1 from family evpn
user@PE1# set term 1 from evpn-tag [ 10 12 13 ]
user@PE1# set term 1 then community add COM5
user@PE1# set term 1 then as-path-prepend 999

```

2. Configure the BGP group protocol session.

```

[edit protocols bgp group evpn-session ]
user@PE1# set type internal
user@PE1# set local address 10.255.255.4
user@PE1# set family evpn signaling
user@PE1# set import bgp-evpn-exp
user@PE1# set neighbor 10.255.255.1
user@PE1# set neighbor 10.255.255.6
user@PE1# set neighbor 10.255.255.8
user@PE1# set vpn-apply-export

```

Results

To see your configuration results, from configuration mode at the top of the CLI hierarchy, confirm your configuration by entering the `show policy-options policy-statement bgp-evpn-exp`, and `show protocols bgp group`

evpn-sessions commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show policy-options policy-statement bgp-evpn-exp
term 1 {
    from {
        family evpn;
        evpn-tag [ 10 12 13 ];
    }
    then {
        community add COM5;
        as-path-prepend 999;
    }
}
```

```
user@PE1# show protocols bgp group evpn-sessions
group evpn-sessions {
    type internal;
    local-address 10.255.255.4;
    import bgp-evpn-exp;
    family evpn {
        signaling;
    }
    local-address 10.255.255.1;
    local-address 10.255.255.6;
    local-address 10.255.255.8;
    vpn-apply-export;
}
```

Filtering BGP EVPN routes based on ESI

CLI Quick Configuration

You can use Ethernet Segment Identifier (ESI) based policy filters for Type 1, Type 2, Type 4, Type 7, and Type 8 routes, which are the only types to contain ESI information in the prefix.

A complete list of set commands used in the example are presented first, followed by the same commands in step-by-step format, as well as instructions for confirming your configuration. Verification commands that you can use to see relevant output from a properly configured system are shown at the end of this topic.

Filtering BGP EVPN routes based on ESI

```

set policy-options policy-statement bgp-evpn-exp term 1 from family evpn
set policy-options policy-statement bgp-evpn-exp term 1 from evpn-esi
00:11:22:33:44:55:66:77:88:99
set policy-options policy-statement bgp-evpn-exp term 1 then community add COM1
set protocols bgp group evpn-session type internal
set protocols bgp group evpn-session local address 10.255.255.8
set protocols bgp group evpn-session family evpn signaling
set protocols bgp group evpn-session export bgp-evpn-exp
set protocols bgp group evpn-session vpn-apply-export
set protocols bgp group evpn-session neighbor 10.255.255.1
set protocols bgp group evpn-session neighbor 10.255.255.4
set protocols bgp group evpn-session neighbor 10.255.255.6

```

Step-by-Step Procedure

To set up the filtering of BGP EVPN routes based on the ESI:

1. Configure the EVPN ESI you want to filter on and the action to take on the matching routes.

```

[edit policy-options policy-statement bgp-evpn-exp]
user@PE1# set term 1 from family evpn
user@PE1# set term 1 from evpn-esi 00:11:22:33:44:55:66:77:88:99
user@PE1# set term 1 then community add COM1

```

2. Configure the BGP group protocol session.

```

[edit protocols bgp group evpn-session ]
user@PE1# set type internal
user@PE1# set local address 10.255.255.8
user@PE1# set family evpn signaling
user@PE1# set export bgp-evpn-exp
user@PE1# set vpn-apply-export
user@PE1# set neighbor 10.255.255.1
user@PE1# set neighbor 10.255.255.4
user@PE1# set neighbor 10.255.255.6

```

Results

To see your configuration results, from configuration mode at the top of the CLI hierarchy, confirm your configuration by entering the `show policy-options policy-statement bgp-evpn-exp`, and `show protocols bgp group evpn-sessions` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show policy-options policy-statement bgp-evpn-exp
term 1 {
  from {
    family evpn;
    evpn-esi 00:11:22:33:44:55:66:77:88:99;
  }
  then {
    community add COM1;
  }
}
```

```
user@PE1# show protocols bgp group evpn-sessions
group evpn-sessions {
  type internal;
  local-address 10.255.255.8;
  family evpn {
    signaling;
    export bgp-evpn-exp;
    vpn-apply-export;
    local-address 10.255.255.1;
    local-address 10.255.255.4;
    local-address 10.255.255.6;
  }
}
```

Filtering BGP EVPN Type 2 and Type 5 routes based on IP address.

CLI Quick Configuration

You can use IPv4 or IPv6 addresses embedded in the EVPN prefix field to filter EVPN Type 2 and Type 5 routes. The following prefix-list and route-filter qualifiers are also supported:

- from prefix-list

- from prefix-list-filter [exact | longer | orlonger]
- from route-filter [address-mask | exact | longer | orlonger | prefix-length-range | through | upto]
- from route-filter-list

A complete list of set commands used in the example are presented first, followed by the same commands in step-by-step format, as well as instructions for confirming your configuration. Verification commands that you can use to see relevant output from a properly configured system are shown at the end of this topic.

Filtering BGP EVPN Type 2 and Type 5 routes based on the IP address

```
set policy-options prefix-list pp1 10.1.1.10/32
set policy-options prefix-list pp1 10.1.1.11/32
set policy-options policy-statement bgp-evpn-exp term 1 from family evpn
set policy-options policy-statement bgp-evpn-exp term 1 from prefix-list pp1
set policy-options policy-statement bgp-evpn-exp term 1 then community add COM1
set protocols bgp group evpn-session type internal
set protocols bgp group evpn-session local-address 10.255.255.8
set protocols bgp group evpn-session family evpn signaling
set protocols bgp group evpn-session export bgp-evpn-exp
set protocols bgp group evpn-session vpn-apply-export
set protocols bgp group evpn-session neighbor 10.255.255.1
set protocols bgp group evpn-session neighbor 10.255.255.4
set protocols bgp group evpn-session neighbor 10.255.255.6
```

Step-by-Step Procedure

To set up the filtering of BGP EVPN Type 2 and Type 5 routes based on the IP address:

1. Create a prefix list to be used in the policy statement.

```
[ edit policy-options prefix-list pp1]
user@PE1# set 10.1.1.10/32
user@PE1# set 10.1.1.11/32
```

2. Configure the Type 2 and Type 5 IP address you want to filter on and the action to take on the matching routes.

```
[edit policy-options policy-statement bgp-evpn-exp]
user@PE1# set term 1 from family evpn
user@PE1# set term 1 from prefix-list pp1
user@PE1# set term 1 then community add COM1
```

3. Configure the BGP group protocol session.

```
[edit protocols bgp group evpn-session ]
user@PE1# set type internal
user@PE1# set local address 10.255.255.8
user@PE1# set family evpn signaling
user@PE1# set export bgp-evpn-exp
user@PE1# set vpn-apply-export
user@PE1# set neighbor 10.255.255.1
user@PE1# set neighbor 10.255.255.4
user@PE1# set neighbor 10.255.255.6
```

Results

To see your configuration results, from configuration mode at the top of the CLI hierarchy, confirm your configuration by entering the `show policy-options policy-statement bgp-evpn-exp`, and `show protocols bgp group evpn-sessions` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show policy-options prefix-list pp1
10.1.1.10/32;
10.1.1.11/32;
```

```
user@PE1# show policy-options policy-statement bgp-evpn-exp
term 1 {
  from {
    family evpn;
    prefix-list pp1;
  }
  then {
```

```

        community add COM1;
    }
}

```

```

user@PE1# show protocols bgp group evpn-sessions
group evpn-sessions {
    type internal;
    local-address 10.255.255.8;
    family evpn {
        signaling;
    }
    export bgp-evpn-exp;
    vpn-apply-export;
    local-address 10.255.255.1;
    local-address 10.255.255.4;
    local-address 10.255.255.6;
}

```

Filtering BGP EVPN Type 2 routes using MAC address

CLI Quick Configuration

You can use the MAC address in EVPN prefix to filter EVPN Type 2 routes.

A complete list of set commands used in the example are presented first, followed by the same commands in step-by-step format, as well as instructions for confirming your configuration. Verification commands that you can use to see relevant output from a properly configured system are shown at the end of this topic.

Filtering BGP EVPN Type 2 routes using MAC address

```

set policy-options mac-list mfl1 01:87:88:04:50:00
set policy-options mac-list mfl1 02:87:88:04:50:00
set policy-options mac-list mfl1 03:87:88:04:50:00
set policy-options mac-list mfl1 04:87:88:04:50:00
set policy-options mac-list mfl1 05:87:88:04:50:00
set policy-options mac-list mfl1 06:87:88:04:50:00
set policy-options mac-list mfl1 07:87:88:04:50:00
set policy-options mac-list mfl1 08:87:88:04:50:00
set policy-options mac-list mfl1 64:87:88:04:50:00
set policy-options policy-statement bgp-evpn-exp term 1 from family evpn
set policy-options policy-statement bgp-evpn-exp term 1 from mac-filter-list mfl1

```

```

set policy-options policy-statement bgp-evpn-exp term 1 then accept
set protocols bgp group evpn-session type internal
set protocols bgp group evpn-session local address 10.255.255.8
set protocols bgp group evpn-session family evpn signaling
set protocols bgp group evpn-session export bgp-evpn-exp
set protocols bgp group evpn-session vpn-apply-export
set protocols bgp group evpn-session neighbor 10.255.255.1
set protocols bgp group evpn-session neighbor 10.255.255.4
set protocols bgp group evpn-session neighbor 10.255.255.6

```

Step-by-Step Procedure

To set up the filtering of BGP EVPN Type 2 routes using MAC address:

1. Create the list of the MAC addresses you want to filter on (**mfl1** in this example).

```

[edit policy-options mac-list mfl1]
user@PE1# set 01:87:88:04:50:00;
user@PE1# set 02:87:88:04:50:00;
user@PE1# set 03:87:88:04:50:00;
user@PE1# set 04:87:88:04:50:00;
user@PE1# set 05:87:88:04:50:00;
user@PE1# set 06:87:88:04:50:00;
user@PE1# set 07:87:88:04:50:00;
user@PE1# set 08:87:88:04:50:00;

```

2. Apply a list of the MAC addresses you want to filter on, and the action you want to take (**Accept**, in this example).

```

[edit policy-options policy-statement bgp-evpn-exp]
user@PE1# set term 1 from family evpn
user@PE1# set term 1 from mac-filter-list mfl1
user@PE1# set term 1 then accept

```

3. Configure the BGP group protocol session.

```

[edit protocols bgp group evpn-session ]
user@PE1# set type internal
user@PE1# set local address 10.255.255.8
user@PE1# set family evpn signaling

```



```

user@PE1# set export bgp-evpn-exp
user@PE1# set neighbor 10.255.255.1
user@PE1# set neighbor 10.255.255.4
user@PE1# set neighbor 10.255.255.6
user@PE1# set vpn-apply-export

```

Results

To see your configuration results, from configuration mode at the top of the CLI hierarchy, confirm your configuration by entering the `show policy-options mac-list mfl1`, `show policy-options policy-statement bgp-evpn-exp`, and `show protocols bgp group evpn-sessions` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```

user@PE1# show policy-options mac-list mfl1
01:87:88:04:50:00;
02:87:88:04:50:00;
03:87:88:04:50:00;
04:87:88:04:50:00;
05:87:88:04:50:00;
06:87:88:04:50:00;
07:87:88:04:50:00;
08:87:88:04:50:00;

```

```

user@PE1# show policy-options policy-statement bgp-evpn-exp
term 1 {
    from {
        family evpn;
        mac-filter-list mfl1;
    }
    then {
        accept;
    }
}

```

```

user@PE1# show protocols bgp group evpn-sessions
group evpn-sessions {
    type internal;
    local-address 10.255.255.8;
}

```

```

family evpn {
    signaling;
    export bgp-evpn-exp;
    vpn-apply-export;
    local-address 10.255.255.1;
    local-address 10.255.255.4;
    local-address 10.255.255.6;
}

```

Filtering BGP EVPN Type 2 routes that contain (or do not contain) an IP address

CLI Quick Configuration

EVPN Type 2 routes have a MAC address and can additionally have an IP address (IPv4 or IPv6) in the prefix. With BGP EVPN Type 2 filters, you can filter Type 2 routes based according to whether it has only a MAC address, a MAC address and IPv4 address, or a MAC address and IPv6 address (not a specific IP address, but any IP address in the prefix). These options are mutually exclusive.

A complete list of set commands used in the example are presented first, followed by the same commands in step-by-step format, as well as instructions for confirming your configuration. Verification commands that you can use to see relevant output from a properly configured system are shown at the end of this topic.

Filtering BGP EVPN Type 2 routes with MAC address only

```

set policy-options policy-statement bgp-evpn-exp term 1 from family evpn
set policy-options policy-statement bgp-evpn-exp term 1 from evpn-mac-route mac-only
set policy-options policy-statement bgp-evpn-exp term 1 then community add COM1
set protocols bgp group evpn-session type internal
set protocols bgp group evpn-session local address 10.255.255.8
set protocols bgp group evpn-session family evpn signaling
set protocols bgp group evpn-session export bgp-evpn-exp
set protocols bgp group evpn-session vpn-apply-export
set protocols bgp group evpn-session neighbor 10.255.255.1
set protocols bgp group evpn-session neighbor 10.255.255.4
set protocols bgp group evpn-session neighbor 10.255.255.6

```

Step-by-Step Procedure

To set up the filtering of BGP EVPN Type 2 routes with MAC address only:

1. Create a policy and the action you want to take.

```
[edit policy-options policy-statement bgp-evpn-exp]
user@PE1# set term 1 from family evpn
user@PE1# set term 1 from evpn-mac-route mac-only
user@PE1# set term 1 then community add COM1
```

2. Configure the BGP group protocol session (we use export bgp-evpn-exp here to apply the policy).

```
[edit protocols bgp group evpn-session ]
user@PE1# set type internal
user@PE1# set local address 10.255.255.8
user@PE1# set family evpn signaling
user@PE1# set export bgp-evpn-exp
user@PE1# set neighbor 10.255.255.1
user@PE1# set neighbor 10.255.255.4
user@PE1# set neighbor 10.255.255.6
```

Results

To see your configuration results, from configuration mode at the top of the CLI hierarchy, confirm your configuration by entering the, show policy-options policy-statement bgp-evpn-exp, and show protocols bgp group evpn-sessions commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show policy-options policy-statement bgp-evpn-exp
term 1 {
  from {
    family evpn;
    evpn-mac-route mac-only;
  }
  then {
    community add COM1;
  }
}
```

```
user@PE1# show protocols bgp group evpn-sessions
group evpn-sessions {
```

```

type internal;
local-address 10.255.255.8;
family evpn {
    signaling;
    export bgp-evpn-exp;
    vpn-apply-export;
    local-address 10.255.255.1;
    local-address 10.255.255.4;
    local-address 10.255.255.6;
}

```

Filtering BGP EVPN routes according to an EVPN extended community

CLI Quick Configuration

BGP EVPN routes can have a set of extended communities carried in the BGP update message path attribute, and as such, you can use these extended communities for filtering BGP EVPN routes. . The EVPN specific information included in the extended communities includes encapsulation type, MAC-mobility information, EVPN split-horizon label,, ESI mode, E-Tree leaf label, and more.

See [Border Gateway Protocol \(BGP\) Extended Communities](#) for the full list of extended communities.

An extended community is an eight-octet value divided into two main sections, and typically uses a notation of **type:administrator:assigned-number**. However, to specify EVPN extended communities in the Junos configuration for BGP EVPN, instead of using a word to specify the type, all values (including *type*) are in decimal. Type is 2 octet, with the higher-order octet defining the actual type of extended community, and the low-order octet defining the community. The sub-type; val1 and val2 can be specified as [2 + 4] octets, or as [4 + 2] octets.

Typical configuration for extended communities in Junos:

- set policy-options community *name* members type:val1:val2

Specifying an extended community numerically for BGP EVPN configurations in Junos. See [BGP MPLS-Based Ethernet VPN](#) for more information on numerical representations of extended communities.

In the example below, the decimal 780 is used to match the encapsulation extended community (for example, VXLAN). For 780, the value of the high-order octet of the extended type field is 0x03, which indicates that it is transitive. The value of the low-order octet of the extended type field is 0x0c; thus, the first 2 octet value is 0x030c, which is where the decimal 780 comes from. The remaining value fields, where *val1* is 0 and *val2* is 8, are used to identify VXLAN tunnel type.

The full list of tunnel types related to EVPN is defined in RFC 8365, Section 11 (link below), but some pertinent ones are listed here:

- Value 8 = VXLAN Encapsulation
- Value 9 = NVGRE Encapsulation
- Value 10 = MPLS Encapsulation
- Value 11 = MPLS in GRE Encapsulation
- Value 12 = VXLAN GPE Encapsulation

See [RFC 5512, Section 4.5, Reserved field](#) and [RFC 8365, Section 11](#) for details.

- `set policy-options community name members 780:0:8`

A complete list of set commands used in the example are presented first, followed by the same commands in step-by-step format, as well as instructions for confirming your configuration. Verification commands that you can use to see relevant output from a properly configured system are shown at the end of this topic.

Filtering BGP EVPN routes according to the EVPN extended communities

```
set policy-options community COM5 members 780:0:8
set policy-options policy-statement bgp-evpn-exp term 1 from community COM5
set policy-options policy-statement bgp-evpn-exp term 1 then reject
set protocols bgp group evpn-session type internal
set protocols bgp group evpn-session local address 10.255.255.4
set protocols bgp group evpn-session family evpn signaling
set protocols bgp group evpn-session export bgp-evpn-exp
set protocols bgp group evpn-session neighbor 10.255.255.1
set protocols bgp group evpn-session neighbor 10.255.255.6
set protocols bgp group evpn-session neighbor 10.255.255.8
```

Step-by-Step Procedure

To set up the filtering of BGP EVPN routes according to an EVPN extended community:

1. Create a list of the community members you want to filter on, and the action you want to take.

```
[edit policy-options]
user@PE1# set community COM5 members 780:0:8
```

2. Create a list of the community members you want to filter on, and the action you want to take.

```
[edit policy-options policy-statement bgp-evpn-exp]
user@PE1# set term 1 from community COM5
user@PE1# set term 1 then reject
```

3. Configure the BGP group protocol session (we use export bgp-evpn-exp here to apply the policy).

```
[edit protocols bgp group evpn-session ]
user@PE1# set type internal
user@PE1# set local address 10.255.255.4
user@PE1# set family evpn signaling
user@PE1# set export bgp-evpn-exp
user@PE1# set neighbor 10.255.255.1
user@PE1# set neighbor 10.255.255.6
user@PE1# set neighbor 10.255.255.8
```

Results

To see your configuration results, from configuration mode at the top of the CLI hierarchy, confirm your configuration by entering the, show policy-options policy-statement bgp-evpn-exp, and show protocols bgp group evpn-sessions commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show policy-options community COM5 members
members 780:0:8;
```

```
user@PE1# show policy-options policy-statement bgp-evpn-exp
term 1 {
  from {
    community COM5;
  }
  then {
    reject;
```

```
}
}
```

```
user@PE1# show protocols bgp group evpn-sessions
group evpn-sessions {
    type internal;
    local-address 10.255.255.4;
    family evpn {
        signaling;
    }
    export bgp-evpn-exp;
    vpn-apply-export;
    local-address 10.255.255.1;
    local-address 10.255.255.6;
    local-address 10.255.255.8;
}
```

Copying community information from EVPN Type 2 routes into EVPN Type 5 routes

You can use BGP EVPN filtering to include the MAC address (if any) and IPv4 or IPv6 addresses from EVPN type 2 route advertisements received from remote PEs as EVPN Type 5 routes. Likewise, you can copy the community information from EVPN Type 2 routes into EVPN Type 5 route that have been generated from routes in the **vrf.inet table**(specifically, VPN-IPv4 (AFI/SAFI 1/128), VPN-IPv6 (AFI/SAFI 2/128), IPv4 (AFI/SAFI 1/1) and IPv6 (AFI/SAFI 2/1)).

To include any contained MAC address and IPv4 or IPv6 addresses from EVPN Type 2 route advertisements into EVPN Type 5, enable the following command:

- `set routing-instances evpna protocols evpn remote-ip-host-routes no-advertise-community`

You can also control which routing attributes are carried between the IP and EVPN routes. In other words, you can choose which route attributes to include from the import direction when generating IP routes from EVPN Type 5 routes, and for the export direction, also choose which route attributes to include when generating EVPN Type 5 routes from IP routes. These route attributes are, as-path, community, and preference. Note that if you do not explicitly include the community route attribute during import, due to how Junos handles route attributes in the **vrf.inet.0** table, color community information will not be included (and thus this information not available for the nexthop resolution of the affected routes).

To include a given route attribute, use the following commands, and set an import or export action, which can be either allow or skip (here, the import-action is allow):

```
set routing-instances evpna protocols evpn ip-prefix-routes route-attributes as-path import-
action allow
set routing-instances evpna protocols evpn ip-prefix-routes route-attributes preference import-
action allow
set routing-instances evpna protocols evpn ip-prefix-routes route-attributes community import-
action allow
```

Results

To see your configuration results, from configuration mode at the top of the CLI hierarchy, confirm your configuration by entering the, `show policy-options policy-statement bgp-evpn-exp`, and `show protocols bgp group evpn-sessions` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show policy-options policy-statement bgp-evpn-exp
term 1 {
  from {
    family evpn;
    evpn-mac-route mac-only;
  }
  then {
    community add COM1;
  }
}
```

```
user@PE1# show protocols bgp group evpn-sessions
group evpn-sessions {
  type internal;
  local-address 10.255.255.8;
  family evpn {
    signaling;
  }
  export bgp-evpn-exp;
  vpn-apply-export;
  local-address 10.255.255.1;
  local-address 10.255.255.4;
```



```
local-address 10.255.255.6;
}
```

Verification

IN THIS SECTION

- [Verifying the various BGP EVPN filtering | 94](#)

Confirm that the configuration is working properly. For each of the examples given above, run a version of these commands that uses the configuration you want to confirm. The verification example below is based on the example given for filtering BGP EVPN routes based on the EVPN NLRI type.

Verifying the various BGP EVPN filtering

Purpose

Display information about the BGP EVPN routes filtered according to the specified criteria.

Action

From operational mode on the target device, enter following commands:

```
user@device> show evpn instance
user@device> show evpn instance extensive
user@device> show evpn database
user@device> show evpn mac-ip-table
```

From operational mode on PE1, enter following commands:

```
user@PE1> show route table bgp.evpn.0
user@PE1> show route table EVPN-1.evpn.0
user@PE1> show route table default_evpn__.evpn.0
user@PE1> show route advertising-protocol bgp
100.100.100.2 table bgp.evpn.0
user@PE1> show route advertising-protocol bgp
100.100.100.2 table EVPN-1.evpn.0
```

```
user@PE1> show route advertising-protocol bgp
100.100.100.2 table default_evpn__.evpn.0
```

From operational mode on PE2, enter following commands:

```
user@PE2> show route receive-protocol bgp 100.100.100.1
table bgp.evpn.0
user@PE2> show route receive-protocol bgp 100.100.100.1
table EVPN-1.evpn.0
user@PE2> show route receive-protocol bgp 100.100.100.1
table default_evpn__.evpn.0
user@PE2> show route table bgp.evpn.0
user@PE2> show route table EVPN-1.evpn.0
user@PE2> show route table default_evpn__.evpn.0
```

RELATED DOCUMENTATION

[Routing policies for EVPN | 64](#)

[How BGP Communities and Extended Communities Are Evaluated in Routing Policy Match Conditions](#)

[export \(Routing Options\) | 1572](#)

[import | 1588](#)

[vpn-apply-export | 1664](#)

[vrf-export | 1667](#)

[vrf-import | 1668](#)

Configuring EVPN Multihoming

IN THIS CHAPTER

- [EVPN Multihoming Overview | 96](#)
- [Configuring EVPN Active-Standby Multihoming to a Single PE Device | 124](#)
- [Configuring EVPN Active-Standby Multihoming | 127](#)
- [Example: Configuring Basic EVPN Active-Standby Multihoming | 131](#)
- [Example: Configuring EVPN Active-Standby Multihoming | 154](#)
- [Example: Configuring Basic EVPN Active-Active Multihoming | 200](#)
- [Example: Configuring EVPN Active-Active Multihoming | 213](#)
- [Example: Configuring LACP for EVPN Active-Active Multihoming | 279](#)
- [Example: Configuring LACP for EVPN VXLAN Active-Active Multihoming | 301](#)
- [Understanding When to Disable EVPN-VXLAN Core Isolation | 329](#)
- [Configuring Dynamic List Next Hop | 332](#)
- [Example: Configuring an ESI on a Logical Interface With EVPN Multihoming | 336](#)
- [Understanding Automatically Generated ESIs in EVPN Networks | 352](#)

EVPN Multihoming Overview

IN THIS SECTION

- [Introduction to EVPN Multihoming | 97](#)
- [EVPN MPLS Multihoming Features Supported by QFX10000 Switches | 98](#)
- [Understanding EVPN Multihoming Concepts | 99](#)
- [EVPN Multihoming Mode of Operation | 101](#)
- [EVPN Multihoming Implementation | 102](#)
- [Designated Forwarder Election | 115](#)

- [ESIs on Physical, Aggregated Ethernet, and Logical Interfaces | 120](#)
- [Automatically Generated ESIs | 121](#)
- [Convergence in an EVPN Network | 121](#)

Introduction to EVPN Multihoming

An Ethernet VPN (EVPN) comprises of customer edge (CE) devices that are connected to provider edge (PE) devices, which form the edge of the MPLS infrastructure. A CE device can be a host, a router, or a switch. The PE devices provide Layer 2 virtual bridge connectivity between the CE devices. There can be multiple EVPNs in the provider network. Learning between the PE routers occurs in the control plane using BGP, unlike traditional bridging, where learning occurs in the data plane.

NOTE: In releases earlier than Junos OS Release 15.1, EVPN functionality support on MX Series routers was limited to routers using MPC and MIC interfaces only. Starting with Junos OS Release 15.1, MX Series routers using DPCs can be leveraged to provide EVPN support on the CE device-facing interface.

DPC support for EVPN is provided with the following considerations:

- DPCs provide support for EVPN in the active-standby mode of operation including support for the following:
 - EVPN instance (EVI)
 - Virtual switch
 - Integrated routing and bridging (IRB) interfaces
- DPCs intended for providing the EVPN active-standby support must be the CE device-facing line card. The PE device in the EVPN domain must be MPC interfaces or MIC interfaces.

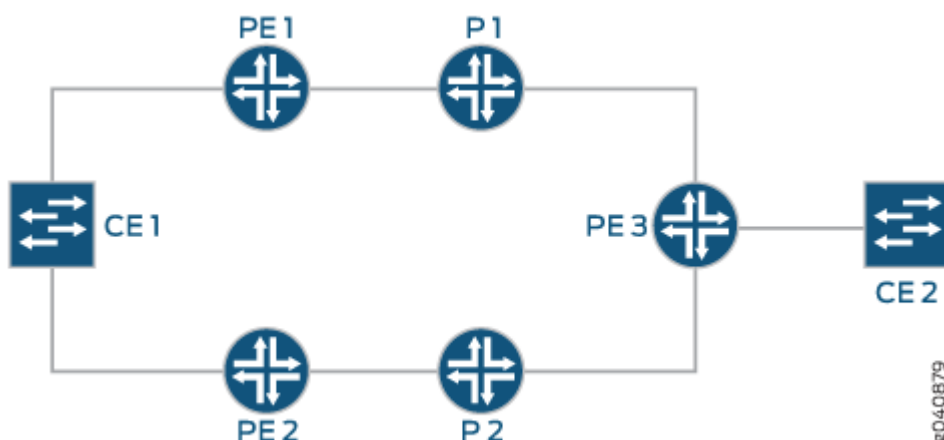
The EVPN multihoming feature enables you to connect a customer site to two or more PE devices to provide redundant connectivity. A CE device can be multihomed to different PE devices or the same PE device. A redundant PE device can provide network service to the customer site as soon as a failure is detected. Thus, EVPN multihoming helps to maintain EVPN service and traffic forwarding to and from the multihomed site in the event of the following types of network failures:

- PE device to CE device link failure
- PE device failure

- MPLS-reachability failure between the local PE device and a remote PE device

Figure 5 on page 98 illustrates how a CE device can multihomed to two PE routers. Device CE 1 is multihomed to Routers PE 1 and PE 2. Device CE 2 has two potential paths to reach Device CE 1, and depending on the multihoming mode of redundancy, only one path or both the paths are active at any time. The multihoming mode of operation also determines which PE router or routers forward traffic to the CE device. The PE router forwarding traffic to the CE device (also called a designated forwarder) uses MPLS LSP or GRE tunnels to forward traffic. If a failure occurs over this path, a new designated forwarder is elected to forward the traffic to Device CE 1.

Figure 5: CE Device Multihomed to Two PE Routers



EVPN MPLS Multihoming Features Supported by QFX10000 Switches

Starting in Junos OS 17.4R1, QFX10000 switches support multihoming for EVPN MPLS. Only active-active multihoming is supported. The following subfeatures are supported:

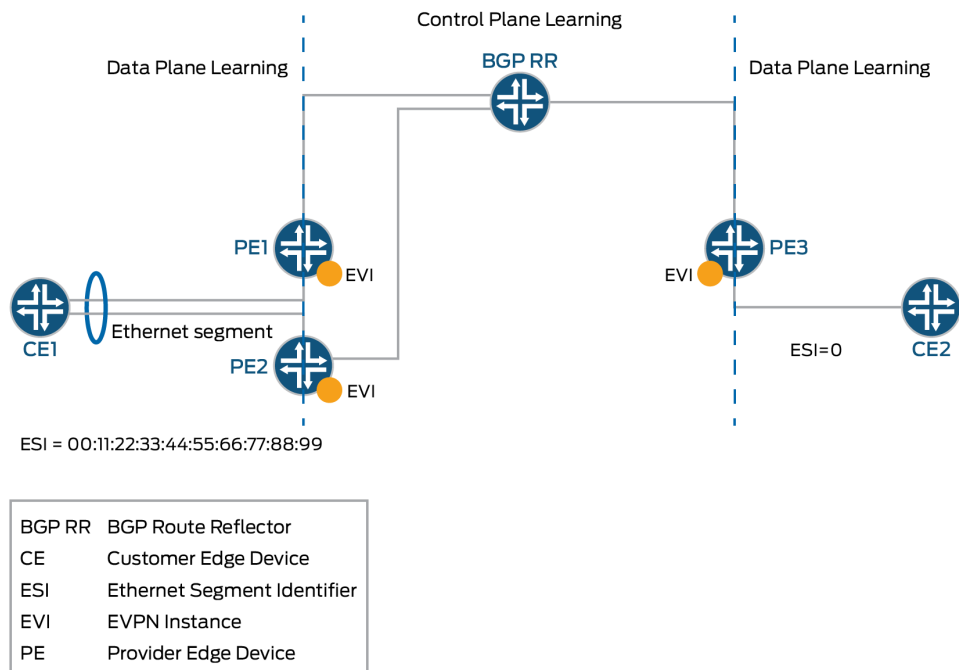
- ESI configuration (only type 0 manual configuration and IFD (physical interfaces) are supported)
- Aliasing and label route
- EVPN route type 4 (Ethernet segment route)
- Extended communities
- BUM traffic
- Designated Forwarder Election (DF) roles: DF and BDF

QFX10000 switches over an MPLS EVPN core only support the default-switch routing instance. An EVPN instance (EVI) is not supported.

Understanding EVPN Multihoming Concepts

Figure 6 on page 99 shows a simple EVPN network topology to define EVPN multihoming concepts. .

Figure 6: Simple EVPN Topology



- **Ethernet segment**—When a CE device is multihomed to two or more PE routers, the set of Ethernet links constitutes an Ethernet segment. An Ethernet segment appears as a link aggregation group (LAG) to the CE device.

The links from Routers PE1 and PE2 to Device CE1 form an Ethernet segment.

In active-standby multihoming, the links that constitute an Ethernet segment form a bridge domain. In active-active multihoming, an Ethernet segment appears as a LAG to the CE device.

- **ESI**—An Ethernet segment must have a unique nonzero identifier, called the Ethernet segment identifier (ESI). The ESI is encoded as a 10-octet integer. When manually configuring an ESI value, the most significant octet, known as the type byte, must be 00. When a single-homed CE device is attached to an Ethernet segment, the entire ESI value is zero.

The Ethernet segment of the multihomed Device CE1 has an ESI value of 00:11:22:33:44:55:66:77:88:99 assigned. The single-homed Device CE2 has an ESI value of 0.

- **EVI**—An EVPN instance (EVI) is an EVPN routing and forwarding instance spanning all the PE routers participating in that VPN. An EVI is configured on the PE routers on a per-customer basis. Each EVI has a unique route distinguisher and one or more route targets.

An EVI is configured on Routers PE1, PE2, and PE3.

- **Ethernet tag**—An Ethernet tag identifies a particular broadcast domain, such as a VLAN. An EVPN instance consists of one or more broadcast domains. Ethernet tags are assigned to the broadcast domains of a given EVPN instance by the provider of that EVPN. Each PE router in that EVPN instance performs a mapping between broadcast domain identifiers understood by each of its attached CE devices and the corresponding Ethernet tag.
- **Ethernet segment route**—The PE routers that are connected to a multihomed CE device use BGP Ethernet segment route messages to discover that each of the PE routers is connected to the same Ethernet segment. The PE routers advertise the Ethernet segment route, which consists of an ESI and ES-import extended community.

Routers PE1 and PE2 advertise an ES route with an ES-import extended community (along with other extended communities like the route target). The PE routers also construct a filter that is based on an ES-import extended community, which results in only these PE routers importing the ES route and identifying that they are connected to the same Ethernet segment.

- **Extended community**— An extended community is similar in most ways to a regular community. EVPNs use extended communities because the 4-octet regular community value does not provide enough expansion and flexibility. An extended community is an 8-octet value divided into two main sections.
- **BUM traffic**—This type of traffic is sent to multiple destinations, including broadcast traffic, unknown unicast traffic that is broadcast in the Ethernet segment, and multicast traffic.
- **DF**—When a CE device is multihomed to two or more PE routers, either one or all of the multihomed PE routers are used to reach the customer site depending on the multihoming mode of operation. The PE router that assumes the primary role for forwarding BUM traffic to the CE device is called the designated forwarder (DF).
- **BDF**—Each router in the set of other PE routers advertising the autodiscovery route per Ethernet segment for the same ESI, and serving as the backup path in case the DF encounters a failure, is called a backup designated forwarder (BDF). A BDF is also called a non-DF router.
- **DF election**—On every Ethernet segment, the PE routers participate in a procedure called *designated forwarder* election to select the DF and the BDF PE routers.

EVPN Multihoming Mode of Operation

The different modes of operation for EVPN multihoming include:

- **Single**—When a PE router is connected to a single-homed customer site, this mode is in operation. The *single* mode is the default mode of operation, and does not require Ethernet segment values to be configured.
- **Active-standby**—When only a single PE router, among a group of PE routers attached to an Ethernet segment, is allowed to forward traffic to and from that Ethernet segment, the Ethernet segment is defined to be operating in the *active-standby* redundancy mode.

To configure the active-standby mode, include the ESI value and the `single-active` statement under the `[edit interfaces]` hierarchy level.

- **Active-active**—When all PE routers attached to an Ethernet segment are allowed to forward traffic to and from the Ethernet segment, the Ethernet segment is defined to be operating in the *active-active* redundancy mode.

NOTE: In Junos OS Release 14.2 and earlier, the EX9200 Series switch supports only the active-standby mode of operation for EVPN multihoming.

NOTE: Starting with Junos OS Release 14.1x53-D30 for QFX5100 switches and Junos OS Release 18.2R1 for EX4600 switches, these switches support the active-active mode of operation for EVPN multihoming. In this scenario, QFX5100 and EX4600 switches function as top-of-rack (ToR) switches in the data center for virtual networks. EVPN multihoming active-active functionality is used to provide access to the bare-metal servers connected to the top-of-rack switches.

NOTE: Starting with Junos OS Release 14.1R4, 14.2, 15.1F6, and 16.1R1, Junos OS supports the active-active mode for EVPN multihoming on MX Series routers.

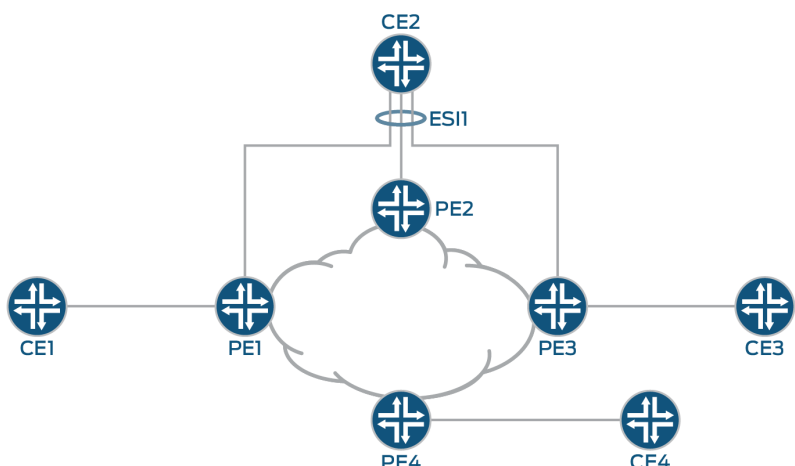
Starting with Junos OS Releases 16.1R4 and 16.2R2, all EX9200 switches support the active-active mode for EVPN multihoming.

Starting with Junos OS Releases 17.4R1 QFX10000 switches support the active-active mode for EVPN multihoming.

To configure the active-active mode, include the ESI value and the `all-active` statement at the `[edit interfaces]` hierarchy level.

Figure 7 on page 102 shows a reference topology for EVPN active-active multihoming. The ESI1 Ethernet segment for Device CE2 is multihomed to Routers PE1, PE2, and PE3. The Ethernet segment on the CE device can either be configured as a link aggregation group (LAG) or as an ECMP path. Devices CE1 and CE3 are the single-homed customer edge devices and have an ESI value of 0. Devices CE4 and PE4 are not shown in the diagram.

Figure 7: Active-Active EVPN Multihoming



EVPN Multihoming Implementation

The EVPN active-standby multihoming mode of operation provides redundancy for access link failures and PE node failure for the multihomed CE device, and is based on the EVPN *draft-ietf-l2vpn-evpn-03*.

The Junos OS implementation of the EVPN multihoming active-standby and active-active modes of operation includes the following:

New BGP NLRIs

To support EVPN multihoming, the following new BGP network layer reachability information (NLRI) routes have been introduced:

Autodiscovery Route per Ethernet Segment

Autodiscovery Route Features

The autodiscovery route NLRI features include:

- This is a Type 1 mandatory route, used for fast convergence and for advertising the split horizon label. It is also known as the mass withdraw route.

- Type 1 route distinguishers are used with the IP address (loopback) of the originating PE router as the route distinguisher value.
- This route carries the ESI in the NLRI (nonzero when it is a multihomed PE, zero otherwise).
- The split horizon label is per ESI only, and carries an explicit NULL (0).
- The bit in the active-standby flag field in the ESI label extended community is used for signaling the active-standby mode (bit set).
- The 3-byte label values in the NLRI and the Ethernet tag is zero.
- This route is advertised and imported by all multihomed and remote PE routers that share the same EVI on the advertising ESI.

Autodiscovery Route Advertisement

- Active-standby mode

In active-standby mode, the designated forwarder (DF) advertises the autodiscovery route per Ethernet segment with an ESI MPLS label extended community that has the standby bit set to 1. The autodiscovery route is advertised per ESI, and the ESI label is set to 0 when active-standby mode is in operation.

The autodiscovery route is imported by all the multihomed and remote PE routers that are part of the EVI. On receiving the autodiscovery route, the PE routers in the network topology learn that active-standby multihoming mode is in operation for the ESI advertised.

- Active-active mode

In active-active mode, each of the multihomed PE device advertises a mandatory autodiscovery route per Ethernet segment as in the active-standby state. However, in the active-active state, the autodiscovery route per Ethernet segment is modified such that the active-standby bit carried in the MPLS extended community is cleared to indicate that the active-active mode is in operation. The autodiscovery route per Ethernet segment in the active-active mode also includes the split horizon label.

In [Figure 7 on page 102](#), for the ESI1 Ethernet segment, Routers PE1, PE2, and PE3 advertise the autodiscovery route. Router PE4 receives this autodiscovery route.

Autodiscovery Route Withdrawal

The autodiscovery route per Ethernet segment withdrawal may result in mass withdrawal. The mass withdrawal feature is used when there is a link failure on the ESI, or when the ESI configuration changes.

When the link between a multihomed CE device and a multihomed PE device fails, the PE device withdraws the autodiscovery route per Ethernet segment. In such a case, the mass withdrawal feature is handled in the following ways by the other PE devices:

- Remote PE device

When a remote PE device receives the BGP update for mass withdrawal, the following is performed at the remote PE device:

1. The current next hop to reach the remote ESI or CE device is deleted.
2. A new next hop through the remaining multihomed PE devices is created to reach the remote ESI or CE device.
3. All the MAC routes behind the CE device are updated with the newly created next hop.

Starting with Junos OS Release 17.4R1, Junos OS supports Dynamic List Next Hops in an EVPN network. Now when the link between the CE device and a multihomed PE device fails, the next hop to the ESI or CE is updated, thus reducing the need for a mass withdrawal. For more information on enabling Dynamic List Next Hop, see ["Configuring Dynamic List Next Hop" on page 332](#).

- Other multihomed PE device

As a result of the mass withdrawal, load balancing on the multihomed CE device happens because of the following:

- When the other multihomed PE devices receive the same set of MAC addresses on the link to the concerned ESI.

In this case, the local routes are preferred. If the remote routes learned from the DF PE device gets withdrawn, it does not affect routes pointing to the local ESI.

- When the other multihomed PE devices have not received the same set of MAC addresses on the link to the concerned ESI.

In this case, the PE devices install the MAC routes pointing to the concerned ESI, although the MACs are remotely learned from the DF PE device. When the DF PE device withdraws these routes, the withdrawn routes are flushed. Packets that are destined to the flushed MAC addresses are flooded on all the local segments.

Ethernet Segment Route

Ethernet Segment Route Features

The Ethernet segment route NRLI features include:

- This is a Type 4 route. The purpose of this route is to enable the PE routers connected to the same Ethernet segment to automatically discover each other with minimal configuration on exchanging this route.
- This route is associated with an ES-import extended community with an ESI value condensed to 6 bytes, similar to a route target.
- This route is advertised and imported only by PE routers that are multihomed on the advertising Ethernet segment.

Ethernet Segment Route Advertisement

The Ethernet segment route is exchanged among all the PE routers within a data center with the ES-import extended community. The ES-import extended community is constructed based on the ESI PE routers that are multihomed, and the Ethernet segment route carries the ESI value related to the Ethernet segment on which the PE routers are multihomed.

The Ethernet segment routes are filtered based on the ES-import extended community, such that only the PE routers that are multihomed on the same Ethernet segment import this route. Each PE router that is connected to a particular Ethernet segment constructs an import filtering rule to import a route that carries the ES-import extended community.

Autodiscovery Route per EVPN Instance

In active-active mode, each of the multihomed PE devices advertise an autodiscovery route per EVPN instance (EVI) with a valid MPLS label. This route is advertised per ESI and is imported by the remote PE devices. The MPLS label included in the autodiscovery route per EVI is used later for aliasing.

New Extended Communities

An extended community is similar in most ways to a regular community. Some networking implementations, such as virtual private networks (VPNs), use extended communities because the 4-octet regular community value does not provide enough expansion and flexibility. An extended community is an 8-octet value divided into two main sections.

To support active-standby multihoming, the following extended communities have been introduced:

ESI-Import

This extended community is attached to the ES route, and is populated from the ESI-import value extracted from the configured ESI value under the interface. To solve the problem of a conflict with another regular route target, the type is set to 0x06, which has been allocated by IANA.

The ESI-import extended community route target populates the list of import route targets configured for the special instance from where the ES route using this community is advertised.

Therefore, incoming ESI routes with the same ESI-import value in the extended community are imported by the PE routers, if the PE router is configured with an Ethernet segment that has the same ESI value. Once the PE router receives a set of these ESI routes that have the same ESI-import extended community value, the DF and BDF election can be done locally.

NOTE: When the ESI-import extended community is not created implicitly, a policy must be configured to attach all the route targets to the autodiscovery route per Ethernet segment.

Split Horizon

With reference to [Figure 7 on page 102](#) for example, when a CE device that is multihomed to two or more PE devices on an Ethernet segment (ESI1) and operating in the active-active redundancy mode sends a BUM packet to one of the non-DF PE devices (say PE1), then Device PE1 forwards that packet to all or a subset of the other PE devices in that EVPN instance, including the DF PE device for that Ethernet segment. In this case the DF PE device that the CE device is multihomed to drops the packet without forwarding it back to the CE device. This filtering is referred to as split horizon.

- Split horizon signaling

The split horizon extended community is attached to the autodiscovery route per Ethernet segment. The value of the extended community is the split horizon or the Poisson label itself, which is 3 bytes, and is advertised as an opaque attribute.

- Split horizon advertisement

- In active-standby mode, the standby bit in the split horizon extended community is set to 1, and the ESI split horizon label is set to 0.
- In the active-active mode, the split horizon extended community is modified to clear the standby bit to 0 and includes a valid ESI label used for split horizon purposes.

- Split horizon MPLS routes

The DF PE device advertises an autodiscovery route per Ethernet segment with a split horizon label A, and an inclusive multicast route with label B for BUM traffic forwarding. On the DF, the BUM packet from the core can come with following labels:

- When the non-DF PE devices receive a BUM packet on their single-homed ESIs, the BUM packet is sent to the DF PE device with multicast label B.

- When the non-DF PE devices receive a BUM packet on ESI1, the BUM packet is sent to the DF PE device with two MPLS labels — the multicast label B as the outer label, and the split horizon label A as the inner label.

In the EVPN multihoming scenario, the multicast label B has the S-bit set to 1 when it is the only label in the label stack. In this case, the BUM packet needs to be flooded on all the local ESIs on the DF PE device. But the label B has the S-bit set to 0 when split horizon label A is the innermost label in the label stack. In this case, the BUM packets need to be flooded on all local ESIs on the DF PE device, except the ESI that maps to the split horizon label A.

Assuming that packets originated from a multihomed CE device to a non-DF PE device on multihomed segment ESI1, when the non-DF PE device sends this packet to the DF PE device, the ESI label that the DF advertised to the non-DF PE device in its autodiscovery route per Ethernet segment is pushed first. The non-DF PE device also pushes the inclusive multicast label that the DF PE device advertised in its inclusive multicast route and further pushes the LSP label. The MPLS header thus contains two labels within a 32-bit field.

The base EVPN functionality uses a table-next hop to stitch the MPLS table with its corresponding EVPN EVI table. In the EVPN EVI table, the mac-lookup is performed to switch the packet.

The following routes are programmed in the mpls.0 table for EVPN multicast:

- The (multicast-label, S=1) route points to the EVPN-EVI table-next hop.
- The (multicast-label, S=0) route points to the MPLS table-next hop. This route loops the packet back to the MPLS table after popping the multicast-label.
- The (split horizon-label) route points to the EVPN-EVI table-next hop. This is the same table-next hop that is used by the multicast-label, S=1 route.

New EVPN Route Types

EVPN multihoming mode supports the following EVPN route types:

- Autodiscovery route per Ethernet segment
- Autodiscovery route per EVPN instance (EVI)
- Ethernet segment route

These route types conform to the following naming convention:

`<route-type>:<RD>::<esi>::<route-specific>/304`

For example:

1. Autodiscovery route per Ethernet segment—1:10.255.0.2:0::112233445566778899::0/304

2. Autodiscovery route per EVI—1:100.100.100.1:1::22222222222222222222::0/304

3. Ethernet segment route—4:10.255.0.1:0::112233445566778899:10.255.0.1/304

where:

- **route-type**—Type of EVPN route.
 - 1—Autodiscovery route per Ethernet segment.
 - 1—Autodiscovery route per EVI.
 - 4—Ethernet segment route.
 - 5—Route with VXLAN/MPLS encapsulation
- **RD**—Route distinguisher value.

The route distinguisher value is set to the IP address of the PE router followed by 0.

- **esi**—Ethernet segment identifier. Displayed as 10 bytes of hexadecimal bytes, and leading 00 bytes are not displayed.
- **route-specific**—Differs per route type.
 - Autodiscovery route per Ethernet segment and autodiscovery route per EVI—This value is an MPLS label.

NOTE: The MPLS label is displayed in the extensive output, although it is not included in the prefix.

- Ethernet segment route—This value is the originating IP address.
- **304**—Maximum number of bits in an EVPN route. This is not very useful information and could be removed from the display. However, it might be useful in quickly identifying an EVPN route, either visually or with match operators.

Multihomed Proxy MAC and IP Address Route Advertisement

Starting in Junos OS Release 18.4R1, Junos sends proxy MAC and IP Address route advertisement from PEs that are multihomed to a CE device. Junos uses a proxy flag in the EVPN layer 2 attributes extended community to identify the message as a proxy MAC and IP Address advertisement. A PE that learns of a MAC and IP Address sends a normal EVPN type 2 (MAC and IP Address) route advertisement. The other PEs on the Ethernet Segment that learns of the new route from the remote PE now send a MAC and IP Address route message with the proxy bit set. If the MAC and IP address entry ages out or if the link between the PE and CE fails, the entries has to be relearned and traffic can be lost. This prevents traffic

loss when one of the connections to a leaf device fails. Multihomed Proxy MAC is automatically enabled.

Update to the MAC Forwarding Table

In active-standby EVPN multihoming, the MAC addresses are treated as routable addresses, and the MP-IBGP protocol is used to carry the customer MAC addresses. MAC learning at the PE routers does not occur in the data plane but in the control plane. This leads to more control applied in terms of the learning mechanism.

A PE router performs MAC learning in the data plane for packets coming from a customer network for a particular EVI. For CE MAC addresses that are behind other PE routers, the MAC addresses are advertised in BGP NLRI using a new MAC advertisement route type.

The MAC learning is of two types:

- Local MAC learning—PE routers must support the local MAC learning process through standard protocols.
- Remote MAC learning—Once the local learning process is completed, the PE routers can advertise the locally learned MAC address to remote PE router nodes through MP-IBGP. This process of receiving the remote MAC addresses of attached customers through MP-IBGP is known as the remote MAC learning process.

The MAC advertisement route type is used to advertise locally learned MAC addresses in BGP to remote PE routers. If an individual MAC address is advertised, the IP address field corresponds to that MAC address. If the PE router sees an ARP request for an IP address from a CE device, and if the PE router has the MAC address binding for that IP address, the PE router performs ARP proxy and responds to the ARP request.

NOTE: The ARP proxy is performed only for the gateway and not for the host.

The MPLS label field depends on the type of allocation. The PE router can advertise a single MPLS label for all MAC addresses per EVI, which requires the least number of MPLS labels and saves the PE router memory. However, when forwarding to the customer network, the PE router must perform a MAC lookup which can cause a delay and increase the number of CPU cycles.

Traffic Flow

In EVPN multihoming, traffic flow is performed in the forwarding-plane. Flood routes are created for flooding the packets, and are used in the following scenarios:

- When a packet is received on a local ESI

- When a packet is received from the core

The traffic flows in EVPN multihoming can be based on the two traffic types:

- Unicast traffic

Unicast traffic is a point-to-point communication with one sender and one receiver. In a multihomed EVPN, unicast traffic is forwarded as follows:

- In active-standby mode
 - CE to core—Traffic is learned and forwarded by the DF PE router.
 - Core to CE—The remote PE router learns the MAC addresses from the DF, and forwards all unicast traffic to the DF PE router.
- In active-active mode
 - CE to core—Traffic is load-balanced to all the connected multihomed PE devices.
 - Core to CE—Traffic from the remote PE devices is load-balanced to all the multihomed PE devices connected to the remote CE device.

- BUM traffic

Traffic that is sent to multiple destinations, including broadcast traffic, unknown unicast traffic that is broadcast in the Ethernet segment, and multicast traffic is known as BUM traffic. In a multihomed EVPN, BUM traffic is forwarded as follows:

- In active-standby mode
 - CE to core—The CE device floods any BUM traffic to all the links in the Ethernet segment. The DF PE router with the active path forwards the BUM packets to the core. The BDF PE router in the standby mode drops all the traffic from the CE device, because the EVPN multihomed status of the interface is in blocking state. However, if the CE device is connected to the PE devices using separate links or LAGs, the BUM traffic reaches both the DF and BDF PE devices.
 - Core to CE—The remote PE routers flood all BUM traffic to both the DF and BDF PE routers. Only the DF forwards the BUM traffic to the CE device. The BDF PE router drops all the traffic, because the EVPN multihomed status of the interface is in blocking state.
- In active-active mode

Based on the requirements, flooding and switching among local ESIs can be enabled or disabled in the active-active mode. This is referred to as the no-local-switching behavior.

The core of EVPN service provides a full-mesh connectivity among the multihomed PE devices. Because of this, EVPN uses split horizon in the core, so a packet received from the core is never

switched or flooded back to the core. Instead, ingress replication is used to replicate the packets to the remote PE devices.

To flood packets to remote PE devices, the multicast and the split horizon next hops are used. The multicast next hop tunnels the packet with the inclusive multicast label, and the split horizon next hop tunnels the packet with a multicast-label and a split horizon label. One such next hop is required per multihomed ESI per remote PE device.

The following flood routes are used in the active-active mode:

- All-CE flood route

This flood route is used by the local ESIs for the following:

- Flooding the packet on the local ESIs (when local-switching is allowed).
- Flooding the packet to the remote PE devices. The remote PE devices flood the packet on their local ESIs.

Because BUM traffic is forwarded only by the Designated Forwarder (DF), and not by the non-DF multihomed PE devices, the non-DFs use the split horizon next hop to flood this packet to other PE devices. However, the multihomed local ESIs for which the PE device is a non-DF does not participate in the flooding.

The all-CE flood route is not used by the non-DF ESIs, and the next hop for these flood routes is created accordingly. In such cases, the non-DF ESI flood route is used.

- All-VE flood route

This flood route is used when the packet is received from the core. It is used for flooding the packet received from the core to the local ESIs. Because the packet received from the core can come with multicast-label only or with both multicast-label and split horizon label, appropriate forwarding rules must be followed to drop the packet on the multihomed ESI that maps to the split horizon label.

- Non-DF flood route

This flood route is used for the following:

- Flooding the packet on the local ESIs.
- Flooding the packet to the remote PE devices using ingress replication with SH-label for the DF for the ESI.

Aliasing

Starting in Junos OS Release 15.1, Junos OS supports aliasing in an EVPN. Aliasing is the ability of a remote PE device to load balance Layer 2 unicast traffic on all the other PE devices that have same Ethernet segment towards a CE device.

Aliasing in the Active-Active Mode

In [Figure 7 on page 102](#), aliasing in the active-active mode works as follows:

1. ESI1 is configured on Routers PE1, PE2, and PE3. Routers PE1, PE2, and PE3 advertise the autodiscovery route per Ethernet segment for ESI1.
2. Device CE1 sends Layer 2 traffic with source MAC address (MAC1) to Router PE1.
3. Router PE1 learns the MAC1 address on (ESI1, vlan X) and advertises it to all PE routers using BGP.
4. Router PE4 receives the MAC1 route through BGP.
5. Because Router PE4 also received the autodiscovery route per EVI from Routers PE2 and PE3, it knows that MAC1 must be reachable through Routers PE2 and PE3. Router PE4 builds its forwarding state to load-balance the Layer 2 traffic for MAC1 among Routers PE1, PE2, and PE3.

Aliasing and Autodiscovery Routes

Autodiscovery routes from Routers PE2 and PE3 can come in any order. As a result, these routes are installed by the Layer 2 process as follows:

1. After receiving MAC1 from Router PE1, and if any autodiscovery routes have not been received by Router PE4, MAC1 is programmed by PE4 with a next hop pointing toward Router PE1. When PE4 receives the autodiscovery route from Router PE2 for the same ESI, the next hop is installed so the traffic for MAC1 is load-balanced to Routers PE1 and PE2. When PE4 receives the autodiscovery route from Router PE3 for the same ESI, the next hop is updated to load-balance the traffic for MAC1 among Routers PE1, PE2, and PE3.
2. If Router PE4 has already received the autodiscovery routes from more than one PE device (PE1, PE2, and PE3), PE4 installs the MAC routes with the multi-destination next hop.

Aliasing and Label Route

Any PE device that advertises the autodiscovery route per EVI with a valid MPLS label programs the advertised label in the mpls.0 routing table. For instance, if Router PE2 advertised the autodiscovery route per EVI with label A, the mpls.0 entry is as follows:

Label A route points to the EVPN-EVI table-next hop.

When the remote Router PE4 sends a unicast data packet toward Router PE2 with this label A, lookup is done in Router PE2's forwarding table, and as a result of this lookup, the packet is forwarded on ESI1.

Aliasing and Unicast Packet Forwarding

When the unicast packets for MAC1 come from the remote Router PE4 to Router PE2, there could be two cases:

- Router PE2 also received the same set of MACs on its link to ESI1—In this case, local routes are preferred and as a result of the MAC lookup, packets are forwarded to ESI1.
- Router PE2 has not received the same set of MACs on its link to ESI1—In this case, Router PE2 still installs MAC routes pointing to ESI1, although MACs are remotely learned from Router PE1. As a result, the packets are forwarded to ESI1.

EVPN Active-Active Multihoming and Multichassis Link Aggregation

When a CE device is configured with a LAG toward the PE devices, the following two options are available to run LACP on the PE devices:

- Configure the same LACP system ID on all the PE devices.
- Configure multichassis link aggregation on the PE devices.

When multichassis link aggregation is configured with EVPN, a reduced set of procedures for active-active multichassis link aggregation are required. These procedures provide link and node level redundancy. The multichassis link aggregation is completely transparent to the CE device, and is realized as pure LAG. Multichassis link aggregation operates at the port level as well. This essentially means that if multichassis link aggregation is configured as active-active, all VLANs on the multichassis link aggregation ports work in the active-active multihoming mode.

When multichassis link aggregation is configured along with EVPN, the following is considered:

- Both multichassis link aggregation and EVPN ESI must be enabled to work in the active-active mode only.
- The following functions are not required for multichassis link aggregation with EVPN:
 - Mac synchronization—This is performed in the BGP control plane of EVPN.
 - ICL linking—This is handled by the aliasing feature of EVPN.
 - ARP synchronization—This is handled by the BGP control plane with IRB functionality.

EVPN Active-Active Multihoming and IRB

When IRB is configured, the EVPN routes contain both MAC and IP information. The active-active multihoming requires ARP synchronization among the multihomed PE devices because the ARP responses can get hashed to a particular PE device.

Sample Configuration

The following is a sample configuration for EVPN active-standby multihoming on the following types of interfaces:

- Ethernet interface configuration

```
ge-0/1/2 {
    encapsulation ethernet-bridge;
    esi XX:XX:XX:XX:XX:XX:XX:XX:XX:XX;
    unit 0 {
        family bridge;
    }
}
```

- Single VLAN interface configuration

```
ge-0/1/3 {
    encapsulation extended-vlan-bridge;
    esi XX:XX:XX:XX:XX:XX:XX:XX:XX:XX;
    vlan-tagging
    unit 0 {
        family bridge;
        vlan-id 1;
    }
}
```

NOTE:

- An ESI value of 0 and all FFs are reserved and are not used for configuring a multihomed Ethernet segment.

- Two interfaces in the same EVI cannot be configured with the same ESI value.

The following is a sample routing instance configuration for EVPN active-standby multihoming:

- Routing instance configuration

```
routing-instances {
  evpn-0 {
    instance-type evpn;
    route-distinguisher value;
    vrf-target value;
    vlan-id vlan-ID;
    interface ge-0/1/2.0;
    interface ge-1/1/1.0;
    interface ge-2/2/2.0;

    protocols {
      evpn {
        designated-forwarder-election hold-time time;
      }
    }
  }
}
```

NOTE: With the active-standby mode configuration, the autodiscovery route per Ethernet segment is advertised with the active-standby bit set to 1 for each Ethernet segment.

Designated Forwarder Election

The following sections discuss DF election:

DF Election Roles

The designated forwarder (DF) election process involves selecting the designated forwarder (DF) PE router and the backup designated forwarder (BDF) or a non-DF (non-designated forwarder PE router roles.

- **DF**—The MAC address from the customer site is reachable only through the PE router announcing the associated MAC advertisement route. This PE router is the primary PE router that is selected to

forward BUM traffic to the multihomed CE device, and is called the designated forwarder (DF) PE router.

- **BDF**—Each PE router in the set of other PE routers advertising the autodiscovery route per Ethernet segment for the same ESI, and serving as the backup path in case the DF encounters a failure, is called a backup designated forwarder (BDF) or a non-DF (non-designated forwarder) PE router.

As a result of the DF election process, if a local PE router is elected as the BDF, the multihomed interface connecting to the customer site is put into a blocking state for the active-standby mode. The interface remains in the blocking state until the PE router is elected as the DF for the Ethernet segment that the interface belongs to.

DF Election as Per RFC 7432

DF Election Procedure

The default procedure for DF election at the granularity of the ESI and EVI is referred to as service carving. With *service carving*, it is possible to elect multiple DFs per Ethernet segment (one per EVI) in order to perform load-balancing of multidestination traffic destined for a given Ethernet segment. The load-balancing procedures carve up the EVI space among the PE nodes evenly, in such a way that every PE is the DF for a disjoint set of EVIs.

The procedure for service carving is as follows:

1. When a PE router discovers the ESI of the attached Ethernet segment, it advertises an autodiscovery route per Ethernet segment with the associated ES-import extended community attribute.
2. The PE router then starts a timer (default value of 3 seconds) to allow the reception of the autodiscovery routes from other PE nodes connected to the same Ethernet segment. This timer value must be the same across all the PE routers connected to the same Ethernet segment.

The default wait timer can be overwritten using the `designated-forwarder-election hold-time` configuration statement.

3. When the timer expires, each PE router builds an ordered list of the IP addresses of all the PE nodes connected to the Ethernet segment (including itself), in increasing numeric order. Every PE router is then given an ordinal indicating its position in the ordered list, starting with 0 as the ordinal for the PE with the numerically lowest IP address. The ordinals are used to determine which PE node is the DF for a given EVI on the Ethernet segment.
4. The PE router that is elected as the DF for a given EVI unblocks traffic for the Ethernet tags associated with that EVI. The DF PE unblocks multidestination traffic in the egress direction toward the Ethernet segment. All the non-DF PE routers continue to drop multidestination traffic (for the associated EVIs) in the egress direction toward the Ethernet segment.

In [Figure 7 on page 102](#), the election of the DF for active-active multihoming is performed among Routers PE1, PE2, and PE3. As a result of this DF election, each one of these routers can become the DF for a particular VLAN from a range of VLANs configured on ESI1. The DF is responsible for forwarding BUM traffic on that ESI and VLAN for which it is elected as the DF. The non-DF PE routers block the BUM traffic on that particular Ethernet segment.

DF Election Trigger

In general, a DF election process is triggered in the following conditions:

- When an interface is newly configured with a nonzero ESI, or when the PE router transitions from an isolated-from-the-core (no BGP session) state to a connected-to-the-core (has established BGP session) state, a wait timer is imposed. By default, the interface is put into a blocking state until the PE router is elected as the DF.
- After completing a DF election process, a PE router receives a new Ethernet segment route or detects the withdrawal of an existing Ethernet segment route, without an imposed wait timer.
- When an interface of a non-DF PE router recovers from a link failure, the PE router has no knowledge of the wait time imposed by other PE routers. As a result, no wait timer is imposed for the recovered PE router to avoid traffic loss.

Preference-Based DF Election

The DF election based on RFC 7432 does not meet some of the operational requirements needed by some service providers. As a solution to this, starting with Junos OS Release 17.3, the DF election in a multihoming EVPN network can be controlled by using an administrative preference value for an ESI.

In the default DF election procedure (as specified in RFC 7432), the DF is elected randomly from one of the multihoming devices with modulo operation. With the preference-based DF election, the DF is elected manually using interface configuration options, such as the preference value, the Don't Preempt (DP) bit, and router ID or loopback address.

Preference-Based DF Election Procedure

The preference-based DF election is supported on EVPN and PBB-EVPN, and allows for manually electing a DF. This is useful when there is a need to choose the DF based on interface attributes, such as the bandwidth associated with an interface.

The preference-based DF election is executed as follows:

1. The DF election type and preference value are configured under an ESI. By default, the preference-based DF election type is based on the modulo (MOD) operation.

2. The configured preference value and DP bit are advertised to the multihoming PE devices using the DF election extended community in the type 4 routes.
3. After receiving the type 4 route, the PE device builds the list of candidate DF devices, in the order of the preference value, DP bit, and IP address.
4. When the DF timer expires, the PE device selects the DF based on the highest preference value.

By default, the DF is elected based on highest preference per EVI. However, the preference-based DF election allows for electing the DF based on the lowest preference value when the `designated-forwarder-preference-least` statement is included at the `[edit routing-instances routing-instance-name protocols evpn]` hierarchy level.

NOTE: The `designated-forwarder-preference-least` configuration should be the same on both the multihoming EVIs; otherwise there can be two DFs causing traffic loss or loop.

5. When the same preference value is configured, then the PE device selects the DF based on the DP bit. When the DP bit is also the same, the DF is elected based on the lowest IP address.

DF Election Algorithm Mismatch

When there is a mismatch between a locally configured DF election algorithm and a remote PE device's DF election algorithm, then all the PE devices should fall back to the default DF election as specified in RFC 7432.

DF Election Algorithm Migration

During the migration of the old DF election to the new DF election, it is expected to change the configuration during the maintenance window by bringing down the ESI, and changing the DF election algorithm.

To do the migration, do the following:

1. After a software upgrade, on the non-DF device bring down all the interfaces that have the same ESI.
2. Configure the new DF election algorithm on the DF PE.
3. Configure the DF election algorithm on other multihoming PE devices.
4. Bring up all the interfaces on the non-DF PE devices.

Changing Preference for Maintenance

After migrating the DF election algorithm, and all the multihoming PE device are running the preference-based DF election algorithm, maintenance tasks required on the existing DF can be executed by simply changing the configured preference value. This changes the DF for a given ESI.

To change the DF for a given ESI:

1. Change the preference value to a higher value on the current non-DF device.
2. Change the preference value to a lower value on the current DF device.

NOTE: Changing the preference value for an ESI can lead to some traffic loss during the short duration required to integrate the delay in the updated BGP route propagation with the new preference value.

DF Election for Virtual Switch

The virtual switch allows multiple bridge domains in a single EVPN instance (EVI). The virtual switch also supports trunk and access ports. Junos OS allows flexible Ethernet services on the port; therefore different VLANs on a single port can be part of different EVIs.

The DF election for virtual switch depends on the following:

- Port mode—Sub-interface, trunk interface, and access port
- EVI mode—Virtual switch with EVPN and EVPN-EVI

In the virtual switch, multiple Ethernet tags can be associated with a single EVI, wherein the numerically lowest Ethernet tag value in the EVI is used for the DF election.

Handling Failover

A failover can occur when:

- The DF PE router loses its DF role.
- There is a link or port failure on the DF PE router.

On losing the DF role, the customer-facing interface on the DF PE router is put in the blocking state.

In the case of link or port failure, a DF election process is triggered, resulting in the BDF PE router to be selected as the DF. At that time, unicast traffic and BUM flow of traffic are affected as follows:

Unicast Traffic

- CE to Core—The CE device continues to flood traffic on all the links. The previous BDF PE router changes the EVPN multihomed status of the interface from the blocking state to the forwarding state, and traffic is learned and forwarded through this PE router.
- Core to CE—The failed DF PE router withdraws the autodiscovery route per Ethernet segment and the locally-learned MAC routes, causing the remote PE routers to redirect traffic to the BDF.

NOTE: The transition of the BDF PE router to the DF role can take some time, causing the EVPN multihomed status of the interface to continue to be in the blocking state, resulting in traffic loss.

BUM Traffic

- CE to Core—All the traffic is routed toward the BDF.
- Core to CE—The remote PE routers flood the BUM traffic in the core.

ESIs on Physical, Aggregated Ethernet, and Logical Interfaces

In releases before Junos OS Release 15.1F6 and 17.1R1 for MX Series routers and Junos OS Release 17.3R1 for EX9200 switches, you can specify an ESI only on a physical or aggregated Ethernet interface, for example, set interfaces ae0 esi 00:11:22:33:44:55:66:77:88:99. If you specify an ESI on a physical or aggregated Ethernet interface, keep in mind that an ESI is a factor in the designated forwarder (DF) election process. For example, assume that you configure EVPN multihoming active-standby on aggregated Ethernet interface ae0, and given the ESI configured on ae0 and other determining factors, the DF election results in ae0 being in the down state. Further, all logical interfaces configured on aggregated Ethernet interface ae0, for example, set interfaces ae0 unit 1 and set interfaces ae0 unit 2 are also in the down state, which renders logical interfaces ae0.1 and ae0.2 unable to provide services to their respective customer sites (VLANs).

To better utilize logical interfaces in EVPN multihoming active-standby or active-active mode, starting with Junos OS Releases 15.1F6 and 17.1R1 for MX Series routers and Junos OS Release 17.3R1 for EX9200 switches, you can specify an ESI on a logical interface. As a result, even if a logical interface is a non-DF, other logical interfaces on the same physical or aggregated Ethernet interface are still able to provide services to their respective customer sites (VLANs).

For more information, see ["Example: Configuring an ESI on a Logical Interface With EVPN Multihoming" on page 336](#).

Automatically Generated ESIs

Starting with Junos OS Release 18.4R1, you can configure aggregated Ethernet interfaces and aggregated Ethernet logical interfaces to automatically derive ESIs from the LACP configuration. We support this feature in the following environments:

- On Juniper Networks devices that support this feature and are multihomed in active-active mode in an EVPN-VLAN overlay network.
- On Juniper Networks devices that support this feature and are multihomed in active-standby or active-active mode in an EVPN-MPLS overlay network.

For more information, see ["Understanding Automatically Generated ESIs in EVPN Networks" on page 352.](#)

Convergence in an EVPN Network

When there are changes in the network topology in a large-scale EVPN system, the convergence time might be significant. You can prioritize NLRI updates that are critical to route selection in routing policies to improve convergence. [Table 7 on page 121](#) lists the NLRI route types and the priority that must be configured in the routing policy.

Table 7: Priority for NLRI Route Type

NLRI Route Type	Description	Priority
NLRI Route Type 1	Ethernet auto-discovery route—Type 1 supports fast convergence and aliasing and is used to signal MAC mass withdrawal.	High
NLRI Route Type 2	MAC/IP advertisement route—Type 2 is used to advertise MAC addresses and IP addresses in EVPN networks.	Low
NLRI Route Type 3	Inclusive multicast Ethernet tag—Type 3 is used to set up a path for BUM traffic.	Low
NLRI Route Type 4	Ethernet segment route—Type 4 is used in the selection of a designated forwarder.	High

To prioritize the NLRI route type, set the `bgp-output-queue-priority` priority for `nlri-route-type` at the `[edit policy-options policy-statement]` hierarchy level on all provider edge routers and route reflectors in the

EVPN network. In this example, a high priority was configured for NRLI route type 1 and NRLI route type 4.

```
user@PE1#show policy-options
policy-statement evpn-rt-priority-policy {
  term 1 {
    from {
      family evpn;
      nlri-route-type 1;
    }
    then {
      bgp-output-queue-priority priority 16;
    }
  }
  term 2 {
    from {
      family evpn;
      nlri-route-type 2;
    }
    then {
      bgp-output-queue-priority priority 1;
    }
  }
  term 3 {
    from {
      family evpn;
      nlri-route-type 3;
    }
    then {
      bgp-output-queue-priority priority 2;
    }
  }
  term 4 {
    from {
      family evpn;
      nlri-route-type 4;
    }
    then {
      bgp-output-queue-priority priority 16;
    }
  }
}
```

```
}
}
```

NOTE: There are 17 prioritized output queues: an expedited queue that has the highest priority, and 16 numbered queues for which 1 is the lowest priority and 16 is the highest.

For more information about how to configure routing policies, see [Routing Policies, Firewall Filters, and Traffic Policers User Guide](#).

Release History Table

Release	Description
18.4R1	Starting with Junos OS Release 18.4R1, you can configure aggregated Ethernet interfaces and aggregated Ethernet logical interfaces to automatically derive ESIs from the LACP configuration.
17.4R1	Starting with Junos OS Release 17.4R1, Junos OS supports Dynamic List Next Hops in an EVPN network.
16.1R4	Starting with Junos OS Releases 16.1R4 and 16.2R2, all EX9200 switches support the active-active mode for EVPN multihoming.
16.1R4	Starting with Junos OS Releases 17.4R1 QFX10000 switches support the active-active mode for EVPN multihoming.
15.1F6	To better utilize logical interfaces in EVPN multihoming active-standby or active-active mode, starting with Junos OS Releases 15.1F6 and 17.1R1 for MX Series routers and Junos OS Release 17.3R1 for EX9200 switches, you can specify an ESI on a logical interface. As a result, even if a logical interface is a non-DF, other logical interfaces on the same physical or aggregated Ethernet interface are still able to provide services to their respective customer sites (VLANs).
15.1	Starting in Junos OS Release 15.1, Junos OS supports aliasing in an EVPN.
14.1x53-D30	Starting with Junos OS Release 14.1x53-D30 for QFX5100 switches and Junos OS Release 18.2R1 for EX4600 switches, these switches support the active-active mode of operation for EVPN multihoming.
14.1R4	Starting with Junos OS Release 14.1R4, 14.2, 15.1F6, and 16.1R1, Junos OS supports the active-active mode for EVPN multihoming on MX Series routers.

RELATED DOCUMENTATION

[Configuring Dynamic List Next Hop | 332](#)

[Example: Configuring EVPN Active-Standby Multihoming | 154](#)

[Example: Configuring EVPN Active-Active Multihoming | 213](#)

Configuring EVPN Active-Standby Multihoming to a Single PE Device

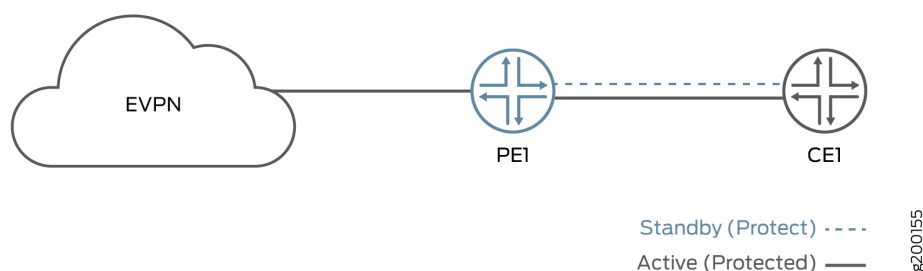
You can configure EVPN active-standby multihoming on a single PE device by configuring a standby (protect) interface to provide redundancy to an active (protected) interface. [Figure 8 on page 124](#) illustrates an EVPN topology with active-standby multihoming to a single PE device. A protect interface provides the benefit of a backup for the protected primary interface in case of failure. The PE device uses the primary interface for network traffic while the primary interface is functioning. If the primary interface fails, the protect interface becomes active and traffic switches to the protect interface. When the primary interface returns, the primary interface becomes the active interface once again.

NOTE: If connectivity fault management (CFM) is enabled, the protect interface will trigger CFM to send Type, Length, and Value (TLV) or Remote Defect Indication (RDI) interface status to the customer edge device

Junos OS does not support the protect interface in the following cases:

- EVPN-VXLAN
- PBB-EVPN
- On an interface that has been configured as an ESI in EVPN multihoming

Figure 8: EVPN Active-Standby Multihoming on a Single PE Device



To configure the protect interface in a routing instance, configure both the protect interface and the primary interface in the routing instance for EVPN and include the protect-interface statement in the primary interface hierarchy.

```

routing-instances {
    routing-instance-name{
        instance-type type;
        interface primary-interface-name {
            protect-interface protect-interface-name
        }
        interface protect-interface-name
    }
    route-distinguisher (as-number:number / ip-address:number);
    vrf-target community;
}

```

To configure the protect interface in a routing instance, configure both the protect interface and the primary interface in the routing instance for EVPN-VPWS and include the protect-interface statement in the evpn protocol hierarchy.

```

routing-instances {
    routing-instance-name {
        instance-type evpn-vpws;
        interface primary-interface-name ;
        interface protect-interface-name;
        route-distinguisher (as-number:number / ip-address:number);
        vrf-target community;
        protocols {
            evpn {
                interface primary-interface-name {
                    vpws-service-id {
                        local service-id;
                        remote service-id;
                    }
                    protect-interface protect-interface-name
                }
            }
        }
    }
}

```


To configure the protect interface in a bridge domain, configure both the protect interface and the primary interface in the bridge domain and include the protect-interface statement in the primary interface hierarchy.

```
bridge-domains {
  bridge-domain-name{
    domain-type bridge;
    vlan-id number;
    interface primary-interface-name {
      protect-interface protect-interface-name
    }
    interface protect-interface-name
  }
}
```

To display the protect interface, use the show evpn instance extensive operational command.

```
user@PE1> show evpn instance extensive
Instance: blue
Route Distinguisher: 10.255.255.1:100
Per-instance MAC route label: 299776
MAC database status Local Remote
MAC advertisements: 0 0
MAC+IP advertisements: 0 0
Default gateway MAC advertisements: 0 0
Number of local interfaces: 5 (5 up)
Interface name ESI Mode Status AC-Role
ae0.0 00:11:22:33:44:55:66:77:88:99 all-active Up Root
ge-0/0/3.0 00:00:00:00:00:00:00:00:00:00 single-homed Up Root
ge-0/0/4.0 00:11:11:11:44:55:66:77:88:99 all-active Up Root
ge-0/0/4.1 00:22:22:22:44:55:66:77:88:99 all-active Up Root
ge-0/0/4.50 00:00:00:00:00:00:00:00:00:00 single-homed Up Root
Number of IRB interfaces: 1 (0 up)
Interface name VLAN VNI Status L3 context
irb.1 25 Down vrf
Number of protect interfaces: 1
Interface name Protect Interface name Status
ge-0/0/3.1 ge-0/0/4.50 Protect-inactive
```

The show interfaces detail command shows that the protect interface has a CCC-DOWN status

```

user@PE1> show interfaces ae0.0 detail
Logical interface ae80.0 (Index 399) (SNMP ifIndex 612) (Generation 223)
  Flags: Up SNMP-Traps CCC-Down 0x20004000 VLAN-Tag [ 0x8100.10 ] Encapsulation: VLAN-Bridge
  Statistics      Packets      pps      Bytes      bps
Bundle:
  Input :          0          0          0          0
  Output:          0          0          0          0
Adaptive Statistics:
  Adaptive Adjusts:      0
  Adaptive Scans :      0
  Adaptive Updates:      0
Link:
  ge-0/1/8.0
  Input :          0          0          0          0
  Output:          0          0          0          0

Aggregate member links: 1

Marker Statistics:  Marker Rx      Resp Tx      Unknown Rx      Illegal Rx
ge-0/1/8.0          0          0          0          0
Protocol bridge, MTU: 1522, Generation: 263, Route table: 11, Mesh Group: __all_ces__

```

RELATED DOCUMENTATION

| *protect-interface*

Configuring EVPN Active-Standby Multihoming

You can configure an Ethernet VPN (EVPN) with multihoming support to provide multihoming functionality with active-standby redundancy mode of operation in the EVPN and virtual switch routing instance. This mode enables autodiscovery of Ethernet segments, Ethernet segment route construction, and Ethernet segment identifier (ESI) label assignment.

When configuring active-standby EVPN multihoming, be aware of the following limitations:

- An interface or ESI can be attached to more than one EVPN instance (EVI), with a maximum limit of 200 EVIs per ESI.
- For an EVPN routing instance, only one logical interface per physical interface or ESI can be attached to an EVI.
- For a virtual switch routing instance, only one logical interface per physical interface or ESI can be configured under a bridge domain.
- All the PE routers in the network topology should be running Junos OS Release 14.1 or later releases, which are based on the EVPN draft-ietf-l2vpn-evpn-03. Junos OS releases prior to 14.1 support the older version of the EVPN draft, causing interoperability issues when Junos OS Release 14.1 and a previous release are running.

Before you begin:

1. Configure the router interfaces.
2. Configure the router ID and autonomous system number for the device.
3. Configure OSPF or any other IGP protocol.
4. Configure a BGP internal group.
5. Include the EVPN signaling network layer reachability information (NLRI) to the internal BGP group.
6. Configure LDP.
7. Configure MPLS.
8. Configure RSVP MPLS LSP or GRE tunnels.

To configure the PE device:

1. Enable EVPN active-standby multihoming on the multihomed interfaces.

```
[edit interfaces]
user@PE1# set interface-name vlan-tagging
user@PE1# set interface-name encapsulation flexible-ethernet-services
user@PE1# set interface-name esi esi-value
user@PE1# set interface-name esi single-active
user@PE1# set interface-name unit 0 encapsulation vlan-bridge
user@PE1# set interface-name unit 0 vlan-id VLAN-ID
```

For example:

```
[edit interfaces]
user@PE1# set ge-0/0/4 vlan-tagging
user@PE1# set ge-0/0/4 encapsulation flexible-ethernet-services
user@PE1# set ge-0/0/4 esi 00:22:44:66:88:00:22:44:66:88
user@PE1# set ge-0/0/4 esi single-active
user@PE1# set ge-0/0/4 unit 0 encapsulation vlan-bridge
user@PE1# set ge-0/0/4 unit 0 vlan-id 300
```

2. Configure the routing instance for the active-standby mode of redundancy.

The active-standby multihoming can be configured under any EVPN routing-instance. Both evpn and virtual-switch instance types are supported in active-standby EVPN multihoming. The vrf routing-instance is configured to illustrate the EVPN IRB functionality, in addition to multihoming, and is not mandatory for the active-standby EVPN multihoming feature to work. For example:

Virtual-switch Routing Instance

```
[edit routing-instances]
user@PE1# set virtual-switch-instance instance-type virtual-switch
user@PE1# set virtual-switch-instance protocols evpn extended-vlan-list VLAN-ID
user@PE1# set virtual-switch-instance bridge-domains bridge-domain-name domain-type bridge
user@PE1# set virtual-switch-instance bridge-domains bridge-domain-name vlan-id VLAN-ID
user@PE1# set virtual-switch-instance bridge-domains bridge-domain-name interface interface-name
user@PE1# set virtual-switch-instance bridge-domains bridge-domain-name routing-interface interface-name
user@PE1# set virtual-switch-instance route-distinguisher route-distinguisher-value
user@PE1# set virtual-switch-instance vrf-target vrf-target
```

OR

EVPN Routing Instance

```
[edit routing-instances]
user@PE1# set evpn-instance instance-type evpn
user@PE1# set evpn-instance vlan-id VLAN-ID
user@PE1# set evpn-instance interface interface-name
user@PE1# set evpn-instance routing-interface interface-name
user@PE1# set evpn-instance route-distinguisher route-distinguisher-value
user@PE1# set evpn-instance vrf-target vrf-target
```

OR

VRF Routing Instance

```
[edit routing-instances]
user@PE1# set vrf-instance instance-type vrf
user@PE1# set vrf-instance interface interface-name
user@PE1# set vrf-instance route-distinguisher route-distinguisher-value
user@PE1# set vrf-instance vrf-target vrf-target
```

3. Verify and commit the configuration.

For example:

```
[edit routing-instances]
user@PE1# set ALPHA instance-type virtual-switch
user@PE1# set ALPHA route-distinguisher 10.255.0.1:100
user@PE1# set ALPHA vrf-target target:100:100
user@PE1# set ALPHA protocols evpn extended-vlan-list 100
user@PE1# set ALPHA bridge-domains ONE domain-type bridge
user@PE1# set ALPHA bridge-domains ONE vlan-id 100
user@PE1# set ALPHA bridge-domains ONE interface ae0.0
user@PE1# set ALPHA bridge-domains ONE interface ge-0/0/2.0
user@PE1# set ALPHA bridge-domains ONE routing-interface irb.0
user@PE1# set BETA instance-type evpn
user@PE1# set BETA vlan-id 300
user@PE1# set BETA interface ge-0/0/4.0
user@PE1# set BETA interface ae1.0
user@PE1# set BETA routing-interface irb.1
user@PE1# set BETA route-distinguisher 10.255.0.1:300
user@PE1# set BETA vrf-target target:300:300
user@PE1# set DELTA instance-type vrf
user@PE1# set DELTA interface irb.0
user@PE1# set DELTA interface irb.1
user@PE1# set DELTA route-distinguisher 10.255.0.1:200
user@PE1# set DELTA vrf-target target:200:200
user@PE1# set DELTA vrf-table-label
```

```
[edit]
user@PE1# commit
commit complete
```

RELATED DOCUMENTATION

[Example: Configuring EVPN Active-Standby Multihoming](#) | 154

Example: Configuring Basic EVPN Active-Standby Multihoming

IN THIS SECTION

- [Requirements](#) | 131
- [Overview and Topology](#) | 132
- [Configuration](#) | 132
- [Verification](#) | 143

This example shows how to configure Ethernet VPN (EVPN) active-standby multihoming.

Requirements

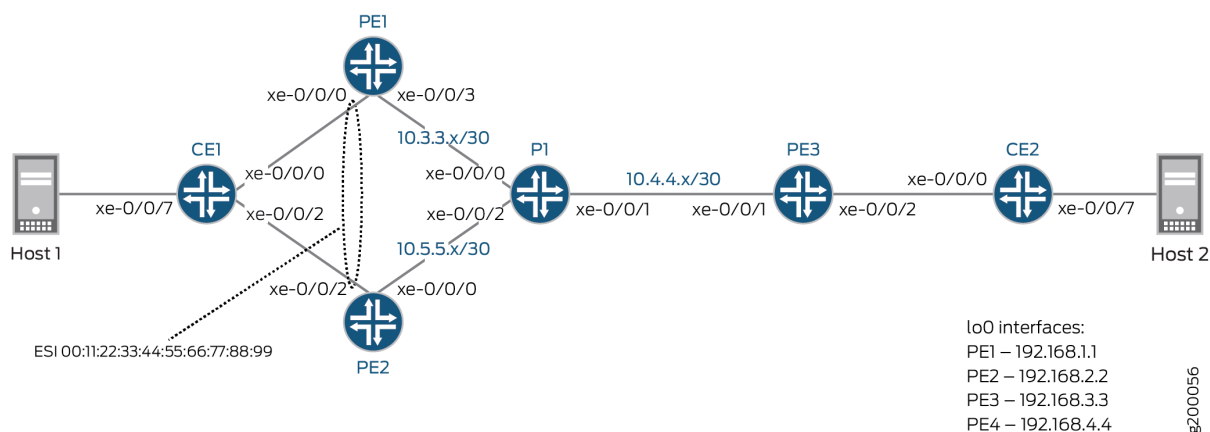
This example uses the following hardware and software components:

- Four MX Series 5G Universal Routing Platforms running Junos OS Release 14.1 (or later), with MPC interfaces, acting as provider edge (PE) and provider (P) routers.
- Two customer edge (CE) devices.

Overview and Topology

Figure 9 on page 132 illustrates a simple EVPN topology. Routers PE1 and PE2 are provider edge (PE) routers connected to multihomed customer edge (CE) Router CE1. Router PE3 is connected to CE Router CE2.

Figure 9: Simple EVPN Multihomed Topology



The network has the following characteristics:

- All PE and P routers are running OSPF.
- There is an IBGP mesh between all PE routers.
- MPLS (RSVP) LSPs are configured between all PE routers.
- On Routers PE1 and PE2, each device's CE-facing interface uses the same Ethernet Segment Identifier (ESI).

Configuration

IN THIS SECTION

- [CLI Quick Configuration | 132](#)

CLI Quick Configuration

The configurations for each device is as follows:

CE1

```

interfaces {
  xe-0/0/0 {
    description to-PE1;
    unit 0 {
      family bridge {
        interface-mode trunk;
        vlan-id-list 10;
      }
    }
  }
  xe-0/0/2 {
    description to-PE2;
    unit 0 {
      family bridge {
        interface-mode trunk;
        vlan-id-list 10;
      }
    }
  }
  xe-0/0/7 {
    description to-Host;
    unit 0 {
      family bridge {
        interface-mode access;
        vlan-id 10;
      }
    }
  }
}
bridge-domains {
  BD {
    vlan-id-list 10;
    bridge-options {
      no-mac-learning; ## Used with single-active PE configurations, ensures traffic is
always flooded to both PEs in case of a DF change.
    }
  }
}

```


CE2

```

interfaces {
  xe-0/0/0 {
    description to-PE3;
    unit 0 {
      family bridge {
        interface-mode trunk;
        vlan-id-list 10;
      }
    }
  }
  xe-0/0/7 {
    description to-Host;
    unit 0 {
      family bridge {
        interface-mode access;
        vlan-id 10;
      }
    }
  }
}
bridge-domains {
  BD {
    vlan-id-list 10;
  }
}

```

PE1

```

interfaces {
  xe-0/0/0 {
    description to-CE1;
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    esi {
      00:11:22:33:44:55:66:77:88:99;
      single-active;
    }
    unit 10 {
      family bridge {

```

```

        interface-mode trunk;
        vlan-id-list 10;
    }
}
xe-0/0/3 {
    description to-P;
    unit 0 {
        family inet {
            address 10.3.3.1/30;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.1.1/32;
        }
    }
}
}
routing-options {
    router-id 192.168.1.1;
    autonomous-system 65432;
    forwarding-table {
        export evpn-pplb;
    }
}
protocols {
    rsvp {
        interface xe-0/0/3.0;
    }
    mpls {
        no-cspf;
        label-switched-path PE1-to-PE2 {
            to 192.168.2.2;
        }
        label-switched-path PE1-to-PE3 {
            to 192.168.3.3;
        }
        interface xe-0/0/3.0;
    }
}

```

```

bgp {
    group EVPN-PE {
        type internal;
        local-address 192.168.1.1;
        family evpn {
            signaling;
        }
        neighbor 192.168.2.2;
        neighbor 192.168.3.3;
    }
}

ospf {
    area 0.0.0.0 {
        interface xe-0/0/3.0;
        interface lo0.0;
    }
}

}

policy-options {
    policy-statement evpn-pplb {
        from protocol evpn;
        then {
            load-balance per-packet;
        }
    }
}

routing-instances {
    EVPN-RI {
        instance-type virtual-switch;
        interface xe-0/0/0.10;
        route-distinguisher 192.168.1.1:10;
        vrf-target target:65432:10;
        protocols {
            evpn {
                extended-vlan-list 10;
            }
        }
        bridge-domains {
            bd10 {
                domain-type bridge;
                vlan-id 10;
            }
        }
    }
}

```

```
    }
}
```

PE2

```
interfaces {
  xe-0/0/0 {
    description to-P;
    unit 0 {
      family inet {
        address 10.5.5.1/30;
      }
      family mpls;
    }
  }
  xe-0/0/2 {
    description to-CE1;
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    esi {
      00:11:22:33:44:55:66:77:88:99;
      single-active;
    }
    unit 10 {
      family bridge {
        interface-mode trunk;
        vlan-id-list 10;
      }
    }
  }
  lo0 {
    unit 0 {
      family inet {
        address 192.168.2.2/32;
      }
    }
  }
}
routing-options {
  router-id 192.168.2.2;
  autonomous-system 65432;
  forwarding-table {
```

```

        export evpn-pplb;
    }
}
protocols {
    rsvp {
        interface xe-0/0/0.0;
    }
    mpls {
        no-cspf;
        label-switched-path PE2-to-PE1 {
            to 192.168.1.1;
        }
        label-switched-path PE2-to-PE3 {
            to 192.168.3.3;
        }
        interface xe-0/0/0.0;
    }
    bgp {
        group EVPN-PE {
            type internal;
            local-address 192.168.2.2;
            family evpn {
                signaling;
            }
            neighbor 192.168.1.1;
            neighbor 192.168.3.3;
        }
    }
    ospf {
        area 0.0.0.0 {
            interface xe-0/0/0.0;
            interface lo0.0;
        }
    }
}
policy-options {
    policy-statement evpn-pplb {
        from protocol evpn;
        then {
            load-balance per-packet;
        }
    }
}
}

```

```

routing-instances {
  EVPN-RI {
    instance-type virtual-switch;
    interface xe-0/0/2.10;
    route-distinguisher 192.168.2.2:10;
    vrf-target target:65432:10;
    protocols {
      evpn {
        extended-vlan-list 10;
      }
    }
    bridge-domains {
      bd10 {
        domain-type bridge;
        vlan-id 10;
      }
    }
  }
}

```

PE3

```

interfaces {
  xe-0/0/1 {
    description to-P;
    unit 0 {
      family inet {
        address 10.4.4.1/30;
      }
      family mpls;
    }
  }
  xe-0/0/2 {
    description to-CE3;
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 10 {
      family bridge {
        interface-mode trunk;
        vlan-id-list 10;
      }
    }
  }
}

```

```

}
lo0 {
    unit 0 {
        family inet {
            address 192.168.3.3/32;
        }
    }
}
}
routing-options {
    router-id 192.168.3.3;
    autonomous-system 65432;
    forwarding-table {
        export evpn-pplb;
    }
}
protocols {
    rsvp {
        interface xe-0/0/1.0;
    }
    mpls {
        no-cspf;
        label-switched-path PE3-to-PE1 {
            to 192.168.1.1;
        }
        label-switched-path PE3-to-PE2 {
            to 192.168.2.2;
        }
        interface xe-0/0/1.0;
    }
    bgp {
        group EVPN-PE {
            type internal;
            local-address 192.168.3.3;
            family evpn {
                signaling;
            }
            neighbor 192.168.1.1;
            neighbor 192.168.2.2;
        }
    }
    ospf {
        area 0.0.0.0 {

```

```

        interface xe-0/0/1.0;
        interface lo0.0;
    }
}
}
policy-options {
    policy-statement evpn-pplb {
        from protocol evpn;
        then {
            load-balance per-packet;
        }
    }
}
routing-instances {
    EVPN-RI {
        instance-type virtual-switch;
        interface xe-0/0/2.10;
        route-distinguisher 192.168.3.3:10;
        vrf-target target:65432:10;
        protocols {
            evpn {
                extended-vlan-list 10;
            }
        }
        bridge-domains {
            bd10 {
                domain-type bridge;
                vlan-id 10;
            }
        }
    }
}
}

```

P1

```

interfaces {
    xe-0/0/0 {
        unit 0 {
            family inet {
                address 10.3.3.2/30;
            }
            family mpls;
        }
    }
}

```



```

    }
}
xe-0/0/1 {
    unit 0 {
        family inet {
            address 10.4.4.2/30;
        }
        family mpls;
    }
}
xe-0/0/2 {
    unit 0 {
        family inet {
            address 10.5.5.2/30;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.4.4/32;
        }
    }
}
}
routing-options {
    router-id 192.168.4.4;
    autonomous-system 65432;
}
protocols {
    rsvp {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
    mpls {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
}

```

```
ospf {
  area 0.0.0.0 {
    interface xe-0/0/0.0;
    interface xe-0/0/1.0;
    interface xe-0/0/2.0;
    interface lo0.0;
  }
}
```

Verification

IN THIS SECTION

Verifying OSPF | 143

Verifying BGP | 144

Verifying MPLS | 145

Verifying EVPN Configuration and Multihoming Status | 147

Verifying Route Exchange and ESI Autodiscovery | 150

Verifying Ethernet Segment (ES) Route Exchange | 152

Confirm that the configuration is working properly.

Verifying OSPF

Purpose

Verify that OSPF is working properly.

Action

Verify that Router P1 has adjacencies established with all PE devices.

```
user@P1> show ospf neighbor
```

Address	Interface	State	ID	Pri	Dead
10.3.3.1	xe-0/0/0.0	Full	192.168.1.1	128	33

10.4.4.1	xe-0/0/1.0	Full	192.168.3.3	128	38
10.5.5.1	xe-0/0/2.0	Full	192.168.2.2	128	37

Meaning

Adjacencies have been established with the PE devices.

Verifying BGP

Purpose

Verify that BGP is working properly.

Action

Verify that MP-IBGP peerings are established using EVPN signaling between all PE devices.

```
user@PE1> show bgp summary
Groups: 1 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State   Pending
bgp.evpn.0
                4          4          0          0          0          0
Peer           AS      InPkt    OutPkt    OutQ    Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
192.168.2.2     65432      89       55        0        1    22:18 Establ
  EVPN-RI.evpn.0: 2/2/2/0
  bgp.evpn.0: 3/3/3/0
  __default_evpn__.evpn.0: 1/1/1/0
192.168.3.3     65432      59       48        0        1    22:18 Establ
  EVPN-RI.evpn.0: 1/1/1/0
  bgp.evpn.0: 1/1/1/0
  __default_evpn__.evpn.0: 0/0/0/0
```

```
user@PE2> show bgp summary

Groups: 1 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State   Pending
bgp.evpn.0
                5          5          0          0          0          0
Peer           AS      InPkt    OutPkt    OutQ    Flaps Last Up/Dwn State|#Active/
```

```

Received/Accepted/Damped...
192.168.1.1          65432      80      50      0      1      22:49 Establ
  bgp.evpn.0: 4/4/4/0
  EVPN-RI.evpn.0: 3/3/3/0
  __default_evpn__.evpn.0: 1/1/1/0
192.168.3.3          65432      73      87      0      0      27:26 Establ
  bgp.evpn.0: 1/1/1/0
  EVPN-RI.evpn.0: 1/1/1/0
  __default_evpn__.evpn.0: 0/0/0/0

user@PE3> show bgp summary
Groups: 1 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State   Pending
bgp.evpn.0
                5          5          0          0          0          0
Peer           AS      InPkt    OutPkt    OutQ    Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
192.168.1.1     65432      66      51      0      1      23:05 Establ
  bgp.evpn.0: 3/3/3/0
  EVPN-RI.evpn.0: 3/3/3/0
  __default_evpn__.evpn.0: 0/0/0/0
192.168.2.2     65432     104      64      0      0      27:42 Establ
  bgp.evpn.0: 2/2/2/0
  EVPN-RI.evpn.0: 2/2/2/0
  __default_evpn__.evpn.0: 0/0/0/0

```

Meaning

EVPN-signaled MP-IBGP peerings have been established between all PE devices.

Verifying MPLS

Purpose

Verify that MPLS is working properly.

Action

Verify that MPLS LSPs are established between all PE devices.

```
user@PE1> show mpls lsp
```

Ingress LSP: 2 sessions

To	From	State	Rt	P	ActivePath	LSPname
192.168.2.2	192.168.1.1	Up	0	*		PE1-to-PE2
192.168.3.3	192.168.1.1	Up	0	*		PE1-to-PE3

Total 2 displayed, Up 2, Down 0

Egress LSP: 2 sessions

To	From	State	Rt	Style	Labelin	Labelout	LSPname
192.168.1.1	192.168.2.2	Up	0	1 FF	3	-	PE2-to-PE1
192.168.1.1	192.168.3.3	Up	0	1 FF	3	-	PE3-to-PE1

Total 2 displayed, Up 2, Down 0

Transit LSP: 0 sessions

Total 0 displayed, Up 0, Down 0

```
user@PE2> show mpls lsp
```

Ingress LSP: 2 sessions

To	From	State	Rt	P	ActivePath	LSPname
192.168.1.1	192.168.2.2	Up	0	*		PE2-to-PE1
192.168.3.3	192.168.2.2	Up	0	*		PE2-to-PE3

Total 2 displayed, Up 2, Down 0

Egress LSP: 2 sessions

To	From	State	Rt	Style	Labelin	Labelout	LSPname
192.168.2.2	192.168.3.3	Up	0	1 FF	3	-	PE3-to-PE2
192.168.2.2	192.168.1.1	Up	0	1 FF	3	-	PE1-to-PE2

Total 2 displayed, Up 2, Down 0

Transit LSP: 0 sessions

Total 0 displayed, Up 0, Down 0

```
user@PE3> show mpls lsp
```

Ingress LSP: 2 sessions

To	From	State	Rt	P	ActivePath	LSPname
192.168.1.1	192.168.3.3	Up	0	*		PE3-to-PE1

```

192.168.2.2    192.168.3.3    Up    0 *                PE3-to-PE2
Total 2 displayed, Up 2, Down 0

Egress LSP: 2 sessions
To            From            State   Rt  Style Labelin Labelout LSPname
192.168.3.3   192.168.1.1   Up      0  1 FF      3      - PE1-to-PE3
192.168.3.3   192.168.2.2   Up      0  1 FF      3      - PE2-to-PE3
Total 2 displayed, Up 2, Down 0

Transit LSP: 0 sessions
Total 0 displayed, Up 0, Down 0

```

Meaning

LSPs have been established between PE devices.

Verifying EVPN Configuration and Multihoming Status

Purpose

Verify that EVPN is configured properly.

Action

Verify that the EVPN routing instances and ESIs are configured and functioning correctly, and confirm that single-active multihoming is enabled.

```

user@PE1> show evpn instance EVPN-RI extensive
Instance: EVPN-RI
Route Distinguisher: 192.168.1.1:10
Per-instance MAC route label: 300128
MAC database status          Local  Remote
MAC advertisements:         0      0
MAC+IP advertisements:      0      0
Default gateway MAC advertisements: 0      0
Number of local interfaces: 1 (1 up)
Interface name  ESI                      Mode      Status    AC-Role
xe-0/0/0.10    00:11:22:33:44:55:66:77:88:99  single-active  Up        Root
Number of IRB interfaces: 0 (0 up)
Number of bridge domains: 1
VLAN  Domain ID  Intfs / up  IRB intf  Mode      MAC sync  IM route label

```

```

10          1    1          Extended    Enabled    300240
Number of neighbors: 2
  Address          MAC      MAC+IP      AD      IM      ES Leaf-label
  192.168.2.2      0        0          1        1        0
  192.168.3.3      0        0          0        1        0
Number of ethernet segments: 1
ESI: 00:11:22:33:44:55:66:77:88:99
  Status: Resolved by IFL xe-0/0/0.10
  Local interface: xe-0/0/0.10, Status: Up/Forwarding
  Number of remote PEs connected: 1
    Remote PE      MAC label  Aliasing label  Mode
    192.168.2.2    0          0              single-active
Designated forwarder: 192.168.1.1
  Backup forwarder: 192.168.2.2
  Last designated forwarder update: Jun 26 23:30:35
  Advertised MAC label: 300224
  Advertised aliasing label: 300224
  Advertised split horizon label: 300256

user@PE2> show evpn instance EVPN-RI extensive
Instance: EVPN-RI
Route Distinguisher: 192.168.2.2:10
Per-instance MAC route label: 300384
MAC database status          Local  Remote
MAC advertisements:          0      0
MAC+IP advertisements:       0      0
Default gateway MAC advertisements: 0      0
Number of local interfaces: 1 (1 up)
  Interface name  ESI                      Mode      Status    AC-Role
  xe-0/0/2.10     00:11:22:33:44:55:66:77:88:99 single-active  Up      Root
Number of IRB interfaces: 0 (0 up)
Number of bridge domains: 1
  VLAN  Domain ID  Intfs / up  IRB intf  Mode      MAC sync  IM route label
  10      1      1          Extended  Enabled   300608
Number of neighbors: 2
  Address          MAC      MAC+IP      AD      IM      ES Leaf-label
  192.168.1.1      0        0          2        1        0
  192.168.3.3      0        0          0        1        0
Number of ethernet segments: 1
ESI: 00:11:22:33:44:55:66:77:88:99
  Status: Resolved by NH 1048575
  Local interface: xe-0/0/2.10, Status: Up/Blocking

```

```

Number of remote PEs connected: 1
  Remote PE      MAC label  Aliasing label  Mode
  192.168.1.1    0          300224         single-active
Designated forwarder: 192.168.1.1
Backup forwarder: 192.168.2.2
Last designated forwarder update: Jun 26 23:30:43
Advertised MAC label: 300544
Advertised aliasing label: 300544
Advertised split horizon label: 300320

user@PE3> show evpn instance EVPN-RI extensive
Instance: EVPN-RI
Route Distinguisher: 192.168.3.3:10
Per-instance MAC route label: 300272
MAC database status
Local Remote
MAC advertisements:      0      0
MAC+IP advertisements:  0      0
Default gateway MAC advertisements: 0      0
Number of local interfaces: 1 (1 up)
  Interface name  ESI                               Mode      Status  AC-Role
  xe-0/0/2.10    00:00:00:00:00:00:00:00:00:00 single-homed Up      Root
Number of IRB interfaces: 0 (0 up)
Number of bridge domains: 1
  VLAN  Domain ID  Intfs / up  IRB intf  Mode      MAC sync  IM route label
  10          1      1          Extended  Enabled  300368
Number of neighbors: 2
  Address      MAC    MAC+IP    AD      IM      ES Leaf-label
  192.168.1.1    0      0        2      1      0
  192.168.2.2    0      0        1      1      0
Number of ethernet segments: 1
ESI: 00:11:22:33:44:55:66:77:88:99
Status: Resolved by NH 1048574
Number of remote PEs connected: 2
  Remote PE      MAC label  Aliasing label  Mode
  192.168.1.1    0          300224         single-active
  192.168.2.2    0          0             single-active

```

Meaning

From the outputs above, the following can be determined:

- All three PE devices confirm that PE1 and PE2 are using single-active mode.
- PE1 and PE2 are using the same ESI.
- PE1 is elected as the designated forwarder (DF), and its CE-facing interface is put into a state of **Up/Forwarding**.
- PE2 is elected as the backup designated forwarder (BDF), and its CE-facing interface is put into a state of **Up/Blocking**.

Verifying Route Exchange and ESI Autodiscovery

Purpose

Verify that EVPN signaling is working properly.

Action

Verify that autodiscovery and other signaling information is being shared between PE devices.

```

user@PE1> show route table EVPN-RI.evpn.0

EVPN-RI.evpn.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:192.168.1.1:10::112233445566778899::0/304 AD/EVI
    *[EVPN/170] 00:19:27
    Indirect
1:192.168.2.2:0::112233445566778899::FFFF:FFFF/304 AD/ESI
    *[BGP/170] 00:18:20, localpref 100, from 192.168.2.2
    AS path: I, validation-state: unverified
    > to 10.3.3.2 via xe-0/0/3.0, label-switched-path PE1-to-PE2
3:192.168.1.1:10::10::192.168.1.1/304 IM
    *[EVPN/170] 00:19:31
    Indirect
3:192.168.2.2:10::10::192.168.2.2/304 IM
    *[BGP/170] 00:18:19, localpref 100, from 192.168.2.2
    AS path: I, validation-state: unverified
    > to 10.3.3.2 via xe-0/0/3.0, label-switched-path PE1-to-PE2
3:192.168.3.3:10::10::192.168.3.3/304 IM
    *[BGP/170] 00:18:13, localpref 100, from 192.168.3.3
    AS path: I, validation-state: unverified
    > to 10.3.3.2 via xe-0/0/3.0, label-switched-path PE1-to-PE3

```

```
user@PE2> show route table EVPN-RI.evpn.0
```

```
EVPN-RI.evpn.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
```

```
+ = Active Route, - = Last Active, * = Both
```

```
1:192.168.1.1:10::112233445566778899::0/304 AD/EVI
    *[BGP/170] 00:18:51, localpref 100, from 192.168.1.1
    AS path: I, validation-state: unverified
    > to 10.5.5.2 via xe-0/0/0.0, label-switched-path PE2-to-PE1
1:192.168.1.1:0::112233445566778899::FFFF:FFFF/304 AD/ESI
    *[BGP/170] 00:18:51, localpref 100, from 192.168.1.1
    AS path: I, validation-state: unverified
    > to 10.5.5.2 via xe-0/0/0.0, label-switched-path PE2-to-PE1
3:192.168.1.1:10::10::192.168.1.1/304 IM
    *[BGP/170] 00:18:51, localpref 100, from 192.168.1.1
    AS path: I, validation-state: unverified
    > to 10.5.5.2 via xe-0/0/0.0, label-switched-path PE2-to-PE1
3:192.168.2.2:10::10::192.168.2.2/304 IM
    *[EVPN/170] 00:18:45
    Indirect
3:192.168.3.3:10::10::192.168.3.3/304 IM
    *[BGP/170] 00:18:40, localpref 100, from 192.168.3.3
    AS path: I, validation-state: unverified
    > to 10.5.5.2 via xe-0/0/0.0, label-switched-path PE2-to-PE3
```

```
user@PE3> show route table EVPN-RI.evpn.0
```

```
EVPN-RI.evpn.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
```

```
+ = Active Route, - = Last Active, * = Both
```

```
1:192.168.1.1:10::112233445566778899::0/304 AD/EVI
    *[BGP/170] 00:18:54, localpref 100, from 192.168.1.1
    AS path: I, validation-state: unverified
    > to 10.4.4.2 via xe-0/0/1.0, label-switched-path PE3-to-PE1
1:192.168.1.1:0::112233445566778899::FFFF:FFFF/304 AD/ESI
    *[BGP/170] 00:18:54, localpref 100, from 192.168.1.1
    AS path: I, validation-state: unverified
    > to 10.4.4.2 via xe-0/0/1.0, label-switched-path PE3-to-PE1
1:192.168.2.2:0::112233445566778899::FFFF:FFFF/304 AD/ESI
    *[BGP/170] 00:18:54, localpref 100, from 192.168.2.2
```

```

        AS path: I, validation-state: unverified
        > to 10.4.4.2 via xe-0/0/1.0, label-switched-path PE3-to-PE2
3:192.168.1.1:10::10::192.168.1.1/304 IM
        *[BGP/170] 00:18:54, localpref 100, from 192.168.1.1
        AS path: I, validation-state: unverified
        > to 10.4.4.2 via xe-0/0/1.0, label-switched-path PE3-to-PE1
3:192.168.2.2:10::10::192.168.2.2/304 IM
        *[BGP/170] 00:18:54, localpref 100, from 192.168.2.2
        AS path: I, validation-state: unverified
        > to 10.4.4.2 via xe-0/0/1.0, label-switched-path PE3-to-PE2
3:192.168.3.3:10::10::192.168.3.3/304 IM
        *[EVPN/170] 00:18:53
        Indirect

```

Meaning

The outputs above show two EVPN route types:

- **Route Type 1: Ethernet Auto-Discovery (AD) Route** - These routes are advertised on a per-EVI and per-ESI basis. Ethernet AD routes are required when a CE device is multihomed. When a CE device is single-homed, the ESI will be zero.
- **Route Type 3: Inclusive Multicast Ethernet Tag Route** - This route sets up a path for broadcast, unknown unicast, and multicast (BUM) traffic from a PE device to the remote PE device on a per VLAN, per ESI basis.

The outputs above show the following information:

- **1:192.168.x.x:10::112233445566778899::0/304 AD/EVI** - This is the per-EVI AD Type 1 EVPN route. As the DF (and active device), Router PE1 has advertised this route to Routers PE2 and PE3.
- **1:192.168.x.x:0::112233445566778899::FFFF:FFFF/304 AD/ESI** - This is the per Ethernet segment AD Type 1 EVPN route. As the multihomed devices, Routers PE1 and PE2 have advertised this route to each other and to Router PE3.
- **3:192.168.x.x:10::10::192.168.x.x/304 IM** - This is the route used to set up a path for BUM traffic. Each PE device has advertised this route to the other PE device.

Verifying Ethernet Segment (ES) Route Exchange

Purpose

Verify that ES route information is being shared correctly.

Action

Verify that the local and advertised autodiscovery routes per Ethernet segment and the Ethernet segment routes are received.

```
user@PE1> show route table __default_evpn__.evpn.0

__default_evpn__.evpn.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:192.168.1.1:0::112233445566778899::FFFF:FFFF/304 AD/ESI
    *[EVPN/170] 00:14:22
    Indirect
4:192.168.1.1:0::112233445566778899:192.168.1.1/304 ES
    *[EVPN/170] 00:14:23
    Indirect
4:192.168.2.2:0::112233445566778899:192.168.2.2/304 ES
    *[BGP/170] 00:14:14, localpref 100, from 192.168.2.2
    AS path: I, validation-state: unverified
    > to 10.3.3.2 via xe-0/0/3.0, label-switched-path PE1-to-PE2

user@PE2> show route table __default_evpn__.evpn.0

__default_evpn__.evpn.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:192.168.2.2:0::112233445566778899::FFFF:FFFF/304 AD/ESI
    *[EVPN/170] 00:14:25
    Indirect
4:192.168.1.1:0::112233445566778899:192.168.1.1/304 ES
    *[BGP/170] 00:14:24, localpref 100, from 192.168.1.1
    AS path: I, validation-state: unverified
    > to 10.5.5.2 via xe-0/0/0.0, label-switched-path PE2-to-PE1
4:192.168.2.2:0::112233445566778899:192.168.2.2/304 ES
    *[EVPN/170] 00:14:26
    Indirect
```

Meaning

The outputs above show two EVPN route types:

- Route Type 1: Ethernet Auto-Discovery (AD) Route - These routes are advertised on a per-EVI and per-ESI basis. Ethernet AD routes are required when a CE device is multihomed. When a CE device is single-homed, the ESI will be zero.
- Route Type 4: Ethernet Segment Route - PE devices that are connected to the same Ethernet Segment will discover each other through the ES route.

The outputs above show the following information:

- 1:192.168.x.x:0::112233445566778899::FFFF:FFFF/304 AD/ESI - This is the per Ethernet segment AD Type 1 EVPN route. In the outputs above, each PE device shows its own route.
- 4:192.168.x.x:0::112233445566778899:192.168.x.x/304 ES - This is the ES route for the local ESI. In the outputs above, each PE device shows both its own route and the one advertised by the other PE device.

RELATED DOCUMENTATION

[Example: Configuring EVPN Active-Standby Multihoming | 154](#)

[Example: Configuring Basic EVPN Active-Active Multihoming | 200](#)

[Example: Configuring EVPN Active-Active Multihoming | 213](#)

Example: Configuring EVPN Active-Standby Multihoming

IN THIS SECTION

- [Requirements | 155](#)
- [Overview and Topology | 155](#)
- [Configuration | 158](#)
- [Verification | 175](#)

This example shows how to configure Ethernet VPN (EVPN) for multihomed customer edge (CE) devices in an EVPN, virtual switch, and VRF routing instance, and with an integrated routing and bridging (IRB) interface configuration.

Requirements

This example uses the following hardware and software components:

- Four MX Series 5G Universal Routing Platforms with MPC interfaces only, where:
 - Two devices are configured as provider edge (PE) routers connected to a common multihomed customer site.
 - One device is configured as a remote PE router connected to a single-homed customer site.
- Eight customer edge (CE) devices, where:
 - Two CE devices are multihomed.
 - Two CE devices are single-homed for each of the PE routers.
- Junos OS Release 14.1 or later running on all the PE routers.

NOTE: Junos OS Release 14.1 and later releases are based on the EVPN draft-ietf-l2vpn-evpn-03. Releases prior to 14.1, support the older version of the EVPN draft, causing interoperability issues when Junos OS Release 14.1 and a previous release are running.

Before you begin:

1. Configure the router interfaces.
2. Configure OSPF or any other IGP protocol.
3. Configure BGP.
4. Configure LDP.
5. Configure MPLS.
6. Configure RSVP MPLS LSP or GRE tunnels.

Overview and Topology

Starting with Junos OS Release 14.1, the EVPN solution on MX Series routers with MPC interfaces is extended to provide multihoming functionality with active-standby mode of operation. The multihoming functions include autodiscovery of Ethernet segments, Ethernet segment route construction, and Ethernet segment identifier (ESI) label assignment.

NOTE: Prior to Junos OS Release 15.1, the EVPN functionality support on MX Series Routers was limited to routers using MPC and MIC interfaces only. However, starting with Junos OS Release 15.1, MX Series Routers using DPCs can be leveraged to provide EVPN support on the CE device-facing interface.

The DPC support for EVPN is provided with the following considerations:

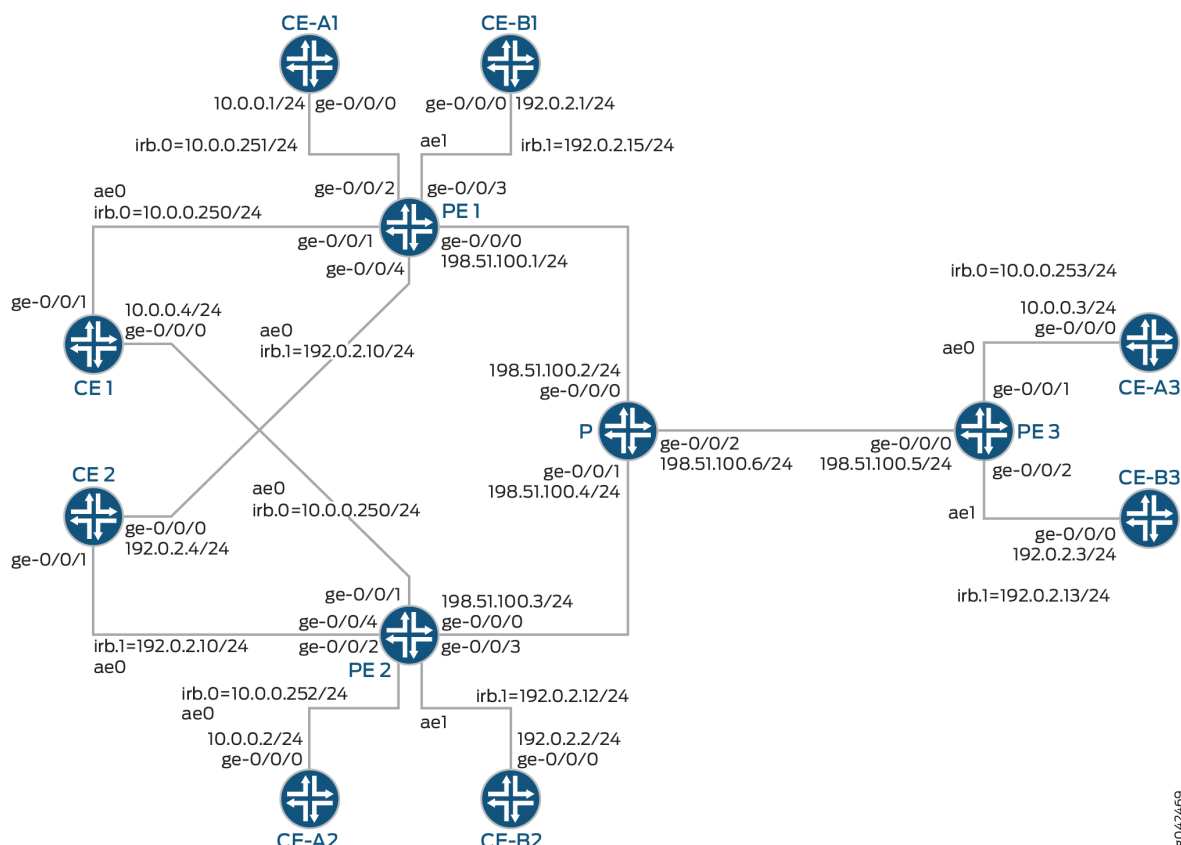
- DPCs provide support for EVPN in the active-standby mode of operation including support for the following:
 1. EVPN instance (EVI)
 2. Virtual switch (VS)
 3. Integrated routing and bridging (IRB) interfaces
- DPCs intended for providing the EVPN active-standby support should be the CE device-facing line card. The PE device interfaces in the EVPN domain should use only MPC and MIC interfaces.

NOTE: When configuring active-standby EVPN multihoming, be aware of the following limitations:

- An interface or ESI can be attached to more than one EVI, with a maximum limit of 200 EVIs per ESI.
- For an EVPN routing instance, only one logical interface per physical interface or ESI can be attached to an EVI.
- For a virtual switch routing instance, only one logical interface per physical interface or ESI can be configured under a bridge domain.
- All the PE routers in the network topology should be running Junos OS Release 14.1 or later releases, which are based on the EVPN draft-ietf-12vpn-evpn-03. Junos OS releases prior to 14.1

support the older version of the EVPN draft, causing interoperability issues when Junos OS Release 14.1 and a previous release are running.

Figure 10: EVPN Active-Standby Multihoming



In Figure 10 on page 157:

- Routers PE1 and PE2 are provider edge (PE) routers connected to multihomed customer edge (CE) devices “Device CE1” and “Device CE1.”
- Router PE3 is a remote PE router connected to a single-homed customer site.
- Router P is the provider router connected to Routers PE1, PE2, and PE3.
- There are three routing instances running in the topology—ALPHA, BETA, and DELTA—with the virtual switch, EVPN, and VRF type of routing instance, respectively.

- All the PE routers are connected to one single-homed CE device each for the ALPHA and BETA routing instances.
- Device CE1 belongs to the ALPHA routing instance, and Device CE2 belongs to the BETA routing instance.

For Router PE1, Device CE-A1 and Device CE-B1 are the single-homed CE devices for the routing instances ALPHA and BETA, respectively. In the same way, Device CE-A2 and Device CE-A3 belong to the ALPHA routing instance, and Device CE-B2 and Device CE-B3 belong to the BETA routing instances connected to Routers PE2 and PE3, respectively.

NOTE: The active-standby multihoming can be configured under any EVPN routing-instance. Both `evpn` and `virtual-switch` instance types are supported in active-standby EVPN multihoming. The `vrf` routing-instance is configured to illustrate the EVPN IRB functionality, in addition to multihoming, and is not mandatory for the active-standby EVPN multihoming feature to work.

Configuration

IN THIS SECTION

- [CLI Quick Configuration | 158](#)
- [Procedure | 166](#)
- [Results | 171](#)

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the `[edit]` hierarchy level.

CE1

```
set chassis aggregated-devices ethernet device-count 1
set interfaces ge-0/0/0 gigether-options 802.3ad ae0
set interfaces ge-0/0/1 gigether-options 802.3ad ae0
set interfaces ae0 vlan-tagging
set interfaces ae0 mac 00:00:00:00:00:04
```

```
set interfaces ae0 unit 0 vlan-id 100
set interfaces ae0 unit 0 family inet address 10.0.0.4/24
set routing-options static route 0.0.0.0/0 next-hop 10.0.0.250
```

CE-A1

```
set chassis aggregated-devices ethernet device-count 1
set interfaces ge-0/0/0 gigether-options 802.3ad ae0
set interfaces ae0 vlan-tagging
set interfaces ae0 mac 00:00:00:00:00:01
set interfaces ae0 unit 0 vlan-id 100
set interfaces ae0 unit 0 family inet address 10.0.0.1/24
set routing-options static route 0.0.0.0/0 next-hop 10.0.0.251
```

CE-A2

```
set chassis aggregated-devices ethernet device-count 1
set interfaces ge-0/0/0 gigether-options 802.3ad ae0
set interfaces ae0 vlan-tagging
set interfaces ae0 mac 00:00:00:00:00:02
set interfaces ae0 unit 0 vlan-id 100
set interfaces ae0 unit 0 family inet address 10.0.0.2/24
set routing-options static route 0.0.0.0/0 next-hop 10.0.0.252
```

CE-A3

```
set chassis aggregated-devices ethernet device-count 1
set interfaces ge-0/0/0 gigether-options 802.3ad ae0
set interfaces ae0 vlan-tagging
set interfaces ae0 mac 00:00:00:00:00:03
set interfaces ae0 unit 0 vlan-id 100
set interfaces ae0 unit 0 family inet address 10.0.0.3/24
set routing-options static route 0.0.0.0/0 next-hop 10.0.0.253
```

CE2

```
set chassis aggregated-devices ethernet device-count 1
set interfaces ge-0/0/0 gigether-options 802.3ad ae0
set interfaces ge-0/0/1 gigether-options 802.3ad ae0
set interfaces ae0 vlan-tagging
```

```

set interfaces ae0 mac 00:00:00:00:00:04
set interfaces ae0 unit 0 vlan-id 300
set interfaces ae0 unit 0 family inet address 192.0.2.4/24
set routing-options static route 0.0.0.0/0 next-hop 192.0.2.10

```

CE-B1

```

set chassis aggregated-devices ethernet device-count 1
set interfaces ge-0/0/0 gigether-options 802.3ad ae0
set interfaces ae0 vlan-tagging
set interfaces ae0 mac 00:00:00:00:00:01
set interfaces ae0 unit 0 vlan-id 300
set interfaces ae0 unit 0 family inet address 192.0.2.1/24
set routing-options static route 0.0.0.0/0 next-hop 192.0.2.15

```

CE-B2

```

set chassis aggregated-devices ethernet device-count 1
set interfaces ge-0/0/0 gigether-options 802.3ad ae0
set interfaces ae0 vlan-tagging
set interfaces ae0 mac 00:00:00:00:00:02
set interfaces ae0 unit 0 vlan-id 300
set interfaces ae0 unit 0 family inet address 192.0.2.4/24
set routing-options static route 0.0.0.0/0 next-hop 192.0.2.12

```

CE-B3

```

set chassis aggregated-devices ethernet device-count 1
set interfaces ge-0/0/0 gigether-options 802.3ad ae0
set interfaces ae0 vlan-tagging
set interfaces ae0 mac 00:00:00:00:00:03
set interfaces ae0 unit 0 vlan-id 300
set interfaces ae0 unit 0 family inet address 192.0.2.3/24
set routing-options static route 0.0.0.0/0 next-hop 192.0.2.13

```

PE1

```

set interfaces ge-0/0/0 unit 0 family inet address 198.51.100.1/24
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 gigether-options 802.3ad ae0

```

```

set interfaces ge-0/0/2 vlan-tagging
set interfaces ge-0/0/2 encapsulation flexible-ethernet-services
set interfaces ge-0/0/2 unit 0 encapsulation vlan-bridge
set interfaces ge-0/0/2 unit 0 vlan-id 100
set interfaces ge-0/0/3 gigether-options 802.3ad ae1
set interfaces ge-0/0/4 vlan-tagging
set interfaces ge-0/0/4 encapsulation flexible-ethernet-services
set interfaces ge-0/0/4 esi 00:22:44:66:88:00:22:44:66:88
set interfaces ge-0/0/4 esi single-active
set interfaces ge-0/0/4 unit 0 encapsulation vlan-bridge
set interfaces ge-0/0/4 unit 0 vlan-id 300
set interfaces ae0 vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 esi 00:11:22:33:44:55:66:77:88:99
set interfaces ae0 esi single-active
set interfaces ae0 unit 0 encapsulation vlan-bridge
set interfaces ae0 unit 0 vlan-id 100
set interfaces ae1 vlan-tagging
set interfaces ae1 encapsulation flexible-ethernet-services
set interfaces ae1 unit 0 encapsulation vlan-bridge
set interfaces ae1 unit 0 vlan-id 300
set interfaces irb unit 0 family inet address 10.0.0.250/24
set interfaces irb unit 0 family inet address 10.0.0.251/24
set interfaces irb unit 0 mac 00:11:22:33:44:55
set interfaces irb unit 1 family inet address 192.0.2.10/24
set interfaces irb unit 1 family inet address 192.0.2.15/24
set interfaces irb unit 1 mac 00:22:44:66:88:00
set interfaces lo0 unit 0 family inet address 10.255.0.1/32 primary
set interfaces lo0 unit 0 family inet address 10.255.0.1/32 preferred
set routing-options router-id 10.255.0.1
set routing-options autonomous-system 100
set routing-options forwarding-table chained-composite-next-hop ingress evpn
set protocols mpls interface lo0.0
set protocols mpls interface ge-0/0/0.0
set protocols bgp group EVPN-PE type internal
set protocols bgp group EVPN-PE local-address 10.255.0.1
set protocols bgp group EVPN-PE family evpn signaling
set protocols bgp group EVPN-PE neighbor 10.255.0.2
set protocols bgp group EVPN-PE neighbor 10.255.0.3
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ldp interface ge-0/0/0.0
set protocols ldp interface lo0.0

```

```

set routing-instances ALPHA instance-type virtual-switch
set routing-instances ALPHA route-distinguisher 10.255.0.1:100
set routing-instances ALPHA vrf-target target:100:100
set routing-instances ALPHA protocols evpn extended-vlan-list 100
set routing-instances ALPHA bridge-domains ONE domain-type bridge
set routing-instances ALPHA bridge-domains ONE vlan-id 100
set routing-instances ALPHA bridge-domains ONE interface ae0.0
set routing-instances ALPHA bridge-domains ONE interface ge-0/0/2.0
set routing-instances ALPHA bridge-domains ONE routing-interface irb.0
set routing-instances BETA instance-type evpn
set routing-instances BETA vlan-id 300
set routing-instances BETA interface ge-0/0/4.0
set routing-instances BETA interface ae1.0
set routing-instances BETA routing-interface irb.1
set routing-instances BETA route-distinguisher 10.255.0.1:300
set routing-instances BETA vrf-target target:300:300
set routing-instances DELTA instance-type vrf
set routing-instances DELTA interface irb.0
set routing-instances DELTA interface irb.1
set routing-instances DELTA route-distinguisher 10.255.0.1:200
set routing-instances DELTA vrf-target target:200:200
set routing-instances DELTA vrf-table-label

```

PE2

```

set interfaces ge-0/0/0 unit 0 family inet address 198.51.100.3/24
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 gigether-options 802.3ad ae0
set interfaces ge-0/0/2 vlan-tagging
set interfaces ge-0/0/2 encapsulation flexible-ethernet-services
set interfaces ge-0/0/2 unit 0 encapsulation vlan-bridge
set interfaces ge-0/0/2 unit 0 vlan-id 100
set interfaces ge-0/0/3 gigether-options 802.3ad ae1
set interfaces ge-0/0/4 vlan-tagging
set interfaces ge-0/0/4 encapsulation flexible-ethernet-services
set interfaces ge-0/0/4 esi 00:22:44:66:88:00:22:44:66:88
set interfaces ge-0/0/4 esi single-active
set interfaces ge-0/0/4 unit 0 encapsulation vlan-bridge
set interfaces ge-0/0/4 unit 0 vlan-id 300
set interfaces ae0 vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 esi 00:11:22:33:44:55:66:77:88:99

```

```

set interfaces ae0 esi single-active
set interfaces ae0 unit 0 encapsulation vlan-bridge
set interfaces ae0 unit 0 vlan-id 100
set interfaces ae1 vlan-tagging
set interfaces ae1 encapsulation flexible-ethernet-services
set interfaces ae1 unit 0 encapsulation vlan-bridge
set interfaces ae1 unit 0 vlan-id 300
set interfaces irb unit 0 family inet address 10.0.0.250/24
set interfaces irb unit 0 family inet address 10.0.0.252/24
set interfaces irb unit 0 mac 00:11:22:33:44:55
set interfaces irb unit 1 family inet address 192.0.2.10/24
set interfaces irb unit 1 family inet address 192.0.2.12/24
set interfaces irb unit 1 mac 00:22:44:66:88:00
set interfaces lo0 unit 0 family inet address 10.255.0.2/32 primary
set interfaces lo0 unit 0 family inet address 10.255.0.2/32 preferred
set routing-options router-id 10.255.0.2
set routing-options autonomous-system 100
set routing-options forwarding-table chained-composite-next-hop ingress evpn
set protocols mpls interface lo0.0
set protocols mpls interface ge-0/0/0.0
set protocols bgp group EVPN-PE type internal
set protocols bgp group EVPN-PE local-address 10.255.0.2
set protocols bgp group EVPN-PE family evpn signaling
set protocols bgp group EVPN-PE neighbor 10.255.0.1
set protocols bgp group EVPN-PE neighbor 10.255.0.3
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ldp interface ge-0/0/0.0
set protocols ldp interface lo0.0
set routing-instances ALPHA instance-type virtual-switch
set routing-instances ALPHA route-distinguisher 10.255.0.2:100
set routing-instances ALPHA vrf-target target:100:100
set routing-instances ALPHA protocols evpn extended-vlan-list 100
set routing-instances ALPHA bridge-domains ONE domain-type bridge
set routing-instances ALPHA bridge-domains ONE vlan-id 100
set routing-instances ALPHA bridge-domains ONE interface ae0.0
set routing-instances ALPHA bridge-domains ONE interface ge-0/0/2.0
set routing-instances ALPHA bridge-domains ONE routing-interface irb.0
set routing-instances BETA instance-type evpn
set routing-instances BETA vlan-id 300
set routing-instances BETA interface ge-0/0/4.0
set routing-instances BETA interface ae1.0
set routing-instances BETA routing-interface irb.1

```

```

set routing-instances BETA route-distinguisher 10.255.0.2:300
set routing-instances BETA vrf-target target:300:300
set routing-instances DELTA instance-type vrf
set routing-instances DELTA interface irb.0
set routing-instances DELTA interface irb.1
set routing-instances DELTA route-distinguisher 10.255.0.2:200
set routing-instances DELTA vrf-target target:200:200
set routing-instances DELTA vrf-table-label

```

PE3

```

set interfaces ge-0/0/0 unit 0 family inet address 198.51.100.5/24
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 gigether-options 802.3ad ae0
set interfaces ge-0/0/2 gigether-options 802.3ad ae1
set interfaces ae0 vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 unit 0 encapsulation vlan-bridge
set interfaces ae0 unit 0 vlan-id 100
set interfaces ae1 vlan-tagging
set interfaces ae1 encapsulation flexible-ethernet-services
set interfaces ae1 unit 0 encapsulation vlan-bridge
set interfaces ae1 unit 0 vlan-id 300
set interfaces irb unit 0 family inet address 10.0.0.253/24
set interfaces irb unit 1 family inet address 192.0.2.13/24
set interfaces lo0 unit 0 family inet address 10.255.0.3/32 primary
set interfaces lo0 unit 0 family inet address 10.255.0.3/32 preferred
set routing-options router-id 10.255.0.3
set routing-options autonomous-system 100
set routing-options forwarding-table chained-composite-next-hop ingress evpn
set protocols mpls interface lo0.0
set protocols mpls interface ge-0/0/0.0
set protocols bgp group EVPN-PE type internal
set protocols bgp group EVPN-PE local-address 10.255.0.3
set protocols bgp group EVPN-PE family evpn signaling
set protocols bgp group EVPN-PE neighbor 10.255.0.1
set protocols bgp group EVPN-PE neighbor 10.255.0.2
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ldp interface ge-0/0/0.0
set protocols ldp interface lo0.0
set routing-instances ALPHA instance-type virtual-switch

```

```

set routing-instances ALPHA route-distinguisher 10.255.0.3:100
set routing-instances ALPHA vrf-target target:100:100
set routing-instances ALPHA protocols evpn extended-vlan-list 100
set routing-instances ALPHA bridge-domains ONE domain-type bridge
set routing-instances ALPHA bridge-domains ONE vlan-id 100
set routing-instances ALPHA bridge-domains ONE interface ae0.0
set routing-instances ALPHA bridge-domains ONE routing-interface irb.0
set routing-instances BETA instance-type evpn
set routing-instances BETA vlan-id 300
set routing-instances BETA interface ae1.0
set routing-instances BETA routing-interface irb.1
set routing-instances BETA route-distinguisher 10.255.0.3:300
set routing-instances BETA vrf-target target:300:300
set routing-instances DELTA instance-type vrf
set routing-instances DELTA interface irb.0
set routing-instances DELTA interface irb.1
set routing-instances DELTA route-distinguisher 10.255.0.3:200
set routing-instances DELTA vrf-target target:200:200
set routing-instances DELTA vrf-table-label

```

P

```

set interfaces ge-0/0/0 unit 0 family inet address 198.51.100.2/24
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 family inet address 198.51.100.4/24
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/2 unit 0 family inet address 198.51.100.6/24
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.0.4/32 primary
set interfaces lo0 unit 0 family inet address 10.255.0.4/32 preferred
set routing-options router-id 10.255.0.4
set protocols mpls interface lo0.0
set protocols mpls interface ge-0/0/0.0
set protocols mpls interface ge-0/0/1.0
set protocols mpls interface ge-0/0/2.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ldp interface ge-0/0/0.0
set protocols ldp interface ge-0/0/1.0

```



```
set protocols ldp interface ge-0/0/2.0
set protocols ldp interface lo0.0
```

Procedure

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure Router PE1:

NOTE: Repeat this procedure for Router PE2 after modifying the appropriate interface names, addresses, and other parameters.

1. Configure Router PE1 interfaces.

```
[edit interfaces]
user@PE1# set ge-0/0/0 unit 0 family inet address 198.51.100.1/24
user@PE1# set ge-0/0/0 unit 0 family mpls
user@PE1# set ge-0/0/1 gigether-options 802.3ad ae0
user@PE1# set ge-0/0/2 vlan-tagging
user@PE1# set ge-0/0/2 encapsulation flexible-ethernet-services
user@PE1# set ge-0/0/2 unit 0 encapsulation vlan-bridge
user@PE1# set ge-0/0/2 unit 0 vlan-id 100
user@PE1# set ge-0/0/3 gigether-options 802.3ad ae1
user@PE1# set ge-0/0/4 vlan-tagging
user@PE1# set ge-0/0/4 encapsulation flexible-ethernet-services
user@PE1# set ge-0/0/4 esi 00:22:44:66:88:00:22:44:66:88
user@PE1# set ge-0/0/4 esi single-active
user@PE1# set ge-0/0/4 unit 0 encapsulation vlan-bridge
user@PE1# set ge-0/0/4 unit 0 vlan-id 300
user@PE1# set ae0 vlan-tagging
user@PE1# set ae0 encapsulation flexible-ethernet-services
user@PE1# set ae0 esi 00:11:22:33:44:55:66:77:88:99
user@PE1# set ae0 esi single-active
user@PE1# set ae0 unit 0 encapsulation vlan-bridge
user@PE1# set ae0 unit 0 vlan-id 100
user@PE1# set ae1 vlan-tagging
user@PE1# set ae1 encapsulation flexible-ethernet-services
```

```

user@PE1# set ae1 unit 0 encapsulation vlan-bridge
user@PE1# set ae1 unit 0 vlan-id 300
user@PE1# set irb unit 0 family inet address 10.0.0.250/24
user@PE1# set irb unit 0 family inet address 10.0.0.251/24
user@PE1# set irb unit 0 mac 00:11:22:33:44:55
user@PE1# set irb unit 1 family inet address 192.0.2.10/24
user@PE1# set irb unit 1 family inet address 192.0.2.15/24
user@PE1# set irb unit 1 mac 00:22:44:66:88:00
user@PE1# set lo0 unit 0 family inet address 10.255.0.1/32 primary
user@PE1# set lo0 unit 0 family inet address 10.255.0.1/32 preferred

```

2. Configure the loopback address of Router PE1 as the router ID.

```

[edit routing-options]
user@PE1# set router-id 10.255.0.1

```

3. Set the autonomous system number for Router PE1.

```

[edit routing-options]
user@PE1# set autonomous-system 100

```

4. Enable chained composite next hop for the EVPN.

```

[edit routing-options]
user@PE1# set forwarding-table chained-composite-next-hop ingress evpn

```

5. Enable MPLS on the loopback interface of Router PE1 and the interface connecting PE1 to Router P.

```

[edit protocols]
user@PE1# set mpls interface lo0.0
user@PE1# set mpls interface ge-0/0/0.0

```

6. Configure the BGP group for Router PE1.

```

[edit protocols]
user@PE1# set bgp group EVPN-PE type internal

```

7. Assign local and neighbor addresses to the EVPN-PE BGP group for Router PE1 to peer with Routers PE2 and PE3.

```
[edit protocols]
user@PE1# set bgp group EVPN-PE local-address 10.255.0.1
user@PE1# set bgp group EVPN-PE neighbor 10.255.0.2
user@PE1# set bgp group EVPN-PE neighbor 10.255.0.3
```

8. Include the EVPN signaling Network Layer Reachability Information (NLRI) to the EVPN-PE group.

```
[edit protocols]
user@PE1# set bgp group EVPN-PE family evpn signaling
```

9. Configure OSPF on the loopback interface of Router PE1 and the interface connecting PE1 to Router P.

```
[edit protocols]
user@PE1# set ospf area 0.0.0.0 interface lo0.0 passive
user@PE1# set ospf area 0.0.0.0 interface ge-0/0/0.0
```

10. Enable LDP on the loopback interface of Router PE1 and the interface connecting PE1 to Router P.

```
[edit protocols]
user@PE1# set ldp interface lo0.0
user@PE1# set ldp interface ge-0/0/0.0
```

11. Configure the virtual switch routing instance – ALPHA.

```
[edit routing-instances]
user@PE1# set ALPHA instance-type virtual-switch
```

12. Configure the extended VLAN list for the ALPHA routing instance.

```
[edit routing-instances]
user@PE1# set ALPHA protocols evpn extended-vlan-list 100
```

13. Set the type for the bridging domain in the ALPHA routing instance.

```
[edit routing-instances]
user@PE1# set ALPHA bridge-domains ONE domain-type bridge
```

14. Set the VLAN for the bridging domain in the ALPHA routing instance.

```
[edit routing-instances]
user@PE1# set ALPHA bridge-domains ONE vlan-id 100
```

15. Configure the interface names for the ALPHA routing instance.

```
[edit routing-instances]
user@PE1# set ALPHA bridge-domains ONE interface ae0.0
user@PE1# set ALPHA bridge-domains ONE interface ge-0/0/2.0
user@PE1# set ALPHA bridge-domains ONE routing-interface irb.0
```

16. Configure the route distinguisher for the ALPHA routing instance.

```
[edit routing-instances]
user@PE1# set ALPHA route-distinguisher 10.255.0.1:100
```

17. Configure the VPN routing and forwarding (VRF) target community for the ALPHA routing instance.

```
[edit routing-instances]
user@PE1# set ALPHA vrf-target target:100:100
```

18. Configure the EVPN routing instance – BETA.

```
[edit routing-instances]
user@PE1# set BETA instance-type evpn
```

19. Set the VLAN identifier for the bridging domain in the BETA routing instance.

```
[edit routing-instances]
user@PE1# set BETA vlan-id 300
```

20. Configure the interface names for the BETA routing instance.

```
[edit routing-instances]
user@PE1# set BETA interface ge-0/0/4.0
user@PE1# set BETA interface ae1.0
user@PE1# set BETA routing-interface irb.1
```

21. Configure the route distinguisher for the BETA routing instance.

```
[edit routing-instances]
user@PE1# set BETA route-distinguisher 10.255.0.1:300
```

22. Configure the VPN routing and forwarding (VRF) target community for the BETA routing instance.

```
[edit routing-instances]
user@PE1# set BETA vrf-target target:300:300
```

23. Configure the VRF routing instance – DELTA.

```
[edit routing-instances]
user@PE1# set DELTA instance-type vrf
```

24. Configure the interface names for the DELTA routing instance.

```
[edit routing-instances]
user@PE1# set DELTA interface irb.0
user@PE1# set DELTA interface irb.1
```

25. Configure the route distinguisher for the DELTA routing instance.

```
[edit routing-instances]
user@PE1# set DELTA route-distinguisher 10.255.0.1:200
```

26. Configure the VPN routing and forwarding (VRF) target community for the DELTA routing instance.

```
[edit routing-instances]
user@PE1# set DELTA vrf-target target:200:200
user@PE1# set DELTA vrf-table-label
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show routing-options`, `show protocols`, and `show routing-instances` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show interfaces
ge-0/0/0 {
  unit 0 {
    family inet {
      address 198.51.100.1/24;
    }
    family mpls;
  }
}
ge-0/0/1 {
  gigether-options {
    802.3ad ae0;
  }
}
ge-0/0/2 {
  vlan-tagging;
  encapsulation flexible-ethernet-services;
  unit 0 {
    encapsulation vlan-bridge;
    vlan-id 100;
  }
}
ge-0/0/3 {
```

```

    gigaether-options {
        802.3ad ae1;
    }
}
ge-0/0/4 {
    vlan-tagging;
    encapsulation flexible-ethernet-services;
    esi {
        00:22:44:66:88:00:22:44:66:88;
        single-active;
    }
    unit 0 {
        encapsulation vlan-bridge;
        vlan-id 300;
    }
}
ae0 {
    vlan-tagging;
    encapsulation flexible-ethernet-services;
    esi {
        00:11:22:33:44:55:66:77:88:99;
        single-active;
    }
    unit 0 {
        encapsulation vlan-bridge;
        vlan-id 100;
    }
}
ae1 {
    vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 0 {
        encapsulation vlan-bridge;
        vlan-id 300;
    }
}
irb {
    unit 0 {
        family inet {
            address 10.0.0.250/24;
            address 10.0.0.251/24;
        }
        mac 00:11:22:33:44:55;
    }
}

```

```

    }
    unit 1 {
        family inet {
            address 192.0.2.10/24;
            address 192.0.2.15/24;
        }
        mac 00:22:44:66:88:00;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.255.0.1/32 {
                primary;
                preferred;
            }
        }
    }
}
}

```

```

user@PE1# show routing-options
router-id 10.255.0.1;
autonomous-system 100;
forwarding-table {
    chained-composite-next-hop {
        ingress {
            evpn;
        }
    }
}
}

```

```

user@PE1# show protocols
mpls {
    interface lo0.0;
    interface ge-0/0/0.0;
}
bgp {
    group EVPN-PE {
        type internal;
        local-address 10.255.0.1;
    }
}

```



```

        family evpn {
            signaling;
        }
        neighbor 10.255.0.2;
        neighbor 10.255.0.3;
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.0 {
            passive;
        }
        interface ge-0/0/0.0;
    }
}
ldp {
    interface ge-0/0/0.0;
    interface lo0.0;
}

```

```

user@PE1# show routing-instances
ALPHA {
    instance-type virtual-switch;
    route-distinguisher 10.255.0.1:100;
    vrf-target target:100:100;
    protocols {
        evpn {
            extended-vlan-list 100;
        }
    }
    bridge-domains {
        ONE {
            domain-type bridge;
            vlan-id 100;
            interface ae0.0;
            interface ge-0/0/2.0;
            routing-interface irb.0;
        }
    }
}
BETA {

```

```

instance-type evpn;
vlan-id 300;
interface ge-0/0/4.0;
interface ae1.0;
routing-interface irb.1;
route-distinguisher 10.255.0.1:300;
vrf-target target:300:300;
}
DELTA {
instance-type vrf;
interface irb.0;
interface irb.1;
route-distinguisher 10.255.0.1:200;
vrf-target target:200:200;
vrf-table-label;
}

```

Verification

IN THIS SECTION

- [Verifying the EVPN Instance Status | 176](#)
- [Verifying the Autodiscovery Routes per Ethernet Segment | 182](#)
- [Verifying the Ethernet Segment Route | 189](#)
- [Verifying the DF Status | 191](#)
- [Verifying the BDF Status | 191](#)
- [Verifying Remote IRB MAC | 192](#)
- [Verifying Remote IRB and Host IP | 193](#)
- [Verifying ARP Table | 197](#)
- [Verifying Bridge ARP Table | 198](#)
- [Verifying Bridge MAC Table | 199](#)

Confirm that the configuration is working properly in the following different areas on all the PE routers, where Router PE1 is the designated forwarder (DF), Router PE2 is the non-DF, and Router PE3 is the remote PE:

1. EVPN routing instance configuration

2. EVPN multihoming routes
3. DF election process
4. IRB and virtual switch routing instance configuration
5. Host route entries

Verifying the EVPN Instance Status

Purpose

Verify the EVPN routing instances and their status.

Action

Router PE1

From operational mode, run the **show evpn instance extensive** command.

```
user@PE1> show evpn instance extensive
Instance: ALPHA
Route Distinguisher: 10.255.0.1:100
Per-instance MAC route label: 300144
Per-instance multicast route label: 300160
MAC database status          Local  Remote
Total MAC addresses:         3       4
Default gateway MAC addresses: 1       2
Number of local interfaces: 2 (2 up)
Interface name  ESI                      Mode          SH label
ae0.0          00:11:22:33:44:55:66:77:88:99  single-active
ge-0/0/2.0     00:00:00:00:00:00:00:00:00:00  single-homed
Number of IRB interfaces: 1 (1 up)
Interface name  L3 context
irb.0          DELTA
Number of neighbors: 2
10.255.0.2
Received routes
MAC address advertisement:         2
MAC+IP address advertisement:      3
Inclusive multicast:               1
Ethernet auto-discovery:           1
10.255.0.3
```

```

Received routes
  MAC address advertisement:      2
  MAC+IP address advertisement:   2
  Inclusive multicast:            1
  Ethernet auto-discovery:       0
Number of ethernet segments: 1
ESI: 00:11:22:33:44:55:66:77:88:99
  Designated forwarder: 10.255.0.1
  Backup forwarder: 10.255.0.2

Instance: BETA
Route Distinguisher: 10.255.0.1:300
VLAN ID: 300
Per-instance MAC route label: 300176
Per-instance multicast route label: 300192
MAC database status          Local Remote
  Total MAC addresses:        3      4
  Default gateway MAC addresses: 1      2
Number of local interfaces: 2 (2 up)
  Interface name  ESI                      Mode          SH label
  ae1.0          00:00:00:00:00:00:00:00:00 single-homed
  ge-0/0/4.0     00:22:44:66:88:00:22:44:66:88 single-active
Number of IRB interfaces: 1 (1 up)
  Interface name  L3 context
  irb.1          DELTA
Number of neighbors: 2
  10.255.0.2
    Received routes
      MAC address advertisement:      2
      MAC+IP address advertisement:   3
      Inclusive multicast:            1
      Ethernet auto-discovery:       1
  10.255.0.3
    Received routes
      MAC address advertisement:      2
      MAC+IP address advertisement:   2
      Inclusive multicast:            1
      Ethernet auto-discovery:       0
Number of ethernet segments: 1
ESI: 00:22:44:66:88:00:22:44:66:88
  Designated forwarder: 10.255.0.1
  Backup forwarder: 10.255.0.2

```

```

Instance: __default_evpn__
Route Distinguisher: 10.255.0.1:0
VLAN ID: 0
Per-instance MAC route label: 300208
Per-instance multicast route label: 300224
MAC database status          Local  Remote
Total MAC addresses:         0      0
Default gateway MAC addresses: 0      0
Number of local interfaces: 0 (0 up)
Number of IRB interfaces: 0 (0 up)
Number of neighbors: 1
10.255.0.2
Received routes
Ethernet auto-discovery:      0
Ethernet Segment:            2
Number of ethernet segments: 0

```

Router PE2

From operational mode, run the **show evpn instance extensive** command.

```

user@PE2> show evpn instance extensive
Instance: ALPHA
Route Distinguisher: 10.255.0.2:100
Per-instance MAC route label: 300208
Per-instance multicast route label: 300224
MAC database status          Local  Remote
Total MAC addresses:         2      5
Default gateway MAC addresses: 1      2
Number of local interfaces: 2 (2 up)
Interface name  ESI                      Mode          SH label
ae0.0           00:11:22:33:44:55:66:77:88:99  single-active
ge-0/0/2.0      00:00:00:00:00:00:00:00:00:00  single-homed
Number of IRB interfaces: 1 (1 up)
Interface name  L3 context
irb.0           DELTA
Number of neighbors: 2
10.255.0.1
Received routes
MAC address advertisement:      3
MAC+IP address advertisement:  4
Inclusive multicast:            1

```

```

    Ethernet auto-discovery:          1
10.255.0.3
    Received routes
      MAC address advertisement:      2
      MAC+IP address advertisement:   2
      Inclusive multicast:            1
      Ethernet auto-discovery:        0
Number of ethernet segments: 1
ESI: 00:11:22:33:44:55:66:77:88:99
  Designated forwarder: 10.255.0.1
  Backup forwarder: 10.255.0.2

Instance: BETA
Route Distinguisher: 10.255.0.2:300
VLAN ID: 300
Per-instance MAC route label: 300240
Per-instance multicast route label: 300256
MAC database status          Local  Remote
  Total MAC addresses:        2      5
  Default gateway MAC addresses: 1      2
Number of local interfaces: 2 (2 up)
  Interface name  ESI                      Mode          SH label
  ae1.0           00:00:00:00:00:00:00:00:00 single-homed
  ge-0/0/4.0      00:22:44:66:88:00:22:44:66:88 single-active
Number of IRB interfaces: 1 (1 up)
  Interface name  L3 context
  irb.1           DELTA
Number of neighbors: 2
10.255.0.1
  Received routes
    MAC address advertisement:      3
    MAC+IP address advertisement:   4
    Inclusive multicast:            1
    Ethernet auto-discovery:        1
10.255.0.3
  Received routes
    MAC address advertisement:      2
    MAC+IP address advertisement:   2
    Inclusive multicast:            1
    Ethernet auto-discovery:        0
Number of ethernet segments: 1
ESI: 00:22:44:66:88:00:22:44:66:88
  Designated forwarder: 10.255.0.1

```

Backup forwarder: 10.255.0.2

Instance: __default_evpn__

Route Distinguisher: 10.255.0.2:0

VLAN ID: 0

Per-instance MAC route label: 300272

Per-instance multicast route label: 300288

MAC database status	Local	Remote
---------------------	-------	--------

Total MAC addresses:	0	0
----------------------	---	---

Default gateway MAC addresses:	0	0
--------------------------------	---	---

Number of local interfaces: 0 (0 up)

Number of IRB interfaces: 0 (0 up)

Number of neighbors: 1

10.255.0.1

Received routes

Ethernet auto-discovery:	0
--------------------------	---

Ethernet Segment:	2
-------------------	---

Number of ethernet segments: 0

Router PE3

From operational mode, run the **show evpn instance extensive** command.

```
user@PE3> show evpn instance extensive
```

Instance: ALPHA

Route Distinguisher: 10.255.0.3:100

Per-instance MAC route label: 299776

Per-instance multicast route label: 299792

MAC database status	Local	Remote
---------------------	-------	--------

Total MAC addresses:	2	4
----------------------	---	---

Default gateway MAC addresses:	1	1
--------------------------------	---	---

Number of local interfaces: 1 (1 up)

Interface name	ESI	Mode	SH label
ae0.0	00:00:00:00:00:00:00:00:00:00	single-homed	

Number of IRB interfaces: 1 (1 up)

Interface name	L3 context
----------------	------------

irb.0	DELTA
-------	-------

Number of neighbors: 2

10.255.0.1

Received routes

MAC address advertisement:	3
----------------------------	---

MAC+IP address advertisement:	4
-------------------------------	---

```

    Inclusive multicast:          1
    Ethernet auto-discovery:      1
10.255.0.2
    Received routes
    MAC address advertisement:     2
    MAC+IP address advertisement:  3
    Inclusive multicast:          1
    Ethernet auto-discovery:      1
Number of ethernet segments: 0

```

Instance: BETA

Route Distinguisher: 10.255.0.3:300

VLAN ID: 300

Per-instance MAC route label: 299808

Per-instance multicast route label: 299824

MAC database status	Local	Remote
---------------------	-------	--------

Total MAC addresses:	2	4
----------------------	---	---

Default gateway MAC addresses:	1	1
--------------------------------	---	---

Number of local interfaces: 1 (1 up)

Interface name	ESI	Mode	SH label
ae1.0	00:00:00:00:00:00:00:00:00:00	single-homed	

Number of IRB interfaces: 1 (1 up)

Interface name	L3 context
----------------	------------

irb.1	DELTA
-------	-------

Number of neighbors: 2

10.255.0.1

Received routes

MAC address advertisement:	3
----------------------------	---

MAC+IP address advertisement:	4
-------------------------------	---

Inclusive multicast:	1
----------------------	---

Ethernet auto-discovery:	1
--------------------------	---

10.255.0.2

Received routes

MAC address advertisement:	2
----------------------------	---

MAC+IP address advertisement:	3
-------------------------------	---

Inclusive multicast:	1
----------------------	---

Ethernet auto-discovery:	1
--------------------------	---

Number of ethernet segments: 0

Instance: __default_evpn__

Route Distinguisher: 10.255.0.3:0

VLAN ID: 0

Per-instance MAC route label: 299840


```

Per-instance multicast route label: 299856
MAC database status          Local  Remote
Total MAC addresses:         0      0
Default gateway MAC addresses: 0      0
Number of local interfaces: 0 (0 up)
Number of IRB interfaces: 0 (0 up)
Number of neighbors: 0
Number of ethernet segments: 0

```

Meaning

The output provides the following information:

- List of EVPN and virtual switch routing instances.
- Mode of operation of each interface
- Neighbors of each routing instance.
- Number of different routes received from each neighbor.
- ESI attached to each routing instance.
- Number of Ethernet segments on each routing instance.
- DF election roles for each ESI in an EVI.
- VLAN ID and MAC labels for each routing instance.
- IRB interface details
- Number of default gateway MAC addresses received for the virtual switch routing instance (ALPHA).

Verifying the Autodiscovery Routes per Ethernet Segment

Purpose

Verify that the autodiscovery routes per Ethernet segment are received.

Action

Router PE1

From operational mode, run the **show route table ALPHA.evpn.0** command.

```

user@PE1> show route table ALPHA.evpn.0
ALPHA.evpn.0: 20 destinations, 20 routes (20 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:10.255.0.2:0::112233445566778899::0/304
    *[BGP/170] 2d 23:51:27, localpref 100, from 10.255.0.2
    AS path: I, validation-state: unverified
    > to 198.51.100.2 via ge-0/0/0.0, Push 299808
2:10.255.0.1:100::100::00:00:00:00:00:01/304
    *[EVPN/170] 2d 23:52:22
    Indirect
2:10.255.0.1:100::100::00:00:00:00:00:04/304
    *[EVPN/170] 2d 23:52:24
    Indirect
2:10.255.0.1:100::100::00:11:22:33:44:55/304
    *[EVPN/170] 2d 23:53:32
    Indirect
2:10.255.0.2:100::100::00:00:00:00:00:02/304
    *[BGP/170] 2d 23:51:27, localpref 100, from 10.255.0.2
    AS path: I, validation-state: unverified
    > to 198.51.100.2 via ge-0/0/0.0, Push 299808
2:10.255.0.2:100::100::00:11:22:33:44:55/304
    *[BGP/170] 2d 23:51:27, localpref 100, from 10.255.0.2
    AS path: I, validation-state: unverified
    > to 198.51.100.2 via ge-0/0/0.0, Push 299808
2:10.255.0.3:100::100::00:00:00:00:00:03/304
    *[BGP/170] 2d 23:39:04, localpref 100, from 10.255.0.3
    AS path: I, validation-state: unverified
    > to 198.51.100.2 via ge-0/0/0.0, Push 299824
2:10.255.0.3:100::100::00:05:86:71:b3:f0/304
    *[BGP/170] 2d 23:39:24, localpref 100, from 10.255.0.3
    AS path: I, validation-state: unverified
    > to 198.51.100.2 via ge-0/0/0.0, Push 299824
2:10.255.0.1:100::100::00:00:00:00:00:01::10.0.0.1/304
    *[EVPN/170] 2d 23:52:22
    Indirect
2:10.255.0.1:100::100::00:00:00:00:00:04::10.0.0.4/304
    *[EVPN/170] 2d 23:52:24
    Indirect
2:10.255.0.1:100::100::00:11:22:33:44:55::10.0.0.250/304

```

```

*[EVPN/170] 2d 23:53:32
    Indirect
2:10.255.0.1:100::100::00:11:22:33:44:55::10.0.0.251/304
    *[EVPN/170] 2d 23:53:32
        Indirect
2:10.255.0.2:100::100::00:00:00:00:00:02::10.0.0.2/304
    *[BGP/170] 2d 23:51:27, localpref 100, from 10.255.0.2
        AS path: I, validation-state: unverified
        > to 198.51.100.2 via ge-0/0/0.0, Push 299808
2:10.255.0.2:100::100::00:11:22:33:44:55::10.0.0.250/304
    *[BGP/170] 2d 23:51:27, localpref 100, from 10.255.0.2
        AS path: I, validation-state: unverified
        > to 198.51.100.2 via ge-0/0/0.0, Push 299808
2:10.255.0.2:100::100::00:11:22:33:44:55::10.0.0.252/304
    *[BGP/170] 2d 23:51:27, localpref 100, from 10.255.0.2
        AS path: I, validation-state: unverified
        > to 198.51.100.2 via ge-0/0/0.0, Push 299808
2:10.255.0.3:100::100::00:00:00:00:00:03::10.0.0.3/304
    *[BGP/170] 2d 23:39:04, localpref 100, from 10.255.0.3
        AS path: I, validation-state: unverified
        > to 198.51.100.2 via ge-0/0/0.0, Push 299824
2:10.255.0.3:100::100::00:05:86:71:b3:f0::10.0.0.253/304
    *[BGP/170] 2d 23:39:24, localpref 100, from 10.255.0.3
        AS path: I, validation-state: unverified
        > to 198.51.100.2 via ge-0/0/0.0, Push 299824
3:10.255.0.1:100::100::10.255.0.1/304
    *[EVPN/170] 2d 23:52:33
        Indirect
3:10.255.0.2:100::100::10.255.0.2/304
    *[BGP/170] 2d 23:51:27, localpref 100, from 10.255.0.2
        AS path: I, validation-state: unverified
        > to 198.51.100.2 via ge-0/0/0.0, Push 299808
3:10.255.0.3:100::100::10.255.0.3/304
    *[BGP/170] 2d 23:39:24, localpref 100, from 10.255.0.3
        AS path: I, validation-state: unverified
        > to 198.51.100.2 via ge-0/0/0.0, Push 299824

```

Router PE2

From operational mode, run the **show route table ALPHA.evpn.0** command.

```

user@PE2> show show route table ALPHA.evpn.0
ALPHA.evpn.0: 20 destinations, 20 routes (20 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:10.255.0.1:0::112233445566778899::0/304
    *[BGP/170] 10:46:04, localpref 100, from 10.255.0.1
    AS path: I, validation-state: unverified
    > to 198.51.100.4 via ge-0/0/0.0, Push 299776
2:10.255.0.1:100::100::00:00:00:00:00:01/304
    *[BGP/170] 10:43:51, localpref 100, from 10.255.0.1
    AS path: I, validation-state: unverified
    > to 198.51.100.4 via ge-0/0/0.0, Push 299776
2:10.255.0.1:100::100::00:00:00:00:00:04/304
    *[BGP/170] 10:45:06, localpref 100, from 10.255.0.1
    AS path: I, validation-state: unverified
    > to 198.51.100.4 via ge-0/0/0.0, Push 299776
2:10.255.0.1:100::100::00:11:22:33:44:55/304
    *[BGP/170] 10:46:04, localpref 100, from 10.255.0.1
    AS path: I, validation-state: unverified
    > to 198.51.100.4 via ge-0/0/0.0, Push 299776
2:10.255.0.2:100::100::00:00:00:00:00:02/304
    *[EVPN/170] 10:43:48
    Indirect
2:10.255.0.2:100::100::00:11:22:33:44:55/304
    *[EVPN/170] 10:46:04
    Indirect
2:10.255.0.3:100::100::00:00:00:00:00:03/304
    *[BGP/170] 10:46:04, localpref 100, from 10.255.0.3
    AS path: I, validation-state: unverified
    > to 198.51.100.4 via ge-0/0/0.0, Push 299792
2:10.255.0.3:100::100::00:05:86:71:79:f0/304
    *[BGP/170] 10:46:04, localpref 100, from 10.255.0.3
    AS path: I, validation-state: unverified
    > to 198.51.100.4 via ge-0/0/0.0, Push 299792
2:10.255.0.1:100::100::00:00:00:00:00:01::10.0.0.1/304
    *[BGP/170] 10:41:52, localpref 100, from 10.255.0.1
    AS path: I, validation-state: unverified
    > to 198.51.100.4 via ge-0/0/0.0, Push 299776
2:10.255.0.1:100::100::00:00:00:00:00:04::10.0.0.4/304
    *[BGP/170] 10:45:06, localpref 100, from 10.255.0.1

```

```

        AS path: I, validation-state: unverified
        > to 198.51.100.4 via ge-0/0/0.0, Push 299776
2:10.255.0.1:100::100::00:11:22:33:44:55::10.0.0.250/304
    *[BGP/170] 10:46:04, localpref 100, from 10.255.0.1
        AS path: I, validation-state: unverified
        > to 198.51.100.4 via ge-0/0/0.0, Push 299776
2:10.255.0.1:100::100::00:11:22:33:44:55::10.0.0.251/304
    *[BGP/170] 10:46:04, localpref 100, from 10.255.0.1
        AS path: I, validation-state: unverified
        > to 198.51.100.4 via ge-0/0/0.0, Push 299776
2:10.255.0.2:100::100::00:00:00:00:00:02::10.0.0.2/304
    *[EVPN/170] 10:40:25
        Indirect
2:10.255.0.2:100::100::00:11:22:33:44:55::10.0.0.250/304
    *[EVPN/170] 10:46:04
        Indirect
2:10.255.0.2:100::100::00:11:22:33:44:55::10.0.0.252/304
    *[EVPN/170] 10:46:04
        Indirect
2:10.255.0.3:100::100::00:00:00:00:00:03::10.0.0.3/304
    *[BGP/170] 10:46:04, localpref 100, from 10.255.0.3
        AS path: I, validation-state: unverified
        > to 198.51.100.4 via ge-0/0/0.0, Push 299792
2:10.255.0.3:100::100::00:05:86:71:79:f0::10.0.0.253/304
    *[BGP/170] 10:46:04, localpref 100, from 10.255.0.3
        AS path: I, validation-state: unverified
        > to 198.51.100.4 via ge-0/0/0.0, Push 299792
3:10.255.0.1:100::100::10.255.0.1/304
    *[BGP/170] 10:46:04, localpref 100, from 10.255.0.1
        AS path: I, validation-state: unverified
        > to 198.51.100.4 via ge-0/0/0.0, Push 299776
3:10.255.0.2:100::100::10.255.0.2/304
    *[EVPN/170] 10:46:04
        Indirect
3:10.255.0.3:100::100::10.255.0.3/304
    *[BGP/170] 10:46:04, localpref 100, from 10.255.0.3
        AS path: I, validation-state: unverified
        > to 198.51.100.4 via ge-0/0/0.0, Push 299792

```

Router PE3

From operational mode, run the **show route table ALPHA.evpn.0** command.

```

user@PE3> show route table ALPHA.evpn.0
ALPHA.evpn.0: 21 destinations, 21 routes (21 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:10.255.0.1:0::112233445566778899::0/304
    *[BGP/170] 10:47:43, localpref 100, from 10.255.0.1
    AS path: I, validation-state: unverified
    > to 198.51.100.6 via ge-0/0/0.0, Push 299776
1:10.255.0.2:0::112233445566778899::0/304
    *[BGP/170] 10:47:34, localpref 100, from 10.255.0.2
    AS path: I, validation-state: unverified
    > to 198.51.100.6 via ge-0/0/0.0, Push 299808
2:10.255.0.1:100::100::00:00:00:00:00:01/304
    *[BGP/170] 10:45:21, localpref 100, from 10.255.0.1
    AS path: I, validation-state: unverified
    > to 198.51.100.6 via ge-0/0/0.0, Push 299776
2:10.255.0.1:100::100::00:00:00:00:00:04/304
    *[BGP/170] 10:46:36, localpref 100, from 10.255.0.1
    AS path: I, validation-state: unverified
    > to 198.51.100.6 via ge-0/0/0.0, Push 299776
2:10.255.0.1:100::100::00:11:22:33:44:55/304
    *[BGP/170] 10:47:43, localpref 100, from 10.255.0.1
    AS path: I, validation-state: unverified
    > to 198.51.100.6 via ge-0/0/0.0, Push 299776
2:10.255.0.2:100::100::00:00:00:00:00:02/304
    *[BGP/170] 10:45:18, localpref 100, from 10.255.0.2
    AS path: I, validation-state: unverified
    > to 198.51.100.6 via ge-0/0/0.0, Push 299808
2:10.255.0.2:100::100::00:11:22:33:44:55/304
    *[BGP/170] 10:47:34, localpref 100, from 10.255.0.2
    AS path: I, validation-state: unverified
    > to 198.51.100.6 via ge-0/0/0.0, Push 299808
2:10.255.0.3:100::100::00:00:00:00:00:03/304
    *[EVPN/170] 10:59:05
    Indirect
2:10.255.0.3:100::100::00:05:86:71:79:f0/304
    *[EVPN/170] 11:00:23
    Indirect
2:10.255.0.1:100::100::00:00:00:00:00:01::10.0.0.1/304
    *[BGP/170] 10:43:22, localpref 100, from 10.255.0.1

```

```

        AS path: I, validation-state: unverified
        > to 198.51.100.6 via ge-0/0/0.0, Push 299776
2:10.255.0.1:100::100::00:00:00:00:04::10.0.0.4/304
    *[BGP/170] 10:46:36, localpref 100, from 10.255.0.1
        AS path: I, validation-state: unverified
        > to 198.51.100.6 via ge-0/0/0.0, Push 299776
2:10.255.0.1:100::100::00:11:22:33:44:55::10.0.0.250/304
    *[BGP/170] 10:47:43, localpref 100, from 10.255.0.1
        AS path: I, validation-state: unverified
        > to 198.51.100.6 via ge-0/0/0.0, Push 299776
2:10.255.0.1:100::100::00:11:22:33:44:55::10.0.0.251/304
    *[BGP/170] 10:47:43, localpref 100, from 10.255.0.1
        AS path: I, validation-state: unverified
        > to 198.51.100.6 via ge-0/0/0.0, Push 299776
2:10.255.0.2:100::100::00:00:00:00:02::10.0.0.2/304
    *[BGP/170] 10:41:55, localpref 100, from 10.255.0.2
        AS path: I, validation-state: unverified
        > to 198.51.100.6 via ge-0/0/0.0, Push 299808
2:10.255.0.2:100::100::00:11:22:33:44:55::10.0.0.250/304
    *[BGP/170] 10:47:34, localpref 100, from 10.255.0.2
        AS path: I, validation-state: unverified
        > to 198.51.100.6 via ge-0/0/0.0, Push 299808
2:10.255.0.2:100::100::00:11:22:33:44:55::10.0.0.252/304
    *[BGP/170] 10:47:34, localpref 100, from 10.255.0.2
        AS path: I, validation-state: unverified
        > to 198.51.100.6 via ge-0/0/0.0, Push 299808
2:10.255.0.3:100::100::00:00:00:00:03::10.0.0.3/304
    *[EVPN/170] 10:59:05
        Indirect
2:10.255.0.3:100::100::00:05:86:71:79:f0::10.0.0.253/304
    *[EVPN/170] 11:00:23
        Indirect
3:10.255.0.1:100::100::10.255.0.1/304
    *[BGP/170] 10:47:43, localpref 100, from 10.255.0.1
        AS path: I, validation-state: unverified
        > to 198.51.100.6 via ge-0/0/0.0, Push 299776
3:10.255.0.2:100::100::10.255.0.2/304
    *[BGP/170] 10:47:34, localpref 100, from 10.255.0.2
        AS path: I, validation-state: unverified
        > to 198.51.100.6 via ge-0/0/0.0, Push 299808
3:10.255.0.3:100::100::10.255.0.3/304

```

```
*[EVPN/170] 10:59:40
Indirect
```

Meaning

The remote type 1 autodiscovery route is received for the ESI attached to Router PE2, which is the other PE router connected to the multihomed CE device.

Verifying the Ethernet Segment Route

Purpose

Verify that the local and advertised autodiscovery routes per Ethernet segment and the Ethernet segment routes are received.

Action

Router PE1

From operational mode, run the **show route table __default_evpn__.evpn.0** command.

```
user@PE1> show route table __default_evpn__.evpn.0
__default_evpn__.evpn.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:10.255.0.1:0::112233445566778899::0/304
    *[EVPN/170] 3d 00:00:31
    Indirect
1:10.255.0.1:0::224466880022446688::0/304
    *[EVPN/170] 3d 00:00:31
    Indirect
4:10.255.0.1:0::112233445566778899:10.255.0.1/304
    *[EVPN/170] 3d 00:00:31
    Indirect
4:10.255.0.1:0::224466880022446688:10.255.0.1/304
    *[EVPN/170] 3d 00:00:31
    Indirect
4:10.255.0.2:0::112233445566778899:10.255.0.2/304
    *[BGP/170] 3d 00:00:20, localpref 100, from 10.255.0.2
    AS path: I, validation-state: unverified
    > to 198.51.100.2 via ge-0/0/0.0, Push 299808
```



```

4:10.255.0.2:0::224466880022446688:10.255.0.2/304
    *[BGP/170] 3d 00:00:20, localpref 100, from 10.255.0.2
    AS path: I, validation-state: unverified
    > to 198.51.100.2 via ge-0/0/0.0, Push 299808

```

Router PE2

From operational mode, run the **show route table __default_evpn__.evpn.0** command.

```

user@PE2> show route table __default_evpn__.evpn.0
__default_evpn__.evpn.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:10.255.0.2:0::112233445566778899::0/304
    *[EVPN/170] 10:49:26
    Indirect
1:10.255.0.2:0::224466880022446688::0/304
    *[EVPN/170] 10:49:26
    Indirect
4:10.255.0.1:0::112233445566778899:10.255.0.1/304
    *[BGP/170] 10:49:26, localpref 100, from 10.255.0.1
    AS path: I, validation-state: unverified
    > to 198.51.100.4 via ge-0/0/0.0, Push 299776
4:10.255.0.1:0::224466880022446688:10.255.0.1/304
    *[BGP/170] 10:49:26, localpref 100, from 10.255.0.1
    AS path: I, validation-state: unverified
    > to 198.51.100.4 via ge-0/0/0.0, Push 299776
4:10.255.0.2:0::112233445566778899:10.255.0.2/304
    *[EVPN/170] 10:49:26
    Indirect
4:10.255.0.2:0::224466880022446688:10.255.0.2/304
    *[EVPN/170] 10:49:26
    Indirect

```

Meaning

The output displays the local and remote type 1 (autodiscovery) and type 4 (Ethernet segment) routes:

- 1:10.255.0.1:0::112233445566778899::0/304—Autodiscovery route per Ethernet segment for each local ESI attached to Router PE1 and Router PE2.

- 4:10.255.0.1:0::112233445566778899:10.255.0.1/304—Ethernet segment route for each local ESI attached to Router PE1 and Router PE2.
- 4:10.255.0.2:0::112233445566778899:10.255.0.2/304—Remote Ethernet segment route from Router PE2.

Verifying the DF Status

Purpose

Confirm which PE router is the designated forwarder (DF).

Action

From operational mode, run the **show evpn instance ALPHA esi esi/designated-forwarder** command.

```
user@PE1> show evpn instance ALPHA esi 00:11:22:33:44:55:66:77:88:99
designated-forwarder
Instance: ALPHA
  Number of ethernet segments: 1
    ESI: 00:11:22:33:44:55:66:77:88:99
      Designated forwarder: 10.255.0.1
```

Meaning

Router PE1 is the DF for the ALPHA routing instance.

Verifying the BDF Status

Purpose

Confirm which PE router is the backup designated forwarder (BDF).

Action

From operational mode, run the **show evpn instance ALPHA esi esi/backup-forwarder** command.

```
user@PE1> show evpn instance ALPHA esi 00:11:22:33:44:55:66:77:88:99
backup-forwarder
Instance: ALPHA
  Number of ethernet segments: 1
```

```
ESI: 00:11:22:33:44:55:66:77:88:99
```

```
Backup forwarder: 10.255.0.2
```

Meaning

Router PE2 is the BDF for the ALPHA routing instance.

Verifying Remote IRB MAC

Purpose

Verify that the remote gateway MAC addresses are synchronized among all the PE routers.

Action

Router PE1

From operational mode, run the **show bridge evpn peer-gateway-mac** command.

```
user@PE1> show bridge evpn peer-gateway-mac
Routing instance : ALPHA
Bridging domain : ONE, VLAN : 100
Installed GW MAC addresses:
00:05:86:71:79:f0
```

Router PE2

From operational mode, run the **show bridge evpn peer-gateway-mac** command.

```
user@PE2> show bridge evpn peer-gateway-mac
Routing instance : ALPHA
Bridging domain : ONE, VLAN : 100
Installed GW MAC addresses:
00:05:86:71:79:f0
```

Router PE3

From operational mode, run the **show bridge evpn peer-gateway-mac** command.

```
user@PE2> show bridge evpn peer-gateway-mac
Routing instance : ALPHA
Bridging domain : ONE, VLAN : 100
Installed GW MAC addresses:
00:11:22:33:44:55
```

Meaning

The remote gateway MAC addresses are synchronized:

- Router PE3 gateway MAC is installed in Routers PE1 and PE2 peer-gateway-mac table.
- Routers PE1 and PE2 gateway MAC addresses are installed in Router PE3 peer-gateway-mac table.

Verifying Remote IRB and Host IP

Purpose

Verify that the remote IRB IP and the host IP are received.

Action

Router PE1

From operational mode, run the **show route table DELTA** command.

```
user@PE1> show route table DELTA
DELTA.inet.0: 16 destinations, 18 routes (16 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.0/24      *[Direct/0] 11:25:54
                 > via irb.0
                 [Direct/0] 11:25:54
                 > via irb.0
10.0.0.1/32     *[EVPN/7] 11:21:33
                 > via irb.0
10.0.0.2/32     *[EVPN/7] 11:20:06, metric2 1
                 > to 198.51.100.2 via ge-0/0/0.0, Push 300208, Push 299808(top)
10.0.0.3/32     *[EVPN/7] 11:25:54, metric2 1
```

```

> to 198.51.100.2 via ge-0/0/0.0, Push 299776, Push 299792(top)
10.0.0.4/32      *[EVPN/7] 11:24:47
> via irb.0
10.0.0.250/32   *[Local/0] 11:38:29
                Local via irb.0
10.0.0.251/32   *[Local/0] 11:38:29
                Local via irb.0
10.0.0.253/32   *[EVPN/7] 11:25:54, metric2 1
> to 198.51.100.2 via ge-0/0/0.0, Push 299776, Push 299792(top)
192.0.2.0/24    *[Direct/0] 11:25:55
> via irb.1
                [Direct/0] 11:25:55
> via irb.1
192.0.2.1/24    *[EVPN/7] 11:21:20
> via irb.1
192.0.2.2/24    *[EVPN/7] 11:19:54, metric2 1
> to 198.51.100.2 via ge-0/0/0.0, Push 300240, Push 299808(top)
192.0.2.3/24    *[EVPN/7] 11:25:54, metric2 1
> to 198.51.100.2 via ge-0/0/0.0, Push 299808, Push 299792(top)
192.0.2.4/24    *[EVPN/7] 11:24:40
> via irb.1
192.0.2.10/24   *[Local/0] 11:38:29
                Local via irb.1
192.0.2.15/24   *[Local/0] 11:38:29
                Local via irb.1
192.0.2.13/24   *[EVPN/7] 11:25:54, metric2 1
> to 198.51.100.2 via ge-0/0/0.0, Push 299808, Push 299792(top)

```

Router PE2

From operational mode, run the **show route table DELTA** command.

```

user@PE2> show route table DELTA
DELTA.inet.0: 16 destinations, 18 routes (16 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.0/24      *[Direct/0] 11:30:02
> via irb.0
                [Direct/0] 11:30:02
> via irb.0
10.0.0.1/32      *[EVPN/7] 11:25:50, metric2 1
> to 198.51.100.4 via ge-0/0/0.0, Push 300144, Push 299776(top)

```

```

10.0.0.2/32      *[EVPN/7] 11:24:23
                  > via irb.0
10.0.0.3/32      *[EVPN/7] 11:30:02, metric2 1
                  > to 198.51.100.4 via ge-0/0/0.0, Push 299776, Push 299792(top)
10.0.0.4/32      *[EVPN/7] 11:29:04, metric2 1
                  > to 198.51.100.4 via ge-0/0/0.0, Push 300144, Push 299776(top)
10.0.0.250/32    *[Local/0] 11:42:33
                  Local via irb.0
10.0.0.252/32    *[Local/0] 11:42:33
                  Local via irb.0
10.0.0.253/32    *[EVPN/7] 11:30:02, metric2 1
                  > to 198.51.100.4 via ge-0/0/0.0, Push 299776, Push 299792(top)
192.0.2.0/24     *[Direct/0] 11:30:02
                  > via irb.1
                  [Direct/0] 11:30:02
                  > via irb.1
192.0.2.1/24     *[EVPN/7] 11:25:37, metric2 1
                  > to 198.51.100.4 via ge-0/0/0.0, Push 300176, Push 299776(top)
192.0.2.2/24     *[EVPN/7] 11:24:11
                  > via irb.1
192.0.2.3/24     *[EVPN/7] 11:30:02, metric2 1
                  > to 198.51.100.4 via ge-0/0/0.0, Push 299808, Push 299792(top)
192.0.2.4/24     *[EVPN/7] 11:28:57, metric2 1
                  > to 198.51.100.4 via ge-0/0/0.0, Push 300176, Push 299776(top)
192.0.2.10/24    *[Local/0] 11:42:33
                  Local via irb.1
192.0.2.12/24    *[Local/0] 11:42:33
                  Local via irb.1
192.0.2.13/24    *[EVPN/7] 11:30:02, metric2 1
                  > to 198.51.100.4 via ge-0/0/0.0, Push 299808, Push 299792(top)

```

Router PE3

From operational mode, run the **show route table DELTA** command.

```

user@PE3> show route table DELTA
DELTA.inet.0: 16 destinations, 16 routes (16 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.0/24      *[Direct/0] 11:42:15
                  > via irb.0
10.0.0.1/32      *[EVPN/7] 11:25:56, metric2 1

```

```

10.0.0.2/32      > to 198.51.100.6 via ge-0/0/0.0, Push 300144, Push 299776(top)
                 *[EVPN/7] 11:24:29, metric2 1
                 > to 198.51.100.6 via ge-0/0/0.0, Push 300208, Push 299808(top)
10.0.0.3/32      *[EVPN/7] 11:41:39
                 > via irb.0
10.0.0.4/32      *[EVPN/7] 11:29:10, metric2 1
                 > to 198.51.100.6 via ge-0/0/0.0, Push 300144, Push 299776(top)
10.0.0.250/32    *[EVPN/7] 11:30:08, metric2 1
                 > to 198.51.100.6 via ge-0/0/0.0, Push 300208, Push 299808(top)
10.0.0.252/32    *[EVPN/7] 11:30:08, metric2 1
                 > to 198.51.100.6 via ge-0/0/0.0, Push 300208, Push 299808(top)
10.0.0.253/32    *[Local/0] 11:42:57
                 Local via irb.0
192.0.2.0/24     *[Direct/0] 11:42:15
                 > via irb.1
192.0.2.1/24     *[EVPN/7] 11:25:43, metric2 1
                 > to 198.51.100.6 via ge-0/0/0.0, Push 300176, Push 299776(top)
192.0.2.2/24     *[EVPN/7] 11:24:17, metric2 1
                 > to 198.51.100.6 via ge-0/0/0.0, Push 300240, Push 299808(top)
192.0.2.3/24     *[EVPN/7] 11:42:04
                 > via irb.1
192.0.2.4/24     *[EVPN/7] 11:29:03, metric2 1
                 > to 198.51.100.6 via ge-0/0/0.0, Push 300176, Push 299776(top)
192.0.2.10/24    *[EVPN/7] 11:30:08, metric2 1
                 > to 198.51.100.6 via ge-0/0/0.0, Push 300240, Push 299808(top)
192.0.2.12/24    *[EVPN/7] 11:30:08, metric2 1
                 > to 198.51.100.6 via ge-0/0/0.0, Push 300240, Push 299808(top)
192.0.2.13/24    *[Local/0] 11:42:57
                 Local via irb.1

```

Meaning

The output displays the local and remote IRB interfaces. It also displays the local and remote hosts that are installed in the VRF table:

On Router PE1:

- 10.0.0.1/32—Local host in the virtual switch routing instance.
- 10.0.0.2/32 and 10.0.0.3/32—Remote host in the virtual switch routing instance.
- 10.0.0.250/32—Local IRB in the virtual switch routing instance.
- 10.0.0.253/32—Remote IRB in the virtual switch routing instance.

Verifying ARP Table

Purpose

Verify the ARP table entries.

Action

Router PE1

From operational mode, run the **show evpn arp-table** command.

```
user@PE1> show evpn arp-table
```

INET	MAC	Logical	Routing	Bridging
address	address	interface	instance	domain
192.0.2.1	00:00:00:00:00:01	irb.1	BETA	__BETA__
192.0.2.4	00:00:00:00:00:04	irb.1	BETA	__BETA__

Router PE2

From operational mode, run the **show evpn arp-table** command.

```
user@PE2> show evpn arp-table
```

INET	MAC	Logical	Routing	Bridging
address	address	interface	instance	domain
192.0.2.2	00:00:00:00:00:02	irb.1	BETA	__BETA__

Router PE3

From operational mode, run the **show evpn arp-table** command.

```
user@PE3> show evpn arp-table
```

INET	MAC	Logical	Routing	Bridging
address	address	interface	instance	domain
192.0.2.3	00:00:00:00:00:03	irb.1	BETA	__BETA__

Meaning

The EVPN instance and ARP are synchronized with the host MAC and IP address for local hosts.

Verifying Bridge ARP Table

Purpose

Verify the bridge ARP table entries.

Action

Router PE1

From operational mode, run the **show bridge evpn arp-table** command.

```
user@PE3> show bridge evpn arp-table
```

INET	MAC	Logical	Routing	Bridging
address	address	interface	instance	domain
10.0.0.1	00:00:00:00:00:01	irb.0	ALPHA	ONE
10.0.0.4	00:00:00:00:00:04	irb.0	ALPHA	ONE

Router PE2

From operational mode, run the **show bridge evpn arp-table** command.

```
user@PE3> show bridge evpn arp-table
```

INET	MAC	Logical	Routing	Bridging
address	address	interface	instance	domain
10.0.0.2	00:00:00:00:00:02	irb.0	ALPHA	ONE

Router PE3

From operational mode, run the **show bridge evpn arp-table** command.

```
user@PE3> show bridge evpn arp-table
```

INET	MAC	Logical	Routing	Bridging
address	address	interface	instance	domain
10.0.0.3	00:00:00:00:00:03	irb.0	ALPHA	ONE

Meaning

The virtual switch instance and ARP are synchronized with the local host MAC and IP address.

Verifying Bridge MAC Table

Purpose

Verify the bridge MAC table entries.

Action

Router PE1

From operational mode, run the **show bridge mac-table** command.

```
user@PE1> show bridge mac-table
MAC flags (S -static MAC, D -dynamic MAC, L -locally learned, C -Control MAC
          SE -Statistics enabled, NM -Non configured MAC, R -Remote PE MAC)

Routing instance : ALPHA
Bridging domain : ONE, VLAN : 100
```

MAC address	MAC flags	Logical interface	NH Index	RTR ID
00:00:00:00:00:01	D	ge-0/0/2.0		
00:00:00:00:00:02	DC		1048583	1048583
00:00:00:00:00:03	DC		1048574	1048574
00:00:00:00:00:04	D	ae0.0		

Router PE2

From operational mode, run the **show bridge mac-table** command.

```
user@PE2> show bridge mac-table
MAC flags (S -static MAC, D -dynamic MAC, L -locally learned, C -Control MAC
          SE -Statistics enabled, NM -Non configured MAC, R -Remote PE MAC)

Routing instance : ALPHA
Bridging domain : ONE, VLAN : 100
```

MAC address	MAC flags	Logical interface	NH Index	RTR ID
00:00:00:00:00:01	DC		1048577	1048577
00:00:00:00:00:02	D	ge-0/0/2.0		
00:00:00:00:00:03	DC		1048578	1048578
00:00:00:00:00:04	DC		1048577	1048577

Router PE3

From operational mode, run the **show bridge mac-table** command.

```
user@PE3> show bridge mac-table
MAC flags (S -static MAC, D -dynamic MAC, L -locally learned, C -Control MAC
          SE -Statistics enabled, NM -Non configured MAC, R -Remote PE MAC)

Routing instance : ALPHA
Bridging domain : ONE, VLAN : 100
  MAC      MAC      Logical      NH      RTR
address    flags    interface  Index   ID
00:00:00:00:00:01 DC              1048575 1048575
00:00:00:00:00:02 DC              1048582 1048582
00:00:00:00:00:03 D      ae0.0
00:00:00:00:00:04 DC              1048575 1048575
```

Meaning

The virtual switch instance installed local and remote host MAC addresses in the bridge MAC table.

RELATED DOCUMENTATION

| [EVPN Multihoming Overview](#) | 96

Example: Configuring Basic EVPN Active-Active Multihoming

IN THIS SECTION

- [Requirements](#) | 201
- [Overview](#) | 201
- [Configuration](#) | 202

This example shows how to configure an active-active multihomed customer edge (CE) devices and provider edge (PE) devices in an Ethernet VPN (EVPN).

Requirements

This example uses the following hardware and software components:

- Four MX Series 5G Universal Routing Platforms with MPC interfaces only, where:
 - Two devices are configured as provider edge (PE) routers connected to a common multihomed customer site.
 - One device is configured as a remote PE router connected to a single-homed customer site.
 - One device is configured as a provider router.
- Two customer edge (CE) devices, where:
 - Two CE devices that are multihomed.
 - One CE devices that is single-homed .

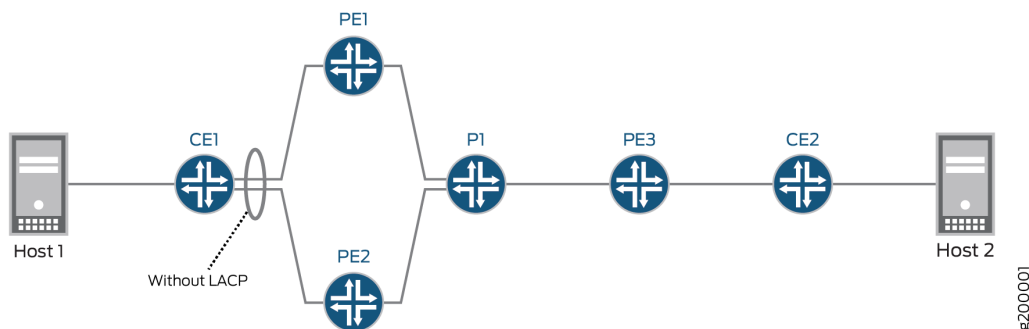
Before you begin:

1. Configure the router interfaces.
2. Configure OSPF or any other IGP protocol.
3. Configure a BGP internal group.
4. Configure MPLS.

Overview

Figure 11 on page 201 illustrates a simple EVPN topology. Routers PE1 and PE2 are provider edge (PE) routers connected to multihomed customer edge (CE) router CE1. Router P is the provider router connected to Routers PE1, PE2, and PE3. Router PE3 is connected to customer edge router CE2.

Figure 11: Simple EVPN Multihomed Topology



Configuration

IN THIS SECTION

- [CLI Quick Configuration | 202](#)

CLI Quick Configuration

The configurations parameters for the routers are as follows:

CE1

```
chassis {
  aggregated-devices {
    ethernet {
      device-count 1;
    }
  }
}
interfaces {
  ge-0/0/1 {
    description To-PE1;
    gigether-options {
      802.3ad ae0;
    }
  }
  ge-0/0/2 {
    description To-PE2;
    gigether-options {
      802.3ad ae0;
    }
  }
  ge-0/0/7 {
    unit 0 {
      family bridge {
        interface-mode access;
        vlan-id 10;
      }
    }
  }
}
```

```

}
ge-0/0/8 {
    unit 0 {
        family bridge {
            interface-mode access;
            vlan-id 20;
        }
    }
}
ae0 {
    description To-PE1_and_PE2;
    unit 0 {
        family bridge {
            interface-mode trunk;
            vlan-id-list [ 10 20 ];
        }
    }
}
}
bridge-domains {
    BD {
        vlan-id-list [ 10 20 ];
    }
}
}

```

PE1

```

interfaces {
    ge-0/0/1 {
        description To-CE1;
        flexible-vlan-tagging;
        encapsulation flexible-ethernet-services;
        esi {
            00:11:11:11:11:11:11:11:11:11;
            all-active;
        }
        unit 10 {
            family bridge {
                interface-mode trunk;
                vlan-id-list [ 10 20 ];
            }
        }
    }
}

```

```

}
ge-0/0/3 {
    unit 0 {
        family inet {
            address 10.3.3.1/30;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.1.1/32;
        }
    }
}
}
routing-options {
    router-id 192.168.1.1;
    autonomous-system 65432;
    forwarding-table {
        export evpn-pplb;
    }
}
protocols {
    rsvp {
        interface ge-0/0/3.0;
    }
    mpls {
        no-cspf;
        label-switched-path PE1-to-PE2 {
            to 192.168.2.2;
        }
        label-switched-path PE1-to-PE3 {
            to 192.168.3.3;
        }
        interface ge-0/0/3.0;
    }
    bgp {
        group EVPN-PE {
            type internal;
            local-address 192.168.1.1;
            family inet-vpn {

```

```

        unicast;
    }
    family evpn {
        signaling;
    }
    neighbor 192.168.3.3;
    neighbor 192.168.2.2;
}
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface all {
            interface-type p2p;
        }
        interface fxp0.0 {
            disable;
        }
    }
}
}
routing-instances {
    evpn-ta {
        instance-type virtual-switch;
        interface ge-0/0/1.10;
        route-distinguisher 65432:10;
        vrf-target target:65432:10;
        protocols {
            evpn {
                extended-vlan-list [ 10 20 ];
            }
        }
        bridge-domains {
            bd10 {
                domain-type bridge;
                vlan-id 10;
            }
            bd20 {
                domain-type bridge;
                vlan-id 20;
            }
        }
    }
}
}

```



```

}
policy-statement evpn-pplb {
    from protocol evpn;
    then {
        load-balance per-packet;
    }
}

```

PE2

```

interfaces {
    ge-0/0/2 {
        description To-CE1;
        flexible-vlan-tagging;
        encapsulation flexible-ethernet-services;
        esi {
            00:11:11:11:11:11:11:11:11:11;
            all-active;
        }
        unit 10 {
            family bridge {
                interface-mode trunk;
                vlan-id-list [ 10 20 ];
            }
        }
    }
    ge-0/0/4 {
        unit 0 {
            family inet {
                address 10.4.4.1/30;
            }
            family mpls;
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 192.168.2.2/32;
            }
        }
    }
}

```

```

routing-options {
    router-id 192.168.2.2;
    autonomous-system 65432;
    forwarding-table {
        export evpn-pplb;
    }
}
protocols {
    rsvp {
        interface ge-0/0/4.0;
    }
    mpls {
        no-cspf;
        label-switched-path PE2-to-PE1 {
            to 192.168.1.1;
        }
        label-switched-path PE2-to-PE3 {
            to 192.168.3.3;
        }
        interface ge-0/0/4.0;
    }
    bgp {
        group EVPN-PE {
            type internal;
            local-address 192.168.2.2;
            family inet-vpn {
                unicast;
            }
            family evpn {
                signaling;
            }
            neighbor 192.168.1.1;
            neighbor 192.168.3.3;
        }
    }
    ospf {
        traffic-engineering;
        area 0.0.0.0 {
            interface all {
                interface-type p2p;
            }
            interface fxp0.0 {
                disable;
            }
        }
    }
}

```

```

    }
  }
}
routing-instances {
  evpn-ta {
    instance-type virtual-switch;
    interface ge-0/0/2.10;
    route-distinguisher 65432:10;
    vrf-target target:65432:10;
    protocols {
      evpn {
        extended-vlan-list [ 10 20 ];
      }
    }
    bridge-domains {
      bd10 {
        domain-type bridge;
        vlan-id 10;
      }
      bd20 {
        domain-type bridge;
        vlan-id 20;
      }
    }
  }
}
policy-statement evpn-pplb {
  from protocol evpn;
  then {
    load-balance per-packet;
  }
}

```

Router PE3

```

interfaces {
  ge-0/0/5 {
    unit 0 {
      family inet {
        address 10.5.5.1/30;
      }
    }
  }
}

```

```

        family mpls;
    }
}
ge-0/0/6 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 10 {
        family bridge {
            interface-mode trunk;
            vlan-id-list 10;
        }
    }
    unit 20 {
        family bridge {
            interface-mode trunk;
            vlan-id-list 20;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.3.3/32;
        }
    }
}
routing-options {
    router-id 192.168.3.3;
    autonomous-system 65432;
    forwarding-table {
        export evpn-pplb;
    }
}
protocols {
    rsvp {
        interface ge-0/0/5.0;
    }
    mpls {
        no-cspf;
        label-switched-path PE3-to-PE1 {
            to 192.168.1.1;
        }
        label-switched-path PE3-to-PE2 {

```

```

        to 192.168.2.2;
    }
    interface ge-0/0/5.0;
}
bgp {
    group EVPN-PE {
        type internal;
        local-address 192.168.3.3;
        family inet-vpn {
            unicast;
        }
        family evpn {
            signaling;
        }
        neighbor 192.168.1.1;
        neighbor 192.168.2.2;
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface all {
            interface-type p2p;
        }
        interface fxp0.0 {
            disable;
        }
    }
}
}
routing-instances {
    evpn-ta {
        instance-type virtual-switch;
        interface ge-0/0/6.10;
        interface ge-0/0/6.20;
        route-distinguisher 65432:10;
        vrf-target target:65432:10;
        protocols {
            evpn {
                extended-vlan-list [ 10 20 ];
            }
        }
    }
    bridge-domains {

```

```

        bd10 {
            domain-type bridge;
            vlan-id 10;
            bridge-options {
                interface ge-0/0/6.10;
            }
        }
        bd20 {
            domain-type bridge;
            vlan-id 20;
            bridge-options {
                interface ge-0/0/6.20;
            }
        }
    }
}

policy-statement evpn-pplb {
    from protocol evpn;
    then {
        load-balance per-packet;
    }
}

```

Router P

```

interfaces {
    ge-0/0/3 {
        unit 0 {
            family inet {
                address 10.3.3.2/30;
            }
            family mpls;
        }
    }
    ge-0/0/4 {
        unit 0 {
            family inet {
                address 10.4.4.2/30;
            }
            family mpls;
        }
    }
}

```

```

    }
}
ge-0/0/5 {
    unit 0 {
        family inet {
            address 10.5.5.2/30;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.4.4/32;
        }
    }
}
}
routing-options {
    router-id 192.168.4.4;
    autonomous-system 65432;
}
protocols {
    rsvp {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
    mpls {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
    ospf {
        traffic-engineering;
        area 0.0.0.0 {
            interface all {
                interface-type p2p;
            }
            interface fxp0.0 {
                disable;
            }
        }
    }
}

```

```

    }
  }
}

```

Example: Configuring EVPN Active-Active Multihoming

IN THIS SECTION

- [Requirements | 213](#)
- [Overview and Topology | 214](#)
- [Configuration | 215](#)
- [Verification | 255](#)

This example shows how to configure Ethernet VPN (EVPN) for multihomed customer edge devices in the active-active redundancy mode, so the Layer 2 unicast traffic can be load-balanced across all the multihomed links on and toward the CE device.

Requirements

This example uses the following hardware and software components:

- Five MX Series 5G Universal Routing Platforms with MPC interfaces only, where:
 - Three devices are configured as provider edge (PE) routers connected to a common multihomed customer site.
 - One device is configured as a remote PE router connected to a single-homed customer site.
- Six customer edge (CE) devices, with one multihomed CE device and the rest of the CE devices being single-homed to each of the PE routers.
- Junos OS Release 16.1 or later running on all the PE routers.

NOTE: The EVPN multihoming active-active mode of operation is supported in Junos OS Releases 16.1 and later releases.

Starting with Junos OS Release 16.1R4, EVPN multihoming active-active mode is supported on all EX9200 switches. For information about configuration specific to EX9200 switches, see ["Configuration on EX9200 Switches" on page 245](#).

Before you begin:

1. Configure the router interfaces.
2. Configure IS-IS or any other IGP protocol.
3. Configure BGP.
4. Configure LDP.
5. Configure MPLS.
6. Configure RSVP MPLS LSP or GRE tunnels.

Overview and Topology

Starting with Junos OS Release 15.1, the EVPN solution on MX Series routers with MPC interfaces is extended to provide multihoming functionality in the active-active redundancy mode of operation. This feature enables load balancing of Layer 2 unicast traffic across all the multihomed links on and toward a customer edge device.

The EVPN active-active multihoming feature provides link-level and node-level redundancy along with effective utilization of resources.

To enable EVPN active-active multihoming, include the `all-active` statement at the `[edit interfaces esi]` hierarchy level.

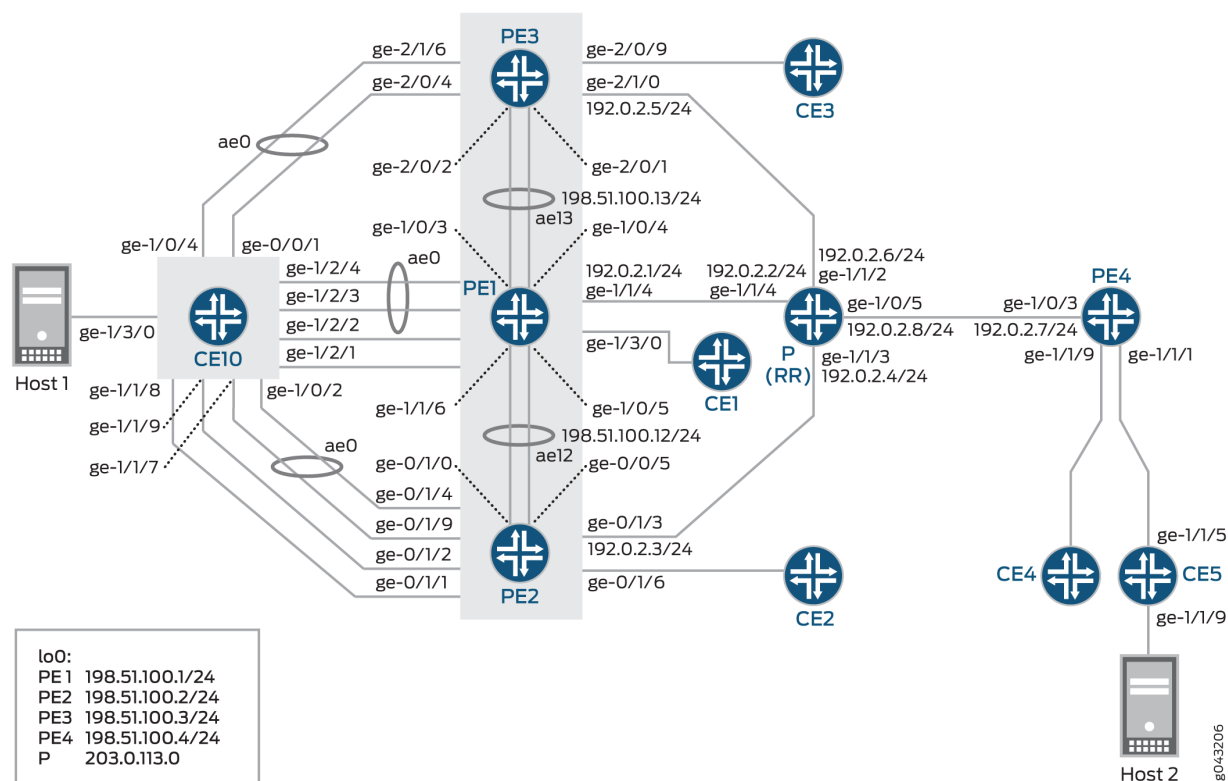
In [Figure 12 on page 215](#), the core consists of four provider edge (PE) routers and a provider router (P) that is configured as a route reflector (RR). Router CE10 is multihomed to Routers PE1, PE2, and PE3. Each PE router is also connected to a single-homed customer site.

There are three routing instances running in the topology – VS-1, VS-2, and mhevpn, along with the default routing instance. The routing instances share nine VLANs with three ESIs each. The VS-1 and VS-2 routing instances are configured as a virtual switch type of routing instance, and the mhevpn routing instance is an EVPN routing instance.

Three aggregated bundles – ae0, ae1, and ae2 – are used to connect the multihomed CE device, CE10, to Routers PE1, PE2, and PE3. These aggregated bundles are configured for three ESIs each. The

aggregated bundles, ae12 and ae13, are used to interconnect Routers PE1 and PE2, and PE1 and PE3, respectively.

Figure 12: EVPN Active-Active Multihoming Topology



Configuration

IN THIS SECTION

- [CLI Quick Configuration | 216](#)
- [Procedure | 234](#)
- [Configuration on EX9200 Switches | 245](#)
- [Results | 246](#)

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the [edit] hierarchy level, and then enter `commit` from configuration mode.

CE10

```
set chassis aggregated-devices ethernet device-count 20
set chassis network-services enhanced-ip
set interfaces ge-0/0/1 gigether-options 802.3ad ae0
set interfaces ge-1/0/2 gigether-options 802.3ad ae0
set interfaces ge-1/0/4 gigether-options 802.3ad ae0
set interfaces ge-1/1/7 gigether-options 802.3ad ae0
set interfaces ge-1/1/8 gigether-options 802.3ad ae2
set interfaces ge-1/1/9 gigether-options 802.3ad ae1
set interfaces ge-1/2/1 gigether-options 802.3ad ae2
set interfaces ge-1/2/2 gigether-options 802.3ad ae1
set interfaces ge-1/2/3 gigether-options 802.3ad ae0
set interfaces ge-1/2/4 gigether-options 802.3ad ae0
set interfaces ge-1/3/0 flexible-vlan-tagging
set interfaces ge-1/3/0 encapsulation flexible-ethernet-services
set interfaces ge-1/3/0 unit 0 encapsulation vlan-bridge
set interfaces ge-1/3/0 unit 0 vlan-id 10
set interfaces ge-1/3/0 unit 1 encapsulation vlan-bridge
set interfaces ge-1/3/0 unit 1 vlan-id 20
set interfaces ge-1/3/0 unit 2 encapsulation vlan-bridge
set interfaces ge-1/3/0 unit 2 vlan-id 30
set interfaces ge-1/3/0 unit 110 encapsulation vlan-bridge
set interfaces ge-1/3/0 unit 110 vlan-id 110
set interfaces ge-1/3/0 unit 120 encapsulation vlan-bridge
set interfaces ge-1/3/0 unit 120 vlan-id 120
set interfaces ge-1/3/0 unit 130 encapsulation vlan-bridge
set interfaces ge-1/3/0 unit 130 vlan-id 130
set interfaces ge-1/3/0 unit 210 encapsulation vlan-bridge
set interfaces ge-1/3/0 unit 210 vlan-id 210
set interfaces ge-1/3/0 unit 220 encapsulation vlan-bridge
set interfaces ge-1/3/0 unit 220 vlan-id 220
set interfaces ge-1/3/0 unit 230 encapsulation vlan-bridge
set interfaces ge-1/3/0 unit 230 vlan-id 230
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
```

```

set interfaces ae0 aggregated-ether-options minimum-links 1
set interfaces ae0 unit 0 encapsulation vlan-bridge
set interfaces ae0 unit 0 vlan-id 10
set interfaces ae0 unit 110 encapsulation vlan-bridge
set interfaces ae0 unit 110 vlan-id 110
set interfaces ae0 unit 210 encapsulation vlan-bridge
set interfaces ae0 unit 210 vlan-id 210
set interfaces ae1 flexible-vlan-tagging
set interfaces ae1 encapsulation flexible-ethernet-services
set interfaces ae1 aggregated-ether-options minimum-links 1
set interfaces ae1 unit 1 encapsulation vlan-bridge
set interfaces ae1 unit 1 vlan-id 20
set interfaces ae1 unit 120 encapsulation vlan-bridge
set interfaces ae1 unit 120 vlan-id 120
set interfaces ae1 unit 220 encapsulation vlan-bridge
set interfaces ae1 unit 220 vlan-id 220
set interfaces ae2 flexible-vlan-tagging
set interfaces ae2 encapsulation flexible-ethernet-services
set interfaces ae2 aggregated-ether-options minimum-links 1
set interfaces ae2 unit 2 encapsulation vlan-bridge
set interfaces ae2 unit 2 vlan-id 30
set interfaces ae2 unit 130 encapsulation vlan-bridge
set interfaces ae2 unit 130 vlan-id 130
set interfaces ae2 unit 230 encapsulation vlan-bridge
set interfaces ae2 unit 230 vlan-id 230
set routing-options forwarding-table export load-balancing-policy
set protocols l2-learning global-mac-table-aging-time 18000
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set bridge-domains bd10 domain-type bridge
set bridge-domains bd10 vlan-id 10
set bridge-domains bd10 interface ae0.0
set bridge-domains bd10 interface ge-1/3/0.0
set bridge-domains bd110 domain-type bridge
set bridge-domains bd110 vlan-id 110
set bridge-domains bd110 interface ae0.110
set bridge-domains bd110 interface ge-1/3/0.110
set bridge-domains bd120 domain-type bridge
set bridge-domains bd120 vlan-id 120
set bridge-domains bd120 interface ge-1/3/0.120
set bridge-domains bd120 interface ae1.120
set bridge-domains bd130 domain-type bridge
set bridge-domains bd130 vlan-id 130
set bridge-domains bd130 interface ge-1/3/0.130

```

```

set bridge-domains bd130 interface ae2.130
set bridge-domains bd20 domain-type bridge
set bridge-domains bd20 vlan-id 20
set bridge-domains bd20 interface ge-1/3/0.1
set bridge-domains bd20 interface ae1.1
set bridge-domains bd210 domain-type bridge
set bridge-domains bd210 vlan-id 210
set bridge-domains bd210 interface ae0.210
set bridge-domains bd210 interface ge-1/3/0.210
set bridge-domains bd220 domain-type bridge
set bridge-domains bd220 vlan-id 220
set bridge-domains bd220 interface ge-1/3/0.220
set bridge-domains bd220 interface ae1.220
set bridge-domains bd230 domain-type bridge
set bridge-domains bd230 vlan-id 230
set bridge-domains bd230 interface ge-1/3/0.230
set bridge-domains bd230 interface ae2.230
set bridge-domains bd30 domain-type bridge
set bridge-domains bd30 vlan-id 30
set bridge-domains bd30 interface ge-1/3/0.2
set bridge-domains bd30 interface ae2.2

```

CE5

```

set chassis aggregated-devices ethernet device-count 20
set chassis network-services enhanced-ip
set interfaces ge-1/1/5 gigether-options 802.3ad ae0
set interfaces ge-1/1/9 flexible-vlan-tagging
set interfaces ge-1/1/9 encapsulation flexible-ethernet-services
set interfaces ge-1/1/9 unit 0 encapsulation vlan-bridge
set interfaces ge-1/1/9 unit 0 vlan-id 10
set interfaces ge-1/1/9 unit 1 encapsulation vlan-bridge
set interfaces ge-1/1/9 unit 1 vlan-id 20
set interfaces ge-1/1/9 unit 2 encapsulation vlan-bridge
set interfaces ge-1/1/9 unit 2 vlan-id 30
set interfaces ge-1/1/9 unit 110 encapsulation vlan-bridge
set interfaces ge-1/1/9 unit 110 vlan-id 110
set interfaces ge-1/1/9 unit 120 encapsulation vlan-bridge
set interfaces ge-1/1/9 unit 120 vlan-id 120
set interfaces ge-1/1/9 unit 130 encapsulation vlan-bridge
set interfaces ge-1/1/9 unit 130 vlan-id 130
set interfaces ge-1/1/9 unit 210 encapsulation vlan-bridge

```

```

set interfaces ge-1/1/9 unit 210 vlan-id 210
set interfaces ge-1/1/9 unit 220 encapsulation vlan-bridge
set interfaces ge-1/1/9 unit 220 vlan-id 220
set interfaces ge-1/1/9 unit 230 encapsulation vlan-bridge
set interfaces ge-1/1/9 unit 230 vlan-id 230
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 unit 0 encapsulation vlan-bridge
set interfaces ae0 unit 0 vlan-id 10
set interfaces ae0 unit 1 encapsulation vlan-bridge
set interfaces ae0 unit 1 vlan-id 20
set interfaces ae0 unit 2 encapsulation vlan-bridge
set interfaces ae0 unit 2 vlan-id 30
set interfaces ae0 unit 110 encapsulation vlan-bridge
set interfaces ae0 unit 110 vlan-id 110
set interfaces ae0 unit 120 encapsulation vlan-bridge
set interfaces ae0 unit 120 vlan-id 120
set interfaces ae0 unit 130 encapsulation vlan-bridge
set interfaces ae0 unit 130 vlan-id 130
set interfaces ae0 unit 210 encapsulation vlan-bridge
set interfaces ae0 unit 210 vlan-id 210
set interfaces ae0 unit 220 encapsulation vlan-bridge
set interfaces ae0 unit 220 vlan-id 220
set interfaces ae0 unit 230 encapsulation vlan-bridge
set interfaces ae0 unit 230 vlan-id 230
set interfaces lo0 unit 6 family inet address 203.0.113.0/24
set interfaces lo0 unit 6 family iso address 47.0005.80ff.f800.0000.0108.0000.6006.0070.0600
set routing-options forwarding-table export load-balancing-policy
set protocols l2-learning global-mac-table-aging-time 18000
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set bridge-domains bd10 domain-type bridge
set bridge-domains bd10 vlan-id 10
set bridge-domains bd10 interface ae0.0
set bridge-domains bd10 interface ge-1/1/9.0
set bridge-domains bd110 domain-type bridge
set bridge-domains bd110 vlan-id 110
set bridge-domains bd110 interface ae0.110
set bridge-domains bd110 interface ge-1/1/9.110
set bridge-domains bd120 domain-type bridge
set bridge-domains bd120 vlan-id 120
set bridge-domains bd120 interface ge-1/1/9.120
set bridge-domains bd120 interface ae0.120
set bridge-domains bd130 domain-type bridge

```

```

set bridge-domains bd130 vlan-id 130
set bridge-domains bd130 interface ge-1/1/9.130
set bridge-domains bd130 interface ae0.130
set bridge-domains bd20 domain-type bridge
set bridge-domains bd20 vlan-id 20
set bridge-domains bd20 interface ae0.1
set bridge-domains bd20 interface ge-1/1/9.1
set bridge-domains bd210 domain-type bridge
set bridge-domains bd210 vlan-id 210
set bridge-domains bd210 interface ae0.210
set bridge-domains bd210 interface ge-1/1/9.210
set bridge-domains bd220 domain-type bridge
set bridge-domains bd220 vlan-id 220
set bridge-domains bd220 interface ge-1/1/9.220
set bridge-domains bd220 interface ae0.220
set bridge-domains bd230 domain-type bridge
set bridge-domains bd230 vlan-id 230
set bridge-domains bd230 interface ge-1/1/9.230
set bridge-domains bd230 interface ae0.230
set bridge-domains bd30 domain-type bridge
set bridge-domains bd30 vlan-id 30
set bridge-domains bd30 interface ge-1/1/9.2
set bridge-domains bd30 interface ae0.2

```

PE1

```

set chassis aggregated-devices ethernet device-count 20
set chassis network-services enhanced-ip
set interfaces ge-1/0/3 gigether-options 802.3ad ae13
set interfaces ge-1/0/4 gigether-options 802.3ad ae13
set interfaces ge-1/0/5 gigether-options 802.3ad ae12
set interfaces ge-1/1/4 unit 0 family inet address 192.0.2.1/24
set interfaces ge-1/1/4 unit 0 family iso
set interfaces ge-1/1/4 unit 0 family mpls
set interfaces ge-1/1/6 gigether-options 802.3ad ae12
set interfaces ge-1/2/1 flexible-vlan-tagging
set interfaces ge-1/2/1 encapsulation flexible-ethernet-services
set interfaces ge-1/2/1 esi 00:33:33:33:33:33:33:33:33:33
set interfaces ge-1/2/1 esi all-active
set interfaces ge-1/2/1 unit 0 encapsulation vlan-bridge
set interfaces ge-1/2/1 unit 0 vlan-id 30
set interfaces ge-1/2/1 unit 130 family bridge interface-mode trunk

```

```

set interfaces ge-1/2/1 unit 130 family bridge vlan-id-list 130
set interfaces ge-1/2/1 unit 230 family bridge interface-mode trunk
set interfaces ge-1/2/1 unit 230 family bridge vlan-id-list 230
set interfaces ge-1/2/2 flexible-vlan-tagging
set interfaces ge-1/2/2 encapsulation flexible-ethernet-services
set interfaces ge-1/2/2 esi 00:22:22:22:22:22:22:22:22
set interfaces ge-1/2/2 esi all-active
set interfaces ge-1/2/2 unit 0 encapsulation vlan-bridge
set interfaces ge-1/2/2 unit 0 vlan-id 20
set interfaces ge-1/2/2 unit 120 family bridge interface-mode trunk
set interfaces ge-1/2/2 unit 120 family bridge vlan-id-list 120
set interfaces ge-1/2/2 unit 220 family bridge interface-mode trunk
set interfaces ge-1/2/2 unit 220 family bridge vlan-id-list 220
set interfaces ge-1/2/3 gigether-options 802.3ad ae0
set interfaces ge-1/2/4 gigether-options 802.3ad ae0
set interfaces ge-1/3/0 flexible-vlan-tagging
set interfaces ge-1/3/0 encapsulation flexible-ethernet-services
set interfaces ge-1/3/0 unit 0 encapsulation vlan-bridge
set interfaces ge-1/3/0 unit 0 vlan-id 10
set interfaces ge-1/3/0 unit 100 family bridge interface-mode trunk
set interfaces ge-1/3/0 unit 100 family bridge vlan-id-list 110
set interfaces ge-1/3/0 unit 100 family bridge vlan-id-list 120
set interfaces ge-1/3/0 unit 100 family bridge vlan-id-list 130
set interfaces ge-1/3/0 unit 200 family bridge interface-mode trunk
set interfaces ge-1/3/0 unit 200 family bridge vlan-id-list 210
set interfaces ge-1/3/0 unit 200 family bridge vlan-id-list 220
set interfaces ge-1/3/0 unit 200 family bridge vlan-id-list 230
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 esi 00:11:11:11:11:11:11:11:11
set interfaces ae0 esi all-active
set interfaces ae0 unit 0 encapsulation vlan-bridge
set interfaces ae0 unit 0 vlan-id 10
set interfaces ae0 unit 110 family bridge interface-mode trunk
set interfaces ae0 unit 110 family bridge vlan-id-list 110
set interfaces ae0 unit 210 family bridge interface-mode trunk
set interfaces ae0 unit 210 family bridge vlan-id-list 210
set interfaces ae12 flexible-vlan-tagging
set interfaces ae12 encapsulation flexible-ethernet-services
set interfaces ae12 aggregated-ether-options minimum-links 1
set interfaces ae12 unit 0 vlan-id 1200
set interfaces ae12 unit 0 family inet address 198.51.100.12/24
set interfaces ae12 unit 0 family iso

```



```

set interfaces ae12 unit 0 family mpls
set interfaces ae13 flexible-vlan-tagging
set interfaces ae13 encapsulation flexible-ethernet-services
set interfaces ae13 aggregated-ether-options minimum-links 1
set interfaces ae13 unit 0 vlan-tags outer 1300
set interfaces ae13 unit 0 vlan-tags inner 13
set interfaces ae13 unit 0 family inet address 198.51.100.13/24
set interfaces ae13 unit 0 family iso
set interfaces ae13 unit 0 family mpls
set interfaces irb unit 0 family inet address 192.0.2.9/24
set interfaces irb unit 0 mac 00:99:99:99:01:99
set interfaces irb unit 1 family inet address 192.0.2.10/24
set interfaces irb unit 1 mac 00:99:99:99:02:99
set interfaces irb unit 2 family inet address 192.0.2.11/24
set interfaces irb unit 2 mac 00:99:99:99:03:99
set interfaces irb unit 10 family inet address 192.0.2.12/24
set interfaces irb unit 10 mac 00:99:99:99:01:90
set interfaces lo0 unit 0 family inet address 198.51.100.1/24 primary
set interfaces lo0 unit 0 family iso
set routing-options router-id 198.51.100.1
set routing-options autonomous-system 65221
set routing-options forwarding-table export load-balancing-policy
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path pe1tope2 from 198.51.100.1
set protocols mpls label-switched-path pe1tope2 to 198.51.100.2
set protocols mpls label-switched-path pe1tope2 primary direct_to_pe2
set protocols mpls label-switched-path pe1tope3 from 198.51.100.1
set protocols mpls label-switched-path pe1tope3 to 198.51.100.3
set protocols mpls label-switched-path pe1tope3 primary direct_to_pe3
set protocols mpls label-switched-path pe1tope4 from 198.51.100.1
set protocols mpls label-switched-path pe1tope4 to 198.51.100.4
set protocols mpls label-switched-path pe1tope4 primary direct_to_pe4
set protocols mpls path pe4_to_pe3 198.51.100.4 strict
set protocols mpls path pe4_to_pe3 198.51.100.3 strict
set protocols mpls path direct_to_pe2 198.51.100.5 strict
set protocols mpls path direct_to_pe3 198.51.100.6 strict
set protocols mpls path direct_to_pe4 198.51.100.9 strict
set protocols mpls path pe2_to_pe3 198.51.100.2 strict
set protocols mpls path pe2_to_pe3 198.51.100.3 strict
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group RR type internal

```

```

set protocols bgp group RR local-address 198.51.100.1
set protocols bgp group RR family evpn signaling
set protocols bgp group RR neighbor 203.0.113.0
set protocols isis level 1 disable
set protocols isis interface all level 2 metric 10
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 level 2 metric 0
set protocols ldp deaggregate
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set protocols evpn
set protocols l2-learning global-mac-table-aging-time 18000
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set routing-instances VS-1 instance-type virtual-switch
set routing-instances VS-1 interface ge-1/2/1.130
set routing-instances VS-1 interface ge-1/2/2.120
set routing-instances VS-1 interface ge-1/3/0.100
set routing-instances VS-1 interface ae0.110
set routing-instances VS-1 route-distinguisher 198.51.100.1:101
set routing-instances VS-1 vrf-target target:100:101
set routing-instances VS-1 protocols evpn extended-vlan-list 110
set routing-instances VS-1 protocols evpn extended-vlan-list 120
set routing-instances VS-1 protocols evpn extended-vlan-list 130
set routing-instances VS-1 protocols evpn default-gateway do-not-advertise
set routing-instances VS-1 bridge-domains bd-110 vlan-id 110
set routing-instances VS-1 bridge-domains bd-110 routing-interface irb.0
set routing-instances VS-1 bridge-domains bd-120 vlan-id 120
set routing-instances VS-1 bridge-domains bd-120 routing-interface irb.1
set routing-instances VS-1 bridge-domains bd-130 vlan-id 130
set routing-instances VS-1 bridge-domains bd-130 routing-interface irb.2
set routing-instances VS-2 instance-type virtual-switch
set routing-instances VS-2 interface ge-1/2/1.230
set routing-instances VS-2 interface ge-1/2/2.220
set routing-instances VS-2 interface ge-1/3/0.200
set routing-instances VS-2 interface ae0.210
set routing-instances VS-2 route-distinguisher 198.51.100.1:201
set routing-instances VS-2 vrf-target target:100:201
set routing-instances VS-2 protocols evpn extended-vlan-list 210
set routing-instances VS-2 protocols evpn extended-vlan-list 220
set routing-instances VS-2 protocols evpn extended-vlan-list 230
set routing-instances VS-2 bridge-domains bd-a vlan-id-list 210
set routing-instances VS-2 bridge-domains bd-a vlan-id-list 220
set routing-instances VS-2 bridge-domains bd-a vlan-id-list 230

```

```

set routing-instances mhevpn instance-type evpn
set routing-instances mhevpn vlan-id 10
set routing-instances mhevpn interface ge-1/2/1.0
set routing-instances mhevpn interface ge-1/2/2.0
set routing-instances mhevpn interface ge-1/3/0.0
set routing-instances mhevpn interface ae0.0
set routing-instances mhevpn routing-interface irb.10
set routing-instances mhevpn route-distinguisher 198.51.100.1:1
set routing-instances mhevpn vrf-target target:100:1
set routing-instances mhevpn protocols evpn default-gateway do-not-advertise
set routing-instances vrf instance-type vrf
set routing-instances vrf interface irb.0
set routing-instances vrf interface irb.1
set routing-instances vrf interface irb.2
set routing-instances vrf interface irb.10
set routing-instances vrf route-distinguisher 198.51.100.1:11
set routing-instances vrf vrf-target target:100:11
set routing-instances vrf vrf-table-label

```

PE2

```

set chassis aggregated-devices ethernet device-count 20
set chassis network-services enhanced-ip
set interfaces ge-0/0/5 gigether-options 802.3ad ae12
set interfaces ge-0/1/0 gigether-options 802.3ad ae12
set interfaces ge-0/1/1 flexible-vlan-tagging
set interfaces ge-0/1/1 encapsulation flexible-ethernet-services
set interfaces ge-0/1/1 esi 00:33:33:33:33:33:33:33:33:33
set interfaces ge-0/1/1 esi all-active
set interfaces ge-0/1/1 unit 0 encapsulation vlan-bridge
set interfaces ge-0/1/1 unit 0 vlan-id 30
set interfaces ge-0/1/1 unit 130 family bridge interface-mode trunk
set interfaces ge-0/1/1 unit 130 family bridge vlan-id-list 130
set interfaces ge-0/1/1 unit 230 family bridge interface-mode trunk
set interfaces ge-0/1/1 unit 230 family bridge vlan-id-list 230
set interfaces ge-0/1/2 flexible-vlan-tagging
set interfaces ge-0/1/2 encapsulation flexible-ethernet-services
set interfaces ge-0/1/2 esi 00:22:22:22:22:22:22:22:22:22
set interfaces ge-0/1/2 esi all-active
set interfaces ge-0/1/2 unit 0 encapsulation vlan-bridge
set interfaces ge-0/1/2 unit 0 vlan-id 20
set interfaces ge-0/1/2 unit 120 family bridge interface-mode trunk

```

```

set interfaces ge-0/1/2 unit 120 family bridge vlan-id-list 120
set interfaces ge-0/1/2 unit 220 family bridge interface-mode trunk
set interfaces ge-0/1/2 unit 220 family bridge vlan-id-list 220
set interfaces ge-0/1/3 unit 0 family inet address 192.0.2.3/24
set interfaces ge-0/1/3 unit 0 family iso
set interfaces ge-0/1/3 unit 0 family mpls
set interfaces ge-0/1/4 gigether-options 802.3ad ae0
set interfaces ge-0/1/6 flexible-vlan-tagging
set interfaces ge-0/1/6 encapsulation flexible-ethernet-services
set interfaces ge-0/1/6 unit 0 encapsulation vlan-bridge
set interfaces ge-0/1/6 unit 0 vlan-id 10
set interfaces ge-0/1/6 unit 100 family bridge interface-mode trunk
set interfaces ge-0/1/6 unit 100 family bridge vlan-id-list 110
set interfaces ge-0/1/6 unit 100 family bridge vlan-id-list 120
set interfaces ge-0/1/6 unit 100 family bridge vlan-id-list 130
set interfaces ge-0/1/6 unit 200 family bridge interface-mode trunk
set interfaces ge-0/1/6 unit 200 family bridge vlan-id-list 210
set interfaces ge-0/1/6 unit 200 family bridge vlan-id-list 220
set interfaces ge-0/1/6 unit 200 family bridge vlan-id-list 230
set interfaces ge-0/1/9 gigether-options 802.3ad ae0
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 esi 00:11:11:11:11:11:11:11:11:11
set interfaces ae0 esi all-active
set interfaces ae0 unit 0 encapsulation vlan-bridge
set interfaces ae0 unit 0 vlan-id 10
set interfaces ae0 unit 110 family bridge interface-mode trunk
set interfaces ae0 unit 110 family bridge vlan-id-list 110
set interfaces ae0 unit 210 family bridge interface-mode trunk
set interfaces ae0 unit 210 family bridge vlan-id-list 210
set interfaces ae12 flexible-vlan-tagging
set interfaces ae12 encapsulation flexible-ethernet-services
set interfaces ae12 aggregated-ether-options minimum-links 1
set interfaces ae12 unit 0 vlan-id 1200
set interfaces ae12 unit 0 family inet address 198.51.100.5/24
set interfaces ae12 unit 0 family iso
set interfaces ae12 unit 0 family mpls
set interfaces irb unit 0 family inet address 192.0.2.9/24
set interfaces irb unit 0 mac 00:99:99:99:01:99
set interfaces irb unit 1 family inet address 192.0.2.10/24
set interfaces irb unit 1 mac 00:99:99:99:02:99
set interfaces irb unit 2 family inet address 192.0.2.11/24
set interfaces irb unit 2 mac 00:99:99:99:03:99

```

```

set interfaces irb unit 10 family inet address 192.0.2.12/24
set interfaces irb unit 10 mac 00:99:99:99:01:90
set interfaces lo0 unit 0 family inet address 198.51.100.2/32 primary
set interfaces lo0 unit 0 family iso
set routing-options router-id 198.51.100.2
set routing-options autonomous-system 65221
set routing-options forwarding-table export load-balancing-policy
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path pe2tope1 from 198.51.100.2
set protocols mpls label-switched-path pe2tope1 to 198.51.100.1
set protocols mpls label-switched-path pe2tope1 primary direct_to_pe1
set protocols mpls label-switched-path pe2tope3 from 198.51.100.2
set protocols mpls label-switched-path pe2tope3 to 198.51.100.3
set protocols mpls label-switched-path pe2tope3 primary direct_to_pe3
set protocols mpls label-switched-path pe2tope4 from 198.51.100.2
set protocols mpls label-switched-path pe2tope4 to 198.51.100.4
set protocols mpls label-switched-path pe2tope4 primary direct_to_pe4
set protocols mpls path direct_to_pe1 198.51.100.12 strict
set protocols mpls path direct_to_pe3 198.51.100.7 strict
set protocols mpls path direct_to_pe4 198.51.100.8 strict
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group RR type internal
set protocols bgp group RR local-address 198.51.100.2
set protocols bgp group RR family evpn signaling
set protocols bgp group RR neighbor 203.0.113.0
set protocols isis level 1 disable
set protocols isis interface all level 2 metric 10
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 level 2 metric 0
set protocols ldp deaggregate
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set protocols l2-learning global-mac-table-aging-time 18000
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set routing-instances VS-1 instance-type virtual-switch
set routing-instances VS-1 interface ge-0/1/1.130
set routing-instances VS-1 interface ge-0/1/2.120
set routing-instances VS-1 interface ge-0/1/6.100
set routing-instances VS-1 interface ae0.110
set routing-instances VS-1 route-distinguisher 198.51.100.2:101
set routing-instances VS-1 vrf-target target:100:101

```

```

set routing-instances VS-1 protocols evpn extended-vlan-list 110
set routing-instances VS-1 protocols evpn extended-vlan-list 120
set routing-instances VS-1 protocols evpn extended-vlan-list 130
set routing-instances VS-1 protocols evpn default-gateway do-not-advertise
set routing-instances VS-1 bridge-domains bd-110 vlan-id 110
set routing-instances VS-1 bridge-domains bd-110 routing-interface irb.0
set routing-instances VS-1 bridge-domains bd-120 vlan-id 120
set routing-instances VS-1 bridge-domains bd-120 routing-interface irb.1
set routing-instances VS-1 bridge-domains bd-130 vlan-id 130
set routing-instances VS-1 bridge-domains bd-130 routing-interface irb.2
set routing-instances VS-2 instance-type virtual-switch
set routing-instances VS-2 interface ge-0/1/1.230
set routing-instances VS-2 interface ge-0/1/2.220
set routing-instances VS-2 interface ge-0/1/6.200
set routing-instances VS-2 interface ae0.210
set routing-instances VS-2 route-distinguisher 198.51.100.2:201
set routing-instances VS-2 vrf-target target:100:201
set routing-instances VS-2 protocols evpn extended-vlan-list 210
set routing-instances VS-2 protocols evpn extended-vlan-list 220
set routing-instances VS-2 protocols evpn extended-vlan-list 230
set routing-instances VS-2 bridge-domains bd-a vlan-id-list 210
set routing-instances VS-2 bridge-domains bd-a vlan-id-list 220
set routing-instances VS-2 bridge-domains bd-a vlan-id-list 230
set routing-instances mhevpn instance-type evpn
set routing-instances mhevpn vlan-id 10
set routing-instances mhevpn interface ge-0/1/1.0
set routing-instances mhevpn interface ge-0/1/2.0
set routing-instances mhevpn interface ge-0/1/6.0
set routing-instances mhevpn interface ae0.0
set routing-instances mhevpn routing-interface irb.10
set routing-instances mhevpn route-distinguisher 198.51.100.2:1
set routing-instances mhevpn vrf-target target:100:1
set routing-instances mhevpn protocols evpn default-gateway do-not-advertise
set routing-instances vrf instance-type vrf
set routing-instances vrf interface irb.0
set routing-instances vrf interface irb.1
set routing-instances vrf interface irb.2
set routing-instances vrf interface irb.10
set routing-instances vrf route-distinguisher 198.51.100.2:11
set routing-instances vrf vrf-target target:100:11
set routing-instances vrf vrf-table-label

```

PE3

```

set chassis aggregated-devices ethernet device-count 20
set chassis network-services enhanced-ip
set interfaces ge-2/0/1 gigether-options 802.3ad ae13
set interfaces ge-2/0/2 gigether-options 802.3ad ae13
set interfaces ge-2/0/4 gigether-options 802.3ad ae0
set interfaces ge-2/0/9 flexible-vlan-tagging
set interfaces ge-2/0/9 encapsulation flexible-ethernet-services
set interfaces ge-2/0/9 unit 0 encapsulation vlan-bridge
set interfaces ge-2/0/9 unit 0 vlan-id 10
set interfaces ge-2/0/9 unit 100 family bridge interface-mode trunk
set interfaces ge-2/0/9 unit 100 family bridge vlan-id-list 110
set interfaces ge-2/0/9 unit 100 family bridge vlan-id-list 120
set interfaces ge-2/0/9 unit 100 family bridge vlan-id-list 130
set interfaces ge-2/0/9 unit 200 family bridge interface-mode trunk
set interfaces ge-2/0/9 unit 200 family bridge vlan-id-list 210
set interfaces ge-2/0/9 unit 200 family bridge vlan-id-list 220
set interfaces ge-2/0/9 unit 200 family bridge vlan-id-list 230
set interfaces ge-2/1/0 unit 0 family inet address 192.0.2.5/24
set interfaces ge-2/1/0 unit 0 family iso
set interfaces ge-2/1/0 unit 0 family mpls
set interfaces ge-2/1/6 gigether-options 802.3ad ae0
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 esi 00:11:11:11:11:11:11:11:11
set interfaces ae0 esi all-active
set interfaces ae0 unit 0 encapsulation vlan-bridge
set interfaces ae0 unit 0 vlan-id 10
set interfaces ae0 unit 110 family bridge interface-mode trunk
set interfaces ae0 unit 110 family bridge vlan-id-list 110
set interfaces ae0 unit 210 family bridge interface-mode trunk
set interfaces ae0 unit 210 family bridge vlan-id-list 210
set interfaces ae13 flexible-vlan-tagging
set interfaces ae13 encapsulation flexible-ethernet-services
set interfaces ae13 aggregated-ether-options minimum-links 1
set interfaces ae13 unit 0 vlan-tags outer 1300
set interfaces ae13 unit 0 vlan-tags inner 13
set interfaces ae13 unit 0 family inet address 198.51.100.6/24
set interfaces ae13 unit 0 family iso
set interfaces ae13 unit 0 family mpls
set interfaces irb unit 0 family inet address 192.0.2.9/24

```

```

set interfaces irb unit 0 mac 00:99:99:99:01:99
set interfaces irb unit 1 family inet address 192.0.2.10/24
set interfaces irb unit 1 mac 00:99:99:99:02:99
set interfaces irb unit 2 family inet address 192.0.2.11/24
set interfaces irb unit 2 mac 00:99:99:99:03:99
set interfaces irb unit 10 family inet address 192.0.2.12/24
set interfaces irb unit 10 mac 00:99:99:99:01:90
set interfaces lo0 unit 0 family inet address 198.51.100.3/32 primary
set interfaces lo0 unit 0 family iso
set routing-options router-id 198.51.100.3
set routing-options autonomous-system 65221
set routing-options forwarding-table export load-balancing-policy
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path pe3tope1 from 198.51.100.3
set protocols mpls label-switched-path pe3tope1 to 198.51.100.1
set protocols mpls label-switched-path pe3tope1 primary direct_to_pe1
set protocols mpls label-switched-path pe3tope2 from 198.51.100.3
set protocols mpls label-switched-path pe3tope2 to 198.51.100.2
set protocols mpls label-switched-path pe3tope2 primary direct_to_pe2
set protocols mpls label-switched-path pe3tope4 from 198.51.100.3
set protocols mpls label-switched-path pe3tope4 to 198.51.100.4
set protocols mpls label-switched-path pe3tope4 primary direct_to_pe4
set protocols mpls path direct_to_pe1 198.51.100.13 strict
set protocols mpls path direct_to_pe2 198.51.100.10 strict
set protocols mpls path direct_to_pe4 198.51.100.11 strict
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group RR type internal
set protocols bgp group RR local-address 198.51.100.3
set protocols bgp group RR family evpn signaling
set protocols bgp group RR neighbor 203.0.113.0
set protocols isis level 1 disable
set protocols isis interface all level 2 metric 10
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 level 2 metric 0
set protocols ldp deaggregate
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set protocols l2-learning global-mac-table-aging-time 18000
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set routing-instances VS-1 instance-type virtual-switch
set routing-instances VS-1 interface ge-2/0/9.100

```



```

set routing-instances VS-1 interface ae0.110
set routing-instances VS-1 route-distinguisher 198.51.100.3:101
set routing-instances VS-1 vrf-target target:100:101
set routing-instances VS-1 protocols evpn extended-vlan-list 110
set routing-instances VS-1 protocols evpn extended-vlan-list 120
set routing-instances VS-1 protocols evpn extended-vlan-list 130
set routing-instances VS-1 protocols evpn default-gateway do-not-advertise
set routing-instances VS-1 bridge-domains bd-110 vlan-id 110
set routing-instances VS-1 bridge-domains bd-110 routing-interface irb.0
set routing-instances VS-1 bridge-domains bd-120 vlan-id 120
set routing-instances VS-1 bridge-domains bd-120 routing-interface irb.1
set routing-instances VS-1 bridge-domains bd-130 vlan-id 130
set routing-instances VS-1 bridge-domains bd-130 routing-interface irb.2
set routing-instances VS-2 instance-type virtual-switch
set routing-instances VS-2 interface ge-2/0/9.200
set routing-instances VS-2 interface ae0.210
set routing-instances VS-2 route-distinguisher 198.51.100.3:201
set routing-instances VS-2 vrf-target target:100:201
set routing-instances VS-2 protocols evpn extended-vlan-list 210
set routing-instances VS-2 protocols evpn extended-vlan-list 220
set routing-instances VS-2 protocols evpn extended-vlan-list 230
set routing-instances VS-2 bridge-domains bd-a vlan-id-list 210
set routing-instances VS-2 bridge-domains bd-a vlan-id-list 220
set routing-instances VS-2 bridge-domains bd-a vlan-id-list 230
set routing-instances mhevpn instance-type evpn
set routing-instances mhevpn vlan-id 10
set routing-instances mhevpn interface ge-2/0/9.0
set routing-instances mhevpn interface ae0.0
set routing-instances mhevpn routing-interface irb.10
set routing-instances mhevpn route-distinguisher 198.51.100.3:1
set routing-instances mhevpn vrf-target target:100:1
set routing-instances mhevpn protocols evpn default-gateway do-not-advertise
set routing-instances vrf instance-type vrf
set routing-instances vrf interface irb.0
set routing-instances vrf interface irb.1
set routing-instances vrf interface irb.2
set routing-instances vrf interface irb.10
set routing-instances vrf route-distinguisher 198.51.100.3:11
set routing-instances vrf vrf-target target:100:11
set routing-instances vrf vrf-table-label

```

PE4

```

set chassis aggregated-devices ethernet device-count 20
set chassis network-services enhanced-ip
set interfaces ge-1/0/3 unit 0 family inet address 192.0.2.7/24
set interfaces ge-1/0/3 unit 0 family iso
set interfaces ge-1/0/3 unit 0 family mpls
set interfaces ge-1/1/1 flexible-vlan-tagging
set interfaces ge-1/1/1 encapsulation flexible-ethernet-services
set interfaces ge-1/1/1 esi 00:44:44:44:44:44:44:44:44
set interfaces ge-1/1/1 esi all-active
set interfaces ge-1/1/1 unit 0 encapsulation vlan-bridge
set interfaces ge-1/1/1 unit 0 vlan-id 10
set interfaces ge-1/1/1 unit 100 family bridge interface-mode trunk
set interfaces ge-1/1/1 unit 100 family bridge vlan-id-list 110
set interfaces ge-1/1/1 unit 100 family bridge vlan-id-list 120
set interfaces ge-1/1/1 unit 100 family bridge vlan-id-list 130
set interfaces ge-1/1/1 unit 200 family bridge interface-mode trunk
set interfaces ge-1/1/1 unit 200 family bridge vlan-id-list 210
set interfaces ge-1/1/1 unit 200 family bridge vlan-id-list 220
set interfaces ge-1/1/1 unit 200 family bridge vlan-id-list 230
set interfaces ge-1/1/9 flexible-vlan-tagging
set interfaces ge-1/1/9 encapsulation flexible-ethernet-services
set interfaces ge-1/1/9 unit 0 encapsulation vlan-bridge
set interfaces ge-1/1/9 unit 0 vlan-id 10
set interfaces ge-1/1/9 unit 100 family bridge interface-mode trunk
set interfaces ge-1/1/9 unit 100 family bridge vlan-id-list 110
set interfaces ge-1/1/9 unit 100 family bridge vlan-id-list 120
set interfaces ge-1/1/9 unit 100 family bridge vlan-id-list 130
set interfaces ge-1/1/9 unit 200 family bridge interface-mode trunk
set interfaces ge-1/1/9 unit 200 family bridge vlan-id-list 210
set interfaces ge-1/1/9 unit 200 family bridge vlan-id-list 220
set interfaces ge-1/1/9 unit 200 family bridge vlan-id-list 230
set interfaces irb unit 0 family inet address 192.0.2.9/24
set interfaces irb unit 0 mac 00:99:99:99:01:99
set interfaces irb unit 1 family inet address 192.0.2.10/24
set interfaces irb unit 1 mac 00:99:99:99:02:99
set interfaces irb unit 2 family inet address 192.0.2.11/24
set interfaces irb unit 2 mac 00:99:99:99:03:99
set interfaces irb unit 10 family inet address 192.0.2.12/24
set interfaces irb unit 10 mac 00:99:99:99:01:90
set interfaces lo0 unit 0 family inet address 198.51.100.4/32 primary

```

```

set routing-options router-id 198.51.100.4
set routing-options autonomous-system 65221
set routing-options forwarding-table export load-balancing-policy
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path pe4tope1 from 198.51.100.4
set protocols mpls label-switched-path pe4tope1 to 198.51.100.1
set protocols mpls label-switched-path pe4tope1 primary direct_to_pe1
set protocols mpls label-switched-path pe4tope2 from 198.51.100.4
set protocols mpls label-switched-path pe4tope2 to 198.51.100.2
set protocols mpls label-switched-path pe4tope2 primary direct_to_pe2
set protocols mpls label-switched-path pe4tope3 from 198.51.100.4
set protocols mpls label-switched-path pe4tope3 to 198.51.100.3
set protocols mpls label-switched-path pe4tope3 primary direct_to_pe3
set protocols mpls path pe2_to_pe3 198.51.100.2 strict
set protocols mpls path pe2_to_pe3 198.51.100.3 strict
set protocols mpls path direct_to_pe1 198.51.100.14 strict
set protocols mpls path direct_to_pe2 198.51.100.15 strict
set protocols mpls path direct_to_pe3 198.51.100.16 strict
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group RR type internal
set protocols bgp group RR local-address 198.51.100.4
set protocols bgp group RR family evpn signaling
set protocols bgp group RR neighbor 203.0.113.0
set protocols isis level 1 disable
set protocols isis interface all level 2 metric 10
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 level 2 metric 0
set protocols ldp deaggregate
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set protocols l2-learning global-mac-table-aging-time 18000
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set routing-instances VS-1 instance-type virtual-switch
set routing-instances VS-1 interface ge-1/1/1.100
set routing-instances VS-1 interface ge-1/1/9.100
set routing-instances VS-1 route-distinguisher 198.51.100.4:101
set routing-instances VS-1 vrf-target target:100:101
set routing-instances VS-1 protocols evpn extended-vlan-list 110
set routing-instances VS-1 protocols evpn extended-vlan-list 120
set routing-instances VS-1 protocols evpn extended-vlan-list 130
set routing-instances VS-1 protocols evpn default-gateway do-not-advertise

```

```

set routing-instances VS-1 bridge-domains bd-110 vlan-id 110
set routing-instances VS-1 bridge-domains bd-110 routing-interface irb.0
set routing-instances VS-1 bridge-domains bd-120 vlan-id 120
set routing-instances VS-1 bridge-domains bd-120 routing-interface irb.1
set routing-instances VS-1 bridge-domains bd-130 vlan-id 130
set routing-instances VS-1 bridge-domains bd-130 routing-interface irb.2
set routing-instances VS-2 instance-type virtual-switch
set routing-instances VS-2 interface ge-1/1/1.200
set routing-instances VS-2 interface ge-1/1/9.200
set routing-instances VS-2 route-distinguisher 198.51.100.4:201
set routing-instances VS-2 vrf-target target:100:201
set routing-instances VS-2 protocols evpn extended-vlan-list 210
set routing-instances VS-2 protocols evpn extended-vlan-list 220
set routing-instances VS-2 protocols evpn extended-vlan-list 230
set routing-instances VS-2 bridge-domains bd-a vlan-id-list 210
set routing-instances VS-2 bridge-domains bd-a vlan-id-list 220
set routing-instances VS-2 bridge-domains bd-a vlan-id-list 230
set routing-instances mhevpn instance-type evpn
set routing-instances mhevpn vlan-id 10
set routing-instances mhevpn interface ge-1/1/1.0
set routing-instances mhevpn interface ge-1/1/9.0
set routing-instances mhevpn routing-interface irb.10
set routing-instances mhevpn route-distinguisher 198.51.100.4:1
set routing-instances mhevpn vrf-target target:100:1
set routing-instances mhevpn protocols evpn default-gateway do-not-advertise
set routing-instances vrf instance-type vrf
set routing-instances vrf interface irb.0
set routing-instances vrf interface irb.1
set routing-instances vrf interface irb.2
set routing-instances vrf interface irb.10
set routing-instances vrf route-distinguisher 198.51.100.4:11
set routing-instances vrf vrf-target target:100:11
set routing-instances vrf vrf-table-label

```

P (RR)

```

set interfaces ge-1/0/5 unit 0 family inet address 192.0.2.8/24
set interfaces ge-1/1/2 unit 0 family inet address 192.0.2.6/24
set interfaces ge-1/1/3 unit 0 family inet address 192.0.2.4/24
set interfaces ge-1/1/4 unit 0 family inet address 192.0.2.2/24
set interfaces lo0 unit 0 family inet address 203.0.113.0
set protocols bgp group RR type internal

```

```

set protocols bgp group RR local-address 203.0.113.0
set protocols bgp group RR family evpn signaling
set protocols bgp group RR cluster 1.2.3.4
set protocols bgp group RR neighbor 198.51.100.1
set protocols bgp group RR neighbor 198.51.100.2
set protocols bgp group RR neighbor 198.51.100.3
set protocols bgp group RR neighbor 198.51.100.4
set protocols isis interface all level 1 disable
set protocols ldp interface all
set routing-options router-id 203.0.113.0
set routing-options autonomous-system 65221

```

Procedure

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure Router PE1:

NOTE: Repeat this procedure for all other multihomed PE routers after modifying the appropriate interface names, addresses, and other parameters.

1. Configure the MX Series router to operate in the enhanced-ip mode because the EVPN active-active functionality is supported on routers with MPCs and MIC interfaces only.

A system reboot is required on committing this configuration.

```

[edit chassis]
user@PE1# set network-services enhanced-ip

```

2. Specify the number of aggregated Ethernet interfaces to be created.

```

[edit chassis]
user@PE1# set aggregated-devices ethernet device-count 20

```

3. Configure Router PE1 interfaces within the ae0 aggregated bundle toward the multihomed customer site, Router CE10.

- a. Assign interfaces ge-1/2/3 and ge-1/2/4 within the ae0 aggregated bundle.

```
[edit interfaces]
user@PE1# set ge-1/2/3 gigether-options 802.3ad ae0
user@PE1# set ge-1/2/4 gigether-options 802.3ad ae0
```

- b. Configure the ae0 aggregated bundle parameters for VLAN tagging and encapsulation.

```
[edit interfaces]
user@PE1# set ae0 flexible-vlan-tagging
user@PE1# set ae0 encapsulation flexible-ethernet-services
```

- c. Assign an ESI value for the first Ethernet segment and enable EVPN active-active multihoming for the ae0 aggregated bundle.

```
[edit interfaces]
user@PE1# set ae0 esi 00:11:11:11:11:11:11:11:11
user@PE1# set ae0 esi all-active
```

- d. Configure a trunk interface on the bridge network for the ae0 aggregated bundle.

```
[edit interfaces]
user@PE1# set ae0 unit 0 encapsulation vlan-bridge
user@PE1# set ae0 unit 0 vlan-id 10
user@PE1# set ae0 unit 110 family bridge interface-mode trunk
user@PE1# set ae0 unit 110 family bridge vlan-id-list 110
user@PE1# set ae0 unit 210 family bridge interface-mode trunk
user@PE1# set ae0 unit 210 family bridge vlan-id-list 210
```

4. Configure the other Router PE1 interfaces toward the multihomed customer site, Router CE10.

- a. Configure the VLAN tagging and encapsulation parameters for the ge-1/2/2 PE1 interface.

```
[edit interfaces]
user@PE1# set ge-1/2/2 flexible-vlan-tagging
user@PE1# set ge-1/2/2 encapsulation flexible-ethernet-services
```

- b. Assign an ESI value for the second Ethernet segment and enable EVPN active-active multihoming for the ge-1/2/2 PE1 interface.

```
[edit interfaces]
user@PE1# set ge-1/2/2 esi 00:22:22:22:22:22:22:22
user@PE1# set ge-1/2/2 esi all-active
```

- c. Configure a trunk interface on the bridge network for the ge-1/2/2 PE1 interface.

```
[edit interfaces]
user@PE1# set ge-1/2/2 unit 0 encapsulation vlan-bridge
user@PE1# set ge-1/2/2 unit 0 vlan-id 20
user@PE1# set ge-1/2/2 unit 120 family bridge interface-mode trunk
user@PE1# set ge-1/2/2 unit 120 family bridge vlan-id-list 120
user@PE1# set ge-1/2/2 unit 220 family bridge interface-mode trunk
user@PE1# set ge-1/2/2 unit 220 family bridge vlan-id-list 220
```

- d. Configure the VLAN tagging and encapsulation parameters for the ge-1/2/1 PE1 interface.

```
[edit interfaces]
user@PE1# set ge-1/2/1 flexible-vlan-tagging
user@PE1# set ge-1/2/1 encapsulation flexible-ethernet-services
```

- e. Assign an ESI value for the third Ethernet segment and enable EVPN active-active multihoming for the ge-1/2/1 PE1 interface.

```
[edit interfaces]
user@PE1# set ge-1/2/1 esi 00:33:33:33:33:33:33:33
user@PE1# set ge-1/2/1 esi all-active
```

- f. Configure a trunk interface on the bridge network for the ge-1/2/1 PE1 interface.

```
[edit interfaces]
user@PE1# set ge-1/2/1 unit 0 encapsulation vlan-bridge
user@PE1# set ge-1/2/1 unit 0 vlan-id 30
user@PE1# set ge-1/2/1 unit 130 family bridge interface-mode trunk
user@PE1# set ge-1/2/1 unit 130 family bridge vlan-id-list 130
```

```

user@PE1# set ge-1/2/1 unit 230 family bridge interface-mode trunk
user@PE1# set ge-1/2/1 unit 230 family bridge vlan-id-list 230

```

5. Configure Router PE1 interfaces toward Router PE2.

- a. Assign the interfaces ge-1/0/5 and ge-1/1/6 within the ae12 aggregated bundle.

```

[edit interfaces]
user@PE1# set ge-1/0/5 gigether-options 802.3ad ae12
user@PE1# set ge-1/1/6 gigether-options 802.3ad ae12

```

- b. Specify the minimum number of links for the ae12 aggregated bundle to be labeled “up”.

```

[edit interfaces]
user@PE1# set ae12 aggregated-ether-options minimum-links 1

```

- c. Assign an IP address for the ae12 aggregated bundle and enable MPLS and IS-IS protocol families on the bundle.

```

[edit interfaces]
user@PE1# set ae12 unit 0 family inet address 198.51.100.12/24
user@PE1# set ae12 unit 0 family iso
user@PE1# set ae12 unit 0 family mpls

```

- d. Configure the VLAN tagging and encapsulation parameters for the ae12 aggregated bundle and assign VLAN ID 1200 for the bundle.

```

[edit interfaces]
user@PE1# set ae12 flexible-vlan-tagging
user@PE1# set ae12 encapsulation flexible-ethernet-services
user@PE1# set ae12 unit 0 vlan-id 1200

```

6. Configure Router PE1 interfaces toward Router PE3.

- a. Assign the interfaces ge-1/0/3 and ge-1/0/4 within the ae13 aggregated bundle.

```
[edit interfaces]
user@PE1# set ge-1/0/3 gigether-options 802.3ad ae13
user@PE1# set ge-1/0/4 gigether-options 802.3ad ae13
```

- b. Specify the minimum number of links for the ae13 aggregated bundle to be labeled “up”.

```
[edit interfaces]
user@PE1# set ae13 aggregated-ether-options minimum-links 1
```

- c. Assign an IP address for the ae13 aggregated bundle and enable MPLS and IS-IS protocol families on the bundle.

```
[edit interfaces]
user@PE1# set ae13 unit 0 family inet address 198.51.100.13/24
user@PE1# set ae13 unit 0 family iso
user@PE1# set ae13 unit 0 family mpls
```

- d. Configure the VLAN tagging and encapsulation parameters for the ae12 aggregated bundle and assign the inner and outer VLAN tags for the bundle.

```
[edit interfaces]
user@PE1# set ae13 flexible-vlan-tagging
user@PE1# set ae13 encapsulation flexible-ethernet-services
user@PE1# set ae13 unit 0 vlan-tags outer 1300
user@PE1# set ae13 unit 0 vlan-tags inner 13
```

7. Configure the Router PE1 interface toward the single-homed customer site, Router CE1.

- a. Configure the VLAN tagging and encapsulation parameters for the ge-1/3/0 PE1 interface.

```
[edit interfaces]
user@PE1# set ge-1/3/0 flexible-vlan-tagging
user@PE1# set ge-1/3/0 encapsulation flexible-ethernet-services
```

- b. Configure a trunk interface on the bridge network for the ge-1/3/0 PE1 interface.

```
[edit interfaces]
user@PE1# set ge-1/3/0 unit 0 encapsulation vlan-bridge
user@PE1# set ge-1/3/0 unit 0 vlan-id 10
user@PE1# set ge-1/3/0 unit 100 family bridge interface-mode trunk
user@PE1# set ge-1/3/0 unit 100 family bridge vlan-id-list 110
user@PE1# set ge-1/3/0 unit 100 family bridge vlan-id-list 120
user@PE1# set ge-1/3/0 unit 100 family bridge vlan-id-list 130
user@PE1# set ge-1/3/0 unit 200 family bridge interface-mode trunk
user@PE1# set ge-1/3/0 unit 200 family bridge vlan-id-list 210
user@PE1# set ge-1/3/0 unit 200 family bridge vlan-id-list 220
user@PE1# set ge-1/3/0 unit 200 family bridge vlan-id-list 230
```

8. Configure the Router PE1 interface toward Router P (RR) and enable the MPLS and IS-IS protocol families for the interface.

```
[edit interfaces]
user@PE1# set ge-1/1/4 unit 0 family inet address 192.0.2.1/24
user@PE1# set ge-1/1/4 unit 0 family iso
user@PE1# set ge-1/1/4 unit 0 family mpls
```

9. Configure an IRB interface for Router PE1.

```
[edit interfaces]
user@PE1# set irb unit 0 family inet address 192.0.2.9/24
user@PE1# set irb unit 0 mac 00:99:99:99:01:99
user@PE1# set irb unit 1 family inet address 192.0.2.10/24
user@PE1# set irb unit 1 mac 00:99:99:99:02:99
user@PE1# set irb unit 2 family inet address 192.0.2.11/24
user@PE1# set irb unit 2 mac 00:99:99:99:03:99
user@PE1# set irb unit 10 family inet address 192.0.2.12/24
user@PE1# set irb unit 10 mac 00:99:99:99:01:90
```

10. Configure the loopback interface for Router PE1.

```
[edit interfaces]
user@PE1# set lo0 unit 0 family inet address 198.51.100.1/24 primary
user@PE1# set lo0 unit 0 family iso
```

11. Assign a router ID and the autonomous system number for Router PE1.

```
[edit routing-options]
user@PE1# set router-id 198.51.100.1
user@PE1# set autonomous-system 65221
```

12. Assign a load-balancing policy to the forwarding table of Router PE1.

```
[edit routing-options]
user@PE1# set forwarding-table export load-balancing-policy
```

13. Configure IS-IS on Router PE1.

```
[edit protocols]
user@PE1# set isis level 1 disable
user@PE1# set isis interface all level 2 metric 10
user@PE1# set isis interface fxp0.0 disable
user@PE1# set isis interface lo0.0 level 2 metric 0
```

14. Configure an internal BGP group for Router PE1 to peer with route reflector, Router P.

```
[edit protocols]
user@PE1# set bgp group RR type internal
user@PE1# set bgp group RR local-address 198.51.100.1
user@PE1# set bgp group RR neighbor 203.0.113.0
```

15. Enable EVPN signaling for the RR BGP group on Router PE1.

```
[edit protocols]
user@PE1# set bgp group RR family evpn signaling
```

16. Configure RSVP, LDP, MPLS, EVPN, and L2 learning on Router PE1.

```
[edit protocols]
user@PE1# set rsvp interface all
user@PE1# set rsvp interface fxp0.0 disable
user@PE1# set ldp deaggregate
user@PE1# set ldp interface all
user@PE1# set ldp interface fxp0.0 disable
user@PE1# set mpls interface all
user@PE1# set mpls interface fxp0.0 disable
user@PE1# set evpn
user@PE1# set l2-learning global-mac-table-aging-time 18000
```

17. Configure label-switched paths between the PE routers.

```
[edit protocols]
user@PE1# set mpls label-switched-path pe1tope2 from 198.51.100.1
user@PE1# set mpls label-switched-path pe1tope2 to 198.51.100.2
user@PE1# set mpls label-switched-path pe1tope2 primary direct_to_pe2
user@PE1# set mpls label-switched-path pe1tope3 from 198.51.100.1
user@PE1# set mpls label-switched-path pe1tope3 to 198.51.100.3
user@PE1# set mpls label-switched-path pe1tope3 primary direct_to_pe3
user@PE1# set mpls label-switched-path pe1tope4 from 198.51.100.1
user@PE1# set mpls label-switched-path pe1tope4 to 198.51.100.4
user@PE1# set mpls label-switched-path pe1tope4 primary direct_to_pe4
```

18. Configure MPLS paths from Router PE1 to other PE routers.

```
user@PE1# set mpls path pe4_to_pe3 198.51.100.4 strict
user@PE1# set mpls path pe4_to_pe3 198.51.100.3 strict
user@PE1# set mpls path direct_to_pe2 198.51.100.5 strict
user@PE1# set mpls path direct_to_pe3 198.51.100.6 strict
user@PE1# set mpls path direct_to_pe4 198.51.100.9 strict
user@PE1# set mpls path pe2_to_pe3 198.51.100.2 strict
user@PE1# set mpls path pe2_to_pe3 198.51.100.3 strict
```

19. Configure the load-balancing policy to enable load balancing per packet.

```
[edit policy-options]
user@PE1# set policy-statement load-balancing-policy then load-balance per-packet
```

20. Configure the first virtual switch routing instance.

- a. Configure the routing-instance type and assign Router PE1 interfaces to the routing instance.

```
[edit routing-instances]
user@PE1# set VS-1 instance-type virtual-switch
user@PE1# set VS-1 interface ge-1/2/1.130
user@PE1# set VS-1 interface ge-1/2/2.120
user@PE1# set VS-1 interface ge-1/3/0.100
user@PE1# set VS-1 interface ae0.110
```

- b. Configure the route distinguisher and the VPN routing and forwarding (VRF) target for the VS-1 routing instance.

```
[edit routing-instances]
user@PE1# set VS-1 route-distinguisher 198.51.100.1:101
user@PE1# set VS-1 vrf-target target:100:101
```

- c. Configure EVPN and assign VLANs to the VS-1 routing instance.

```
[edit routing-instances]
user@PE1# set VS-1 protocols evpn extended-vlan-list 110
user@PE1# set VS-1 protocols evpn extended-vlan-list 120
user@PE1# set VS-1 protocols evpn extended-vlan-list 130
user@PE1# set VS-1 protocols evpn default-gateway do-not-advertise
```

- d. Configure the bridge domains and their associated VLANs and IRB interfaces for the VS-1 routing instance.

```
[edit routing-instances]
user@PE1# set VS-1 bridge-domains bd-110 vlan-id 110
user@PE1# set VS-1 bridge-domains bd-110 routing-interface irb.0
user@PE1# set VS-1 bridge-domains bd-120 vlan-id 120
```

```

user@PE1# set VS-1 bridge-domains bd-120 routing-interface irb.1
user@PE1# set VS-1 bridge-domains bd-130 vlan-id 130
user@PE1# set VS-1 bridge-domains bd-130 routing-interface irb.2

```

21. Configure the second virtual switch routing instance.

- a.** Configure the routing-instance type and assign Router PE1 interfaces to the routing instance.

```

[edit routing-instances]
user@PE1# set VS-2 instance-type virtual-switch
user@PE1# set VS-2 interface ge-1/2/1.230
user@PE1# set VS-2 interface ge-1/2/2.220
user@PE1# set VS-2 interface ge-1/3/0.200
user@PE1# set VS-2 interface ae0.210

```

- b.** Configure the route distinguisher and the VPN routing and forwarding (VRF) target for the VS-2 routing instance.

```

[edit routing-instances]
user@PE1# set VS-2 route-distinguisher 198.51.100.1:201
user@PE1# set VS-2 vrf-target target:100:201

```

- c.** Configure EVPN and assign VLANs to the VS-2 routing instance.

```

[edit routing-instances]
user@PE1# set VS-2 protocols evpn extended-vlan-list 210
user@PE1# set VS-2 protocols evpn extended-vlan-list 220
user@PE1# set VS-2 protocols evpn extended-vlan-list 230

```

- d.** Configure the bridge domains and their associated VLANs for the VS-2 routing instance.

```

[edit routing-instances]
user@PE1# set routing-instances VS-2 bridge-domains bd-a vlan-id-list 210
user@PE1# set routing-instances VS-2 bridge-domains bd-a vlan-id-list 220
user@PE1# set routing-instances VS-2 bridge-domains bd-a vlan-id-list 230

```

22. Configure the multihomed EVPN routing instance.

- a. Configure the routing-instance type and assign VLANs and Router PE1 interfaces to the routing instance.

```
[edit routing-instances]
user@PE1# set mhevpn instance-type evpn
user@PE1# set mhevpn vlan-id 10
user@PE1# set mhevpn interface ge-1/2/1.0
user@PE1# set mhevpn interface ge-1/2/2.0
user@PE1# set mhevpn interface ge-1/3/0.0
user@PE1# set mhevpn interface ae0.0
user@PE1# set mhevpn routing-interface irb.10
```

- b. Configure the route distinguisher and the VPN routing and forwarding (VRF) target for the mhevpn routing instance.

```
[edit routing-instances]
user@PE1# set mhevpn route-distinguisher 198.51.100.1:1
user@PE1# set mhevpn vrf-target target:100:1
```

- c. Configure EVPN to the mhevpn routing instance.

```
[edit routing-instances]
user@PE1# set routing-instances mhevpn protocols evpn default-gateway do-not-advertise
```

23. Configure the default routing instance.

- a. Configure the routing-instance type and assign IRB interfaces to the routing instance.

```
[edit routing-instances]
user@PE1# set vrf instance-type vrf
user@PE1# set vrf interface irb.0
user@PE1# set vrf interface irb.1
user@PE1# set vrf interface irb.2
user@PE1# set vrf interface irb.10
```

- b. Configure the route distinguisher and the VPN routing and forwarding (VRF) target for the vrf routing instance.

```
[edit routing-instances]
user@PE1# set vrf route-distinguisher 198.51.100.1:11
user@PE1# set vrf vrf-target target:100:11
user@PE1# set vrf vrf-table-label
```

Configuration on EX9200 Switches

Step-by-Step Procedure

Several configuration statements used to configure active-active mode differ on EX9200 switches from those used on MX Series routers. This procedure shows which configuration statements are specific to EX9200 switches. All other configuration in this example applies both to EX9200 switches and MX Series routers.

1. To configure a trunk interface, include the family `ethernet-switching` statements instead of the family `bridge` statements in all occurrences.

```
[edit interfaces]
user@PE#1# set interfaces ge-1/2/1 unit 130 family ethernet-switching interface-mode trunk
```

2. To configure the Layer 2 Ethernet switching domain, include the `vlan members` (*vlan-id | name*) statement instead of the `vlan-id-list` *vlan-id* statement in all occurrences.

```
[edit interfaces]
user@PE#1# set interfaces ge-1/2/1 unit 130 family ethernet-switching vlan members 130
```

3. To configure the VLAN domain and associated VLANs for each routing instance, include the `vlands` *name* statement, instead of the `bridge-domains` statement in all occurrences.

```
[edit]
user@PE#1# set routing-instances VS-1 vlands bd-110 vlan-id 110
```


4. To configure the IRB interface in each routing instance, include the `l3-interface irb-interface-name` statement instead of the `routing-interface` statement in all occurrences.

```
[edit]
user@PE#1# set routing-instances VS-1 vlans bd-110 l3-interface irb.0
```

Results

From configuration mode, confirm your configuration by entering the `show chassis`, `show interfaces`, `show routing-options`, `show protocols`, and `show routing-instances` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1 show chassis
aggregated-devices {
  ethernet {
    device-count 20;
  }
}
network-services enhanced-ip;
```

```
user@PE1 show interfaces
ge-1/0/3 {
  gigether-options {
    802.3ad ae13;
  }
}
ge-1/0/4 {
  gigether-options {
    802.3ad ae13;
  }
}
ge-1/0/5 {
  gigether-options {
    802.3ad ae12;
  }
}
ge-1/1/4 {
  unit 0 {
    family inet {
```

```

        address 192.0.2.1/24;
    }
    family iso;
    family mpls;
}
}
ge-1/1/6 {
    together-options {
        802.3ad ae12;
    }
}
ge-1/2/1 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    esi {
        00:33:33:33:33:33:33:33:33;
        all-active;
    }
    unit 0 {
        encapsulation vlan-bridge;
        vlan-id 30;
    }
    unit 130 {
        family bridge {
            interface-mode trunk;
            vlan-id-list 130;
        }
    }
    unit 230 {
        family bridge {
            interface-mode trunk;
            vlan-id-list 230;
        }
    }
}
ge-1/2/2 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    esi {
        00:22:22:22:22:22:22:22:22;
        all-active;
    }
    unit 0 {

```

```

        encapsulation vlan-bridge;
        vlan-id 20;
    }
    unit 120 {
        family bridge {
            interface-mode trunk;
            vlan-id-list 120;
        }
    }
    unit 220 {
        family bridge {
            interface-mode trunk;
            vlan-id-list 220;
        }
    }
}
ge-1/2/3 {
    gigether-options {
        802.3ad ae0;
    }
}
ge-1/2/4 {
    gigether-options {
        802.3ad ae0;
    }
}
ge-1/3/0 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 0 {
        encapsulation vlan-bridge;
        vlan-id 10;
    }
    unit 100 {
        family bridge {
            interface-mode trunk;
            vlan-id-list [ 110 120 130 ];
        }
    }
    unit 200 {
        family bridge {
            interface-mode trunk;
            vlan-id-list [ 210 220 230 ];
        }
    }
}

```

```

    }
  }
}
ae0 {
  flexible-vlan-tagging;
  encapsulation flexible-ethernet-services;
  esi {
    00:11:11:11:11:11:11:11:11:11;
    all-active;
  }
  unit 0 {
    encapsulation vlan-bridge;
    vlan-id 10;
  }
  unit 110 {
    family bridge {
      interface-mode trunk;
      vlan-id-list 110;
    }
  }
  unit 210 {
    family bridge {
      interface-mode trunk;
      vlan-id-list 210;
    }
  }
}
ae12 {
  flexible-vlan-tagging;
  encapsulation flexible-ethernet-services;
  aggregated-ether-options {
    minimum-links 1;
  }
  unit 0 {
    vlan-id 1200;
    family inet {
      address 198.51.100.12/24;
    }
    family iso;
    family mpls;
  }
}
ae13 {

```

```

flexible-vlan-tagging;
encapsulation flexible-ethernet-services;
aggregated-ether-options {
    minimum-links 1;
}
unit 0 {
    vlan-tags outer 1300 inner 13;
    family inet {
        address 198.51.100.13/24;
    }
    family iso;
    family mpls;
}
}
irb {
    unit 0 {
        family inet {
            address 192.0.2.9/24;
        }
        mac 00:99:99:99:01:99;
    }
    unit 1 {
        family inet {
            address 192.0.2.10/24;
        }
        mac 00:99:99:99:02:99;
    }
    unit 2 {
        family inet {
            address 192.0.2.11/24;
        }
        mac 00:99:99:99:03:99;
    }
    unit 10 {
        family inet {
            address 192.0.2.12/24;
        }
        mac 00:99:99:99:01:90;
    }
}
lo0 {
    unit 0 {
        family inet {

```

```

        address 198.51.100.1/24 {
            primary;
        }
    }
    family iso;
}

```

```

user@PE1# show routing-options
router-id 198.51.100.1;
autonomous-system 65221;
forwarding-table {
    export load-balancing-policy;
}

```

```

user@PE1# show protocols
rsvp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
mpls {
    label-switched-path pe1tope2 {
        from 198.51.100.1;
        to 198.51.100.2;
        primary direct_to_pe2;
    }
    label-switched-path pe1tope3 {
        from 198.51.100.1;
        to 198.51.100.3;
        primary direct_to_pe3;
    }
    label-switched-path pe1tope4 {
        from 198.51.100.1;
        to 198.51.100.4;
        primary direct_to_pe4;
    }
    path pe4_to_pe3 {
        198.51.100.4 strict;
    }
}

```

```

        198.51.100.3 strict;
    }
    path direct_to_pe2 {
        198.51.100.5 strict;
    }
    path direct_to_pe3 {
        198.51.100.6 strict;
    }
    path direct_to_pe4 {
        198.51.100.9 strict;
    }
    path pe2_to_pe3 {
        198.51.100.2 strict;
        198.51.100.3 strict;
    }
    interface all;
    interface fxp0.0 {
        disable;
    }
}
bgp {
    group RR {
        type internal;
        local-address 198.51.100.1;
        family evpn {
            signaling;
        }
        neighbor 203.0.113.0;
    }
}
isis {
    level 1 disable;
    interface all {
        level 2 metric 10;
    }
    interface fxp0.0 {
        disable;
    }
    interface lo0.0 {
        level 2 metric 0;
    }
}
ldp {

```

```

    deaggregate;
    interface all;
    interface fxp0.0 {
        disable;
    }
}
evpn {
}
l2-learning {
    global-mac-table-aging-time 18000;
}

```

```

user@PE1# show policy-options
policy-statement load-balancing-policy {
    then {
        load-balance per-packet;
    }
}

```

```

user@PE1# show routing-instances
VS-1 {
    instance-type virtual-switch;
    interface ge-1/2/1.130;
    interface ge-1/2/2.120;
    interface ge-1/3/0.100;
    interface ae0.110;
    route-distinguisher 198.51.100.1:101;
    vrf-target target:100:101;
    protocols {
        evpn {
            extended-vlan-list [ 110 120 130 ];
            default-gateway do-not-advertise;
        }
    }
    bridge-domains {
        bd-110 {
            vlan-id 110;
            routing-interface irb.0;
        }
        bd-120 {

```



```

        vlan-id 120;
        routing-interface irb.1;
    }
    bd-130 {
        vlan-id 130;
        routing-interface irb.2;
    }
}
VS-2 {
    instance-type virtual-switch;
    interface ge-1/2/1.230;
    interface ge-1/2/2.220;
    interface ge-1/3/0.200;
    interface ae0.210;
    route-distinguisher 198.51.100.1:201;
    vrf-target target:100:201;
    protocols {
        evpn {
            extended-vlan-list [ 210 220 230 ];
        }
    }
    bridge-domains {
        bd-a {
            vlan-id-list [ 210 220 230 ];
        }
    }
}
mhevpn {
    instance-type evpn;
    vlan-id 10;
    interface ge-1/2/1.0;
    interface ge-1/2/2.0;
    interface ge-1/3/0.0;
    interface ae0.0;
    routing-interface irb.10;
    route-distinguisher 198.51.100.1:1;
    vrf-target target:100:1;
    protocols {
        evpn {
            default-gateway do-not-advertise;
        }
    }
}

```

```
}  
vrf {  
    instance-type vrf;  
    interface irb.0;  
    interface irb.1;  
    interface irb.2;  
    interface irb.10;  
    route-distinguisher 198.51.100.1:11;  
    vrf-target target:100:11;  
    vrf-table-label;  
}
```

Verification

IN THIS SECTION

- [Verifying VPN Services in the Core | 255](#)
- [Verifying the EVPN Instance Status | 258](#)
- [Verifying the Autodiscovery Routes per Ethernet Segment | 265](#)
- [Verifying the Ethernet Segment Route | 268](#)
- [Verifying the DF Status | 270](#)
- [Verifying the BDF Status | 272](#)
- [Verifying the Remote IRB and Host IP | 273](#)

Confirm that the configuration is working properly.

Verifying VPN Services in the Core

Purpose

Ensure that the protocols in the VPN core are functioning properly.

Action

From operational mode, enter the `show isis adjacency` command.

```
user@PE1> show isis adjacency
```

Interface	System	L State	Hold (secs)	SNPA
ge-1/2/4.0		CE10	2 Up	24 5c:5e:ab:e:6f:4

From operational mode, enter the `show bgp summary` command.

```
user@PE1> show bgp summary
```

Groups: 1 Peers: 1 Down peers: 0

Table	Tot Paths	Act Paths	Suppressed	History	Damp	State	Pending
bgp.evpn.0	45	45	0	0	0	0	0

Peer AS InPkt OutPkt OutQ Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...

203.0.113.0	65221	90	26	0	0	3:18	Establ
-------------	-------	----	----	---	---	------	--------

bgp.evpn.0: 45/45/45/0
VS-1.evpn.0: 19/19/19/0
VS-2.evpn.0: 19/19/19/0
mhvevpn.evpn.0: 13/13/13/0
__default_evpn__.evpn.0: 4/4/4/0

```
user@P> show bgp summary
```

Groups: 1 Peers: 4 Down peers: 0

Table	Tot Paths	Act Paths	Suppressed	History	Damp	State	Pending
bgp.evpn.0	68	68	0	0	0	0	0

Peer AS InPkt OutPkt OutQ Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...

198.51.100.1	65221	25	90	0	0	3:04	Establ
198.51.100.2	65221	32	80	0	0	6:12	Establ
198.51.100.3	65221	31	62	0	0	6:58	Establ
198.51.100.4	65221	28	88	0	0	6:04	Establ

bgp.evpn.0: 22/22/22/0
bgp.evpn.0: 22/22/22/0
bgp.evpn.0: 12/12/12/0
bgp.evpn.0: 12/12/12/0

From operational mode, enter the `show mpls lsp` command.

```
user@PE1> show mpls lsp
Ingress LSP: 3 sessions
```

To	From	State	Rt	P	ActivePath	LSPname
198.51.100.2	198.51.100.1	Up	0	*	direct_to_pe2	pe1tope2
198.51.100.3	198.51.100.1	Up	0	*	direct_to_pe3	pe1tope3
198.51.100.4	198.51.100.1	Up	0	*	direct_to_pe4	pe1tope4

Total 3 displayed, Up 3, Down 0


```
Egress LSP: 3 sessions
```

To	From	State	Rt	Style	Labelin	Labelout	LSPname
198.51.100.1	198.51.100.3	Up	0	1 FF	3	-	pe3tope1
198.51.100.1	198.51.100.4	Up	0	1 FF	3	-	pe4tope1
198.51.100.1	198.51.100.2	Up	0	1 FF	3	-	pe2tope1

Total 3 displayed, Up 3, Down 0


```
Transit LSP: 0 sessions
Total 0 displayed, Up 0, Down 0
```

From operational mode, enter the `show interface ae* terse` command.

```
user@PE1> show interface ae* terse
```

Interface	Admin	Link	Proto	Local	Remote
ae0	up	up			
ae0.0	up	up	bridge		
ae0.110	up	up	bridge		
ae0.210	up	up	bridge		
ae0.32767	up	up	multiservice		
ae12	up	up			
ae12.0	up	up	inet	198.51.100.12/24	
			iso		
			mpls		
			multiservice		
ae12.32767	up	up	multiservice		
ae13	up	up			
ae13.0	up	up	inet	198.51.100.13/24	
			iso		
			mpls		
			multiservice		
ae13.32767	up	up	multiservice		

Meaning

The protocols IS-IS, BGP and MPLS are up and running. The aggregated bundles configured on Router PE1 are up.

Verifying the EVPN Instance Status

Purpose

Verify the EVPN routing instances and their status.

Action

From operational mode, run the `show evpn instance extensive` command.

```

user@PE1> show evpn instance extensive
Instance: VS-1
  Route Distinguisher: 198.51.100.1:101
  Per-instance MAC route label: 301664
  MAC database status
    Local Remote
    Total MAC addresses:      0      0
    Default gateway MAC addresses: 3      0
  Number of local interfaces: 4 (3 up)
    Interface name  ESI                                     Mode                Status
    ae0.110         00:11:11:11:11:11:11:11:11:11:11:11:11 all-active          Up
    ge-1/2/1.130    00:33:33:33:33:33:33:33:33:33:33:33 all-active          Up
    ge-1/2/2.120    00:22:22:22:22:22:22:22:22:22:22:22 all-active          Up
    ge-1/3/0.100    00:00:00:00:00:00:00:00:00:00:00:00 single-homed        Up
  Number of IRB interfaces: 3 (3 up)
    Interface name  VLAN ID  Status  L3 context
    irb.0           110      Up      vrf
    irb.1           120      Up      vrf
    irb.2           130      Up      vrf
  Number of bridge domains: 3
    VLAN ID  Intfs / up  Mode                MAC sync  IM route label
    110      2 1          Extended          Enabled   301984
    120      2 1          Extended          Enabled   302000
    130      2 1          Extended          Enabled   302016
  Number of neighbors: 3
    198.51.100.2
    Received routes
    MAC address advertisement:      0

```

```

    MAC+IP address advertisement:      0
    Inclusive multicast:                3
    Ethernet auto-discovery:           6
198.51.100.3
Received routes
    MAC address advertisement:         0
    MAC+IP address advertisement:      0
    Inclusive multicast:                1
    Ethernet auto-discovery:           2
198.51.100.4
Received routes
    MAC address advertisement:         0
    MAC+IP address advertisement:      0
    Inclusive multicast:                3
    Ethernet auto-discovery:           2
Number of ethernet segments: 4
ESI: 00:11:11:11:11:11:11:11:11
Status: Resolved by IFL ae0.110
Local interface: ae0.110, Status: Up/Forwarding
Number of remote PEs connected: 2
  Remote PE      MAC label  Aliasing label  Mode
  198.51.100.3   0          305584          all-active
  198.51.100.2   0          306000          all-active
Designated forwarder: 198.51.100.3
Backup forwarder: 198.51.100.1
Backup forwarder: 198.51.100.2
Advertised MAC label: 301792
Advertised aliasing label: 301792
Advertised split horizon label: 301808
ESI: 00:22:22:22:22:22:22:22:22
Status: Resolved by IFL ge-1/2/2.120
Local interface: ge-1/2/2.120, Status: Up/Forwarding
Number of remote PEs connected: 1
  Remote PE      MAC label  Aliasing label  Mode
  198.51.100.2   0          306032          all-active
Designated forwarder: 198.51.100.1
Backup forwarder: 198.51.100.2
Advertised MAC label: 301824
Advertised aliasing label: 301824
Advertised split horizon label: 301840
ESI: 00:33:33:33:33:33:33:33:33
Status: Resolved by IFL ge-1/2/1.130
Local interface: ge-1/2/1.130, Status: Up/Forwarding

```

Number of remote PEs connected: 1

Remote PE	MAC label	Aliasing label	Mode
198.51.100.2	0	306064	all-active

Designated forwarder: 198.51.100.1

Backup forwarder: 198.51.100.2

Advertised MAC label: 301856

Advertised aliasing label: 301856

Advertised split horizon label: 301872

ESI: 00:44:44:44:44:44:44:44:44:44

Status: Resolved by NH 1048613

Number of remote PEs connected: 1

Remote PE	MAC label	Aliasing label	Mode
198.51.100.4	0	305152	all-active

Instance: VS-2

Route Distinguisher: 198.51.100.1:201

Per-instance MAC route label: 301696

MAC database status	Local	Remote
Total MAC addresses:	0	0
Default gateway MAC addresses:	0	0

Number of local interfaces: 4 (3 up)

Interface name	ESI	Mode	Status
ae0.210	00:11:11:11:11:11:11:11:11:11	all-active	Up
ge-1/2/1.230	00:33:33:33:33:33:33:33:33:33	all-active	Up
ge-1/2/2.220	00:22:22:22:22:22:22:22:22:22	all-active	Up
ge-1/3/0.200	00:00:00:00:00:00:00:00:00:00	single-homed	Down

Number of IRB interfaces: 0 (0 up)

Number of bridge domains: 3

VLAN ID	Intfs / up	Mode	MAC sync	IM route label
210	2 1	Extended	Enabled	302032
220	2 1	Extended	Enabled	302048
230	2 1	Extended	Enabled	302064

Number of neighbors: 3

198.51.100.2

Received routes

MAC address advertisement:	0
MAC+IP address advertisement:	0
Inclusive multicast:	3
Ethernet auto-discovery:	6

198.51.100.3

Received routes

MAC address advertisement:	0
MAC+IP address advertisement:	0

```

    Inclusive multicast:          1
    Ethernet auto-discovery:      2

```

```
198.51.100.4
```

```
Received routes
```

```

    MAC address advertisement:    0
    MAC+IP address advertisement: 0
    Inclusive multicast:          3
    Ethernet auto-discovery:      2

```

```
Number of ethernet segments: 4
```

```
ESI: 00:11:11:11:11:11:11:11
```

```
Status: Resolved by IFL ae0.210
```

```
Local interface: ae0.210, Status: Up/Forwarding
```

```
Number of remote PEs connected: 2
```

Remote PE	MAC label	Aliasing label	Mode
198.51.100.3	0	305648	all-active
198.51.100.2	0	306096	all-active

```
Designated forwarder: 198.51.100.1
```

```
Backup forwarder: 198.51.100.2
```

```
Backup forwarder: 198.51.100.3
```

```
Advertised MAC label: 301888
```

```
Advertised aliasing label: 301888
```

```
Advertised split horizon label: 301808
```

```
ESI: 00:22:22:22:22:22:22:22
```

```
Status: Resolved by IFL ge-1/2/2.220
```

```
Local interface: ge-1/2/2.220, Status: Up/Forwarding
```

```
Number of remote PEs connected: 1
```

Remote PE	MAC label	Aliasing label	Mode
198.51.100.2	0	306112	all-active

```
Designated forwarder: 198.51.100.1
```

```
Backup forwarder: 198.51.100.2
```

```
Advertised MAC label: 301904
```

```
Advertised aliasing label: 301904
```

```
Advertised split horizon label: 301840
```

```
ESI: 00:33:33:33:33:33:33:33
```

```
Status: Resolved by IFL ge-1/2/1.230
```

```
Local interface: ge-1/2/1.230, Status: Up/Forwarding
```

```
Number of remote PEs connected: 1
```

Remote PE	MAC label	Aliasing label	Mode
198.51.100.2	0	306128	all-active

```
Designated forwarder: 198.51.100.1
```

```
Backup forwarder: 198.51.100.2
```

```
Advertised MAC label: 301920
```

```
Advertised aliasing label: 301920
```



```

    Advertised split horizon label: 301872
    ESI: 00:44:44:44:44:44:44:44:44
    Status: Resolved by NH 1048616
    Number of remote PEs connected: 1
      Remote PE      MAC label  Aliasing label  Mode
      198.51.100.4   0          305184          all-active

```

Instance: __default_evpn__

Route Distinguisher: 198.51.100.1:0

VLAN ID: None

Per-instance MAC route label: 301760

```

MAC database status          Local  Remote
Total MAC addresses:         0      0
Default gateway MAC addresses: 0      0

```

Number of local interfaces: 0 (0 up)

Number of IRB interfaces: 0 (0 up)

Number of bridge domains: 0

Number of neighbors: 2

198.51.100.2

Received routes

```

    Ethernet auto-discovery:      0
    Ethernet Segment:             3

```

198.51.100.3

Received routes

```

    Ethernet auto-discovery:      0
    Ethernet Segment:             1

```

Number of ethernet segments: 0

Instance: mhevpn

Route Distinguisher: 198.51.100.1:1

VLAN ID: 10

Per-instance MAC route label: 301728

```

MAC database status          Local  Remote
Total MAC addresses:         0      0
Default gateway MAC addresses: 1      0

```

Number of local interfaces: 4 (3 up)

Interface name	ESI	Mode	Status
ae0.0	00:11:11:11:11:11:11:11:11	all-active	Up
ge-1/2/1.0	00:33:33:33:33:33:33:33:33	all-active	Up
ge-1/2/2.0	00:22:22:22:22:22:22:22:22	all-active	Up
ge-1/3/0.0	00:00:00:00:00:00:00:00:00	single-homed	Down

Number of IRB interfaces: 1 (1 up)

Interface name	VLAN ID	Status	L3 context
----------------	---------	--------	------------

```

    irb.10      10      Up      vrf
Number of bridge domains: 1
  VLAN ID  Intfs / up  Mode      MAC sync  IM route label
    10      4 3      Extended  Enabled   302080
Number of neighbors: 3
  198.51.100.2
    Received routes
      MAC address advertisement:      0
      MAC+IP address advertisement:   0
      Inclusive multicast:             1
      Ethernet auto-discovery:        6
  198.51.100.3
    Received routes
      MAC address advertisement:      0
      MAC+IP address advertisement:   0
      Inclusive multicast:             1
      Ethernet auto-discovery:        2
  198.51.100.4
    Received routes
      MAC address advertisement:      0
      MAC+IP address advertisement:   0
      Inclusive multicast:             1
      Ethernet auto-discovery:        2
Number of ethernet segments: 4
  ESI: 00:11:11:11:11:11:11:11:11
    Status: Resolved by IFL ae0.0
    Local interface: ae0.0, Status: Up/Forwarding
    Number of remote PEs connected: 2
      Remote PE      MAC label  Aliasing label  Mode
      198.51.100.3   0          305680          all-active
      198.51.100.2   0          306144          all-active
    Designated forwarder: 198.51.100.2
    Backup forwarder: 198.51.100.1
    Backup forwarder: 198.51.100.3
    Advertised MAC label: 301936
    Advertised aliasing label: 301936
    Advertised split horizon label: 301808
  ESI: 00:22:22:22:22:22:22:22:22
    Status: Resolved by IFL ge-1/2/2.0
    Local interface: ge-1/2/2.0, Status: Up/Forwarding
    Number of remote PEs connected: 1
      Remote PE      MAC label  Aliasing label  Mode
      198.51.100.2   0          306160          all-active

```

```

Designated forwarder: 198.51.100.1
Backup forwarder: 198.51.100.2
Advertised MAC label: 301952
Advertised aliasing label: 301952
Advertised split horizon label: 301840
ESI: 00:33:33:33:33:33:33:33:33:33
Status: Resolved by IFL ge-1/2/1.0
Local interface: ge-1/2/1.0, Status: Up/Forwarding
Number of remote PEs connected: 1
  Remote PE      MAC label  Aliasing label  Mode
  198.51.100.2   0          306176         all-active
Designated forwarder: 198.51.100.1
Backup forwarder: 198.51.100.2
Advertised MAC label: 301968
Advertised aliasing label: 301968
Advertised split horizon label: 301872
ESI: 00:44:44:44:44:44:44:44:44:44
Status: Resolved by NH 1048612
Number of remote PEs connected: 1
  Remote PE      MAC label  Aliasing label  Mode
  198.51.100.4   0          305200         all-active

```

Meaning

The output provides the following information:

- List of EVPN and virtual switch routing instances
- Mode of operation of each interface
- Neighbors of each routing instance
- Number of different routes received from each neighbor
- ESI attached to each routing instance
- Number of Ethernet segments on each routing instance
- DF election roles for each ESI in an EVI
- VLAN ID and MAC labels for each routing instance
- IRB interface details

- Number of default gateway MAC addresses received for the virtual switch routing instance (VS-1 and VS-2)

Verifying the Autodiscovery Routes per Ethernet Segment

Purpose

Verify that the autodiscovery routes per Ethernet segment are received.

Action

From operational mode, run the `show route table mhevpn.evpn.0` command.

Router PE1

```
user@PE1> show route table mhevpn.evpn.0
mhevpn.evpn.0: 17 destinations, 17 routes (17 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:198.51.100.1:1::111111111111111111::0/304
    *[EVPN/170] 00:11:37
    Indirect
1:198.51.100.1:1::222222222222222222::0/304
    *[EVPN/170] 00:11:37
    Indirect
1:198.51.100.1:1::333333333333333333::0/304
    *[EVPN/170] 00:11:37
    Indirect
1:198.51.100.2:0::111111111111111111::FFFF:FFFF/304
    *[BGP/170] 00:11:33, localpref 100, from 203.0.113.0
    AS path: I, validation-state: unverified
    > to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
1:198.51.100.2:0::222222222222222222::FFFF:FFFF/304
    *[BGP/170] 00:11:33, localpref 100, from 203.0.113.0
    AS path: I, validation-state: unverified
    > to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
1:198.51.100.2:0::333333333333333333::FFFF:FFFF/304
    *[BGP/170] 00:11:33, localpref 100, from 203.0.113.0
    AS path: I, validation-state: unverified
    > to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
1:198.51.100.2:1::111111111111111111::0/304
    *[BGP/170] 00:11:33, localpref 100, from 203.0.113.0
```

```

        AS path: I, validation-state: unverified
        > to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
1:198.51.100.2:1::22222222222222222222::0/304
        *[BGP/170] 00:11:33, localpref 100, from 203.0.113.0
        AS path: I, validation-state: unverified
        > to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
1:198.51.100.2:1::33333333333333333333::0/304
        *[BGP/170] 00:11:33, localpref 100, from 203.0.113.0
        AS path: I, validation-state: unverified
        > to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
1:198.51.100.3:0::11111111111111111111::FFFF:FFFF/304
        *[BGP/170] 00:11:37, localpref 100, from 203.0.113.0
        AS path: I, validation-state: unverified
        > to 198.51.100.6 via ae13.0, label-switched-path pe1tope3
1:198.51.100.3:1::11111111111111111111::0/304
        *[BGP/170] 00:11:37, localpref 100, from 203.0.113.0
        AS path: I, validation-state: unverified
        > to 198.51.100.6 via ae13.0, label-switched-path pe1tope3
3:198.51.100.1:1::10::198.51.100.1/304
        *[EVPN/170] 00:13:38
        Indirect
3:198.51.100.2:1::10::198.51.100.2/304
        *[BGP/170] 00:11:33, localpref 100, from 203.0.113.0
        AS path: I, validation-state: unverified
        > to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
3:198.51.100.3:1::10::198.51.100.3/304
        *[BGP/170] 00:11:37, localpref 100, from 203.0.113.0
        AS path: I, validation-state: unverified
        > to 198.51.100.6 via ae13.0, label-switched-path pe1tope3

```

Router PE2

```

user@PE2> show route table mhevpn.evpn.0
mhevpn.evpn.0: 17 destinations, 17 routes (17 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:198.51.100.1:0::11111111111111111111::FFFF:FFFF/304
        *[BGP/170] 01:10:26, localpref 100, from 203.0.113.0
        AS path: I, validation-state: unverified
        > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
1:198.51.100.1:0::22222222222222222222::FFFF:FFFF/304
        *[BGP/170] 01:10:26, localpref 100, from 203.0.113.0

```

```

        AS path: I, validation-state: unverified
        > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
1:198.51.100.1:0::33333333333333333333::FFFF:FFFF/304
        *[BGP/170] 01:10:26, localpref 100, from 203.0.113.0
        AS path: I, validation-state: unverified
        > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
1:198.51.100.1:1::11111111111111111111::0/304
        *[BGP/170] 01:10:26, localpref 100, from 203.0.113.0
        AS path: I, validation-state: unverified
        > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
1:198.51.100.1:1::22222222222222222222::0/304
        *[BGP/170] 01:10:26, localpref 100, from 203.0.113.0
        AS path: I, validation-state: unverified
        > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
1:198.51.100.1:1::33333333333333333333::0/304
        *[BGP/170] 01:10:26, localpref 100, from 203.0.113.0
        AS path: I, validation-state: unverified
        > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
1:198.51.100.2:1::11111111111111111111::0/304
        *[EVPN/170] 01:10:26
        Indirect
1:198.51.100.2:1::22222222222222222222::0/304
        *[EVPN/170] 01:10:26
        Indirect
1:198.51.100.2:1::33333333333333333333::0/304
        *[EVPN/170] 01:10:26
        Indirect
1:198.51.100.3:0::11111111111111111111::FFFF:FFFF/304
        *[BGP/170] 01:10:26, localpref 100, from 203.0.113.0
        AS path: I, validation-state: unverified
        > to 198.51.100.7 via ge-0/0/4.0, label-switched-path pe2tope3
1:198.51.100.3:1::11111111111111111111::0/304
        *[BGP/170] 01:10:26, localpref 100, from 203.0.113.0
        AS path: I, validation-state: unverified
        > to 198.51.100.7 via ge-0/0/4.0, label-switched-path pe2tope3
1:198.51.100.4:0::44444444444444444444::FFFF:FFFF/304
        *[BGP/170] 01:10:17, localpref 100, from 203.0.113.0
        AS path: I, validation-state: unverified
        > to 198.51.100.8 via ge-0/1/8.0, label-switched-path pe2tope4
1:198.51.100.4:1::44444444444444444444::0/304
        *[BGP/170] 01:10:17, localpref 100, from 203.0.113.0
        AS path: I, validation-state: unverified
        > to 198.51.100.8 via ge-0/1/8.0, label-switched-path pe2tope4

```

```

3:198.51.100.1:1::10::198.51.100.1/304
    *[BGP/170] 01:10:26, localpref 100, from 203.0.113.0
    AS path: I, validation-state: unverified
    > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
3:198.51.100.2:1::10::198.51.100.2/304
    *[EVPN/170] 01:12:14
    Indirect
3:198.51.100.3:1::10::198.51.100.3/304
    *[BGP/170] 01:10:26, localpref 100, from 203.0.113.0
    AS path: I, validation-state: unverified
    > to 198.51.100.7 via ge-0/0/4.0, label-switched-path pe2tope3
3:198.51.100.4:1::10::198.51.100.4/304
    *[BGP/170] 01:10:17, localpref 100, from 203.0.113.0
    AS path: I, validation-state: unverified
    > to 198.51.100.8 via ge-0/1/8.0, label-switched-path pe2tope4

```

Meaning

The remote type 1 autodiscovery route is received for the ESI attached to Router PE2, which is the other PE router connected to the multihomed CE device.

Verifying the Ethernet Segment Route

Purpose

Verify that the local and advertised autodiscovery routes per Ethernet segment and the Ethernet segment routes are received.

Action

From operational mode, run the `show route table __default_evpn__.evpn.0` command.

Router PE1

```

user@PE1> show route table __default_evpn__.evpn.0
__default_evpn__.evpn.0: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:198.51.100.1:0::111111111111111111::FFFF:FFFF/304
    *[EVPN/170] 00:25:18
    Indirect

```

```

1:198.51.100.1:0::222222222222222222::FFFF:FFFF/304
    *[EVPN/170] 00:25:18
    Indirect
1:198.51.100.1:0::333333333333333333::FFFF:FFFF/304
    *[EVPN/170] 00:25:18
    Indirect
4:198.51.100.1:0::111111111111111111:198.51.100.1/304
    *[EVPN/170] 00:25:18
    Indirect
4:198.51.100.1:0::222222222222222222:198.51.100.1/304
    *[EVPN/170] 00:25:18
    Indirect
4:198.51.100.1:0::333333333333333333:198.51.100.1/304
    *[EVPN/170] 00:25:18
    Indirect
4:198.51.100.2:0::111111111111111111:198.51.100.2/304
    *[BGP/170] 00:25:14, localpref 100, from 203.0.113.0
    AS path: I, validation-state: unverified
    > to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
4:198.51.100.2:0::222222222222222222:198.51.100.2/304
    *[BGP/170] 00:25:14, localpref 100, from 203.0.113.0
    AS path: I, validation-state: unverified
    > to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
4:198.51.100.2:0::333333333333333333:198.51.100.2/304
    *[BGP/170] 00:25:14, localpref 100, from 203.0.113.0
    AS path: I, validation-state: unverified
    > to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
4:198.51.100.3:0::111111111111111111:198.51.100.3/304
    *[BGP/170] 00:25:18, localpref 100, from 203.0.113.0
    AS path: I, validation-state: unverified
    > to 198.51.100.6 via ae13.0, label-switched-path pe1tope3

```

Router PE2

```

user@PE2> show route table __default_evpn__.evpn.0
__default_evpn__.evpn.0: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:198.51.100.2:0::111111111111111111::FFFF:FFFF/304
    *[EVPN/170] 01:17:59
    Indirect
1:198.51.100.2:0::222222222222222222::FFFF:FFFF/304

```



```

*[EVPN/170] 01:17:59
    Indirect
1:198.51.100.2:0::33333333333333333333::FFFF:FFFF/304
*[EVPN/170] 01:17:59
    Indirect
4:198.51.100.1:0::11111111111111111111:198.51.100.1/304
*[BGP/170] 01:17:59, localpref 100, from 203.0.113.0
    AS path: I, validation-state: unverified
    > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
4:198.51.100.1:0::22222222222222222222:198.51.100.1/304
*[BGP/170] 01:17:59, localpref 100, from 203.0.113.0
    AS path: I, validation-state: unverified
    > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
4:198.51.100.1:0::33333333333333333333:198.51.100.1/304
*[BGP/170] 01:17:59, localpref 100, from 203.0.113.0
    AS path: I, validation-state: unverified
    > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
4:198.51.100.2:0::11111111111111111111:198.51.100.2/304
*[EVPN/170] 01:17:59
    Indirect
4:198.51.100.2:0::22222222222222222222:198.51.100.2/304
*[EVPN/170] 01:17:59
    Indirect
4:198.51.100.2:0::33333333333333333333:198.51.100.2/304
*[EVPN/170] 01:17:59
    Indirect
4:198.51.100.3:0::11111111111111111111:198.51.100.3/304
*[BGP/170] 01:17:59, localpref 100, from 203.0.113.0
    AS path: I, validation-state: unverified
    > to 198.51.100.7 via ge-0/0/4.0, label-switched-path pe2tope3

```

Meaning

The output displays the local and remote type 1 (autodiscovery) and type 4 (Ethernet segment) routes.

Verifying the DF Status

Purpose

Confirm which PE router is the designated forwarder (DF) for each routing instance.

Action

From operational mode, run the `show evpn instance designated-forwarder` command.

```
user@PE1> show evpn instance designated forwarder
Instance: VS-1
  Number of ethernet segments: 4
    ESI: 00:11:11:11:11:11:11:11:11:11
      Designated forwarder: 198.51.100.3
    ESI: 00:22:22:22:22:22:22:22:22:22
      Designated forwarder: 198.51.100.1
    ESI: 00:33:33:33:33:33:33:33:33:33
      Designated forwarder: 198.51.100.1
    ESI: 00:44:44:44:44:44:44:44:44:44
      Designated forwarder: No local attachment to ethernet segment

Instance: VS-2
  Number of ethernet segments: 4
    ESI: 00:11:11:11:11:11:11:11:11:11
      Designated forwarder: 198.51.100.1
    ESI: 00:22:22:22:22:22:22:22:22:22
      Designated forwarder: 198.51.100.1
    ESI: 00:33:33:33:33:33:33:33:33:33
      Designated forwarder: 198.51.100.1
    ESI: 00:44:44:44:44:44:44:44:44:44
      Designated forwarder: No local attachment to ethernet segment

Instance: mhevpn
  Number of ethernet segments: 4
    ESI: 00:11:11:11:11:11:11:11:11:11
      Designated forwarder: 198.51.100.2
    ESI: 00:22:22:22:22:22:22:22:22:22
      Designated forwarder: 198.51.100.1
    ESI: 00:33:33:33:33:33:33:33:33:33
      Designated forwarder: 198.51.100.1
    ESI: 00:44:44:44:44:44:44:44:44:44
      Designated forwarder: No local attachment to ethernet segment
```

Meaning

The designated forwarder is displayed for each routing instance and ESI.

Verifying the BDF Status

Purpose

Confirm which PE router is the backup designated forwarder (BDF) for each routing instance.

Action

From operational mode, run the `show evpn instance backup-forwarder` command.

```

user@PE1> show evpn instance backup-forwarder
Instance: VS-1
  Number of ethernet segments: 4
    ESI: 00:11:11:11:11:11:11:11:11
      Backup forwarder: 198.51.100.1
      Backup forwarder: 198.51.100.2
    ESI: 00:22:22:22:22:22:22:22:22
      Backup forwarder: 198.51.100.2
    ESI: 00:33:33:33:33:33:33:33:33
      Backup forwarder: 198.51.100.2
    ESI: 00:44:44:44:44:44:44:44:44
      Backup forwarder: No local attachment to ethernet segment

Instance: VS-2
  Number of ethernet segments: 4
    ESI: 00:11:11:11:11:11:11:11:11
      Backup forwarder: 198.51.100.2
      Backup forwarder: 198.51.100.3
    ESI: 00:22:22:22:22:22:22:22:22
      Backup forwarder: 198.51.100.2
    ESI: 00:33:33:33:33:33:33:33:33
      Backup forwarder: 198.51.100.2
    ESI: 00:44:44:44:44:44:44:44:44
      Backup forwarder: No local attachment to ethernet segment

Instance: mhevpn
  Number of ethernet segments: 4
    ESI: 00:11:11:11:11:11:11:11:11
      Backup forwarder: 198.51.100.1
      Backup forwarder: 198.51.100.3
    ESI: 00:22:22:22:22:22:22:22:22
      Backup forwarder: 198.51.100.2

```

```

ESI: 00:33:33:33:33:33:33:33:33:33
Backup forwarder: 198.51.100.2
ESI: 00:44:44:44:44:44:44:44:44:44
Backup forwarder: No local attachment to ethernet segment

```

Meaning

The backup designated forwarder is displayed for each routing instance and ESI.

Verifying the Remote IRB and Host IP

Purpose

Verify that the remote IRB IP and the host IP are received.

Action

Router PE1

From operational mode, run the `show route table mhevpn` command.

```

user@PE1> show route table mhevpn
mhevpn.evpn.0: 17 destinations, 17 routes (17 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:198.51.100.1:1::11111111111111111111::0/304
    *[EVPN/170] 01:34:26
    Indirect
1:198.51.100.1:1::22222222222222222222::0/304
    *[EVPN/170] 01:34:26
    Indirect
1:198.51.100.1:1::33333333333333333333::0/304
    *[EVPN/170] 01:34:26
    Indirect
1:198.51.100.2:0::11111111111111111111::FFFF:FFFF/304
    *[BGP/170] 01:34:22, localpref 100, from 203.0.113.0
    AS path: I, validation-state: unverified
    > to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
1:198.51.100.2:0::22222222222222222222::FFFF:FFFF/304
    *[BGP/170] 01:34:22, localpref 100, from 203.0.113.0
    AS path: I, validation-state: unverified

```

```

> to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
1:198.51.100.2:0::333333333333333333333333::FFFF:FFFF/304
*[BGP/170] 01:34:22, localpref 100, from 203.0.113.0
AS path: I, validation-state: unverified
> to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
1:198.51.100.2:1::111111111111111111111111::0/304
*[BGP/170] 01:34:22, localpref 100, from 203.0.113.0
AS path: I, validation-state: unverified
> to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
1:198.51.100.2:1::222222222222222222222222::0/304
*[BGP/170] 01:34:22, localpref 100, from 203.0.113.0
AS path: I, validation-state: unverified
> to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
1:198.51.100.2:1::333333333333333333333333::0/304
*[BGP/170] 01:34:22, localpref 100, from 203.0.113.0
AS path: I, validation-state: unverified
> to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
1:198.51.100.3:0::111111111111111111111111::FFFF:FFFF/304
*[BGP/170] 01:34:26, localpref 100, from 203.0.113.0
AS path: I, validation-state: unverified
> to 198.51.100.6 via ae13.0, label-switched-path pe1tope3
1:198.51.100.3:1::111111111111111111111111::0/304
*[BGP/170] 01:34:26, localpref 100, from 203.0.113.0
AS path: I, validation-state: unverified
> to 198.51.100.6 via ae13.0, label-switched-path pe1tope3
3:198.51.100.1:1::10::198.51.100.1/304
*[EVPN/170] 01:36:27
Indirect
3:198.51.100.2:1::10::198.51.100.2/304
*[BGP/170] 01:34:22, localpref 100, from 203.0.113.0
AS path: I, validation-state: unverified
> to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
3:198.51.100.3:1::10::198.51.100.3/304
*[BGP/170] 01:34:26, localpref 100, from 203.0.113.0
AS path: I, validation-state: unverified
> to 198.51.100.6 via ae13.0, label-switched-path pe1tope3

```

Router PE2

From operational mode, run the `show route table mhevpn` command.

```

user@PE2> show route table mhevpn
mhevpn.evpn.0: 17 destinations, 17 routes (17 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:198.51.100.1:0::11111111111111111111::FFFF:FFFF/304
    *[BGP/170] 01:35:11, localpref 100, from 203.0.113.0
    AS path: I, validation-state: unverified
    > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
1:198.51.100.1:0::22222222222222222222::FFFF:FFFF/304
    *[BGP/170] 01:35:11, localpref 100, from 203.0.113.0
    AS path: I, validation-state: unverified
    > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
1:198.51.100.1:0::33333333333333333333::FFFF:FFFF/304
    *[BGP/170] 01:35:11, localpref 100, from 203.0.113.0
    AS path: I, validation-state: unverified
    > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
1:198.51.100.1:1::11111111111111111111::0/304
    *[BGP/170] 01:35:11, localpref 100, from 203.0.113.0
    AS path: I, validation-state: unverified
    > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
1:198.51.100.1:1::22222222222222222222::0/304
    *[BGP/170] 01:35:11, localpref 100, from 203.0.113.0
    AS path: I, validation-state: unverified
    > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
1:198.51.100.1:1::33333333333333333333::0/304
    *[BGP/170] 01:35:11, localpref 100, from 203.0.113.0
    AS path: I, validation-state: unverified
    > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
1:198.51.100.2:1::11111111111111111111::0/304
    *[EVPN/170] 01:35:11
    Indirect
1:198.51.100.2:1::22222222222222222222::0/304
    *[EVPN/170] 01:35:11
    Indirect
1:198.51.100.2:1::33333333333333333333::0/304
    *[EVPN/170] 01:35:11
    Indirect
1:198.51.100.3:0::11111111111111111111::FFFF:FFFF/304
    *[BGP/170] 01:35:11, localpref 100, from 203.0.113.0
    AS path: I, validation-state: unverified

```

```

> to 198.51.100.7 via ge-0/0/4.0, label-switched-path pe2tope3
1:198.51.100.3:1::11111111111111111111::0/304
*[BGP/170] 01:35:11, localpref 100, from 203.0.113.0
AS path: I, validation-state: unverified
> to 198.51.100.7 via ge-0/0/4.0, label-switched-path pe2tope3
1:198.51.100.4:0::44444444444444444444::FFFF:FFFF/304
*[BGP/170] 01:35:02, localpref 100, from 203.0.113.0
AS path: I, validation-state: unverified
> to 198.51.100.8 via ge-0/1/8.0, label-switched-path pe2tope4
1:198.51.100.4:1::44444444444444444444::0/304
*[BGP/170] 01:35:02, localpref 100, from 203.0.113.0
AS path: I, validation-state: unverified
> to 198.51.100.8 via ge-0/1/8.0, label-switched-path pe2tope4
3:198.51.100.1:1::10::198.51.100.1/304
*[BGP/170] 01:35:11, localpref 100, from 203.0.113.0
AS path: I, validation-state: unverified
> to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
3:198.51.100.2:1::10::198.51.100.2/304
*[EVPN/170] 01:36:59
Indirect
3:198.51.100.3:1::10::198.51.100.3/304
*[BGP/170] 01:35:11, localpref 100, from 203.0.113.0
AS path: I, validation-state: unverified
> to 198.51.100.7 via ge-0/0/4.0, label-switched-path pe2tope3
3:198.51.100.4:1::10::198.51.100.4/304
*[BGP/170] 01:35:02, localpref 100, from 203.0.113.0
AS path: I, validation-state: unverified
> to 198.51.100.8 via ge-0/1/8.0, label-switched-path pe2tope4

```

Router PE3

From operational mode, run the show route table mhevpn command.

```

user@PE3> show route table mhevpn
mhevpn.evpn.0: 19 destinations, 19 routes (19 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:198.51.100.1:0::11111111111111111111::FFFF:FFFF/304
*[BGP/170] 01:36:10, localpref 100, from 203.0.113.0
AS path: I, validation-state: unverified
> to 198.51.100.13 via ae13.0, label-switched-path pe3tope1
1:198.51.100.1:0::22222222222222222222::FFFF:FFFF/304

```

```

*[BGP/170] 01:36:10, localpref 100, from 203.0.113.0
  AS path: I, validation-state: unverified
  > to 198.51.100.13 via ae13.0, label-switched-path pe3tope1
1:198.51.100.1:0::33333333333333333333333333333333::FFFF:FFFF/304
*[BGP/170] 01:36:10, localpref 100, from 203.0.113.0
  AS path: I, validation-state: unverified
  > to 198.51.100.13 via ae13.0, label-switched-path pe3tope1
1:198.51.100.1:1::11111111111111111111111111111111::0/304
*[BGP/170] 01:36:10, localpref 100, from 203.0.113.0
  AS path: I, validation-state: unverified
  > to 198.51.100.13 via ae13.0, label-switched-path pe3tope1
1:198.51.100.1:1::22222222222222222222222222222222::0/304
*[BGP/170] 01:36:10, localpref 100, from 203.0.113.0
  AS path: I, validation-state: unverified
  > to 198.51.100.13 via ae13.0, label-switched-path pe3tope1
1:198.51.100.1:1::33333333333333333333333333333333::0/304
*[BGP/170] 01:36:10, localpref 100, from 203.0.113.0
  AS path: I, validation-state: unverified
  > to 198.51.100.13 via ae13.0, label-switched-path pe3tope1
1:198.51.100.2:0::11111111111111111111111111111111::FFFF:FFFF/304
*[BGP/170] 01:36:06, localpref 100, from 203.0.113.0
  AS path: I, validation-state: unverified
  > to 198.51.100.10 via ge-2/0/3.0, label-switched-path pe3tope2
1:198.51.100.2:0::22222222222222222222222222222222::FFFF:FFFF/304
*[BGP/170] 01:36:06, localpref 100, from 203.0.113.0
  AS path: I, validation-state: unverified
  > to 198.51.100.10 via ge-2/0/3.0, label-switched-path pe3tope2
1:198.51.100.2:0::33333333333333333333333333333333::FFFF:FFFF/304
*[BGP/170] 01:36:06, localpref 100, from 203.0.113.0
  AS path: I, validation-state: unverified
  > to 198.51.100.10 via ge-2/0/3.0, label-switched-path pe3tope2
1:198.51.100.2:1::11111111111111111111111111111111::0/304
*[BGP/170] 01:36:06, localpref 100, from 203.0.113.0
  AS path: I, validation-state: unverified
  > to 198.51.100.10 via ge-2/0/3.0, label-switched-path pe3tope2
1:198.51.100.2:1::22222222222222222222222222222222::0/304
*[BGP/170] 01:36:06, localpref 100, from 203.0.113.0
  AS path: I, validation-state: unverified
  > to 198.51.100.10 via ge-2/0/3.0, label-switched-path pe3tope2
1:198.51.100.2:1::33333333333333333333333333333333::0/304
*[BGP/170] 01:36:06, localpref 100, from 203.0.113.0
  AS path: I, validation-state: unverified
  > to 198.51.100.10 via ge-2/0/3.0, label-switched-path pe3tope2

```



```

1:198.51.100.3:1::11111111111111111111::0/304
    *[EVPN/170] 01:36:25
    Indirect
1:198.51.100.4:0::44444444444444444444::FFFF:FFFF/304
    *[BGP/170] 01:35:58, localpref 100, from 203.0.113.0
    AS path: I, validation-state: unverified
    > to 198.51.100.11 via ge-2/1/3.0, label-switched-path pe3tope4
1:198.51.100.4:1::44444444444444444444::0/304
    *[BGP/170] 01:35:58, localpref 100, from 203.0.113.0
    AS path: I, validation-state: unverified
    > to 198.51.100.11 via ge-2/1/3.0, label-switched-path pe3tope4
3:198.51.100.1:1::10::198.51.100.1/304
    *[BGP/170] 01:36:10, localpref 100, from 203.0.113.0
    AS path: I, validation-state: unverified
    > to 198.51.100.13 via ae13.0, label-switched-path pe3tope1
3:198.51.100.2:1::10::198.51.100.2/304
    *[BGP/170] 01:36:06, localpref 100, from 203.0.113.0
    AS path: I, validation-state: unverified
    > to 198.51.100.10 via ge-2/0/3.0, label-switched-path pe3tope2
3:198.51.100.3:1::10::198.51.100.3/304
    *[EVPN/170] 01:37:33
    Indirect
3:198.51.100.4:1::10::198.51.100.4/304
    *[BGP/170] 01:35:58, localpref 100, from 203.0.113.0
    AS path: I, validation-state: unverified
    > to 198.51.100.11 via ge-2/1/3.0, label-switched-path pe3tope4

```

Meaning

The output displays the local and remote IRB interfaces. It also displays the local and remote hosts that are installed in the VRF table.

Release History Table

Release	Description
16.1R4	Starting with Junos OS Release 16.1R4, EVPN multihoming active-active mode is supported on all EX9200 switches.

Example: Configuring LACP for EVPN Active-Active Multihoming

IN THIS SECTION

- [Requirements | 279](#)
- [Overview | 280](#)
- [Configuration | 283](#)
- [Verification | 296](#)

This example shows how to configure the Link Aggregation Control Protocol (LACP) on multihomed customer edge (CE) and provider edge (PE) devices in an Ethernet VPN (EVPN) active-active multihomed network.

Requirements

This example uses the following hardware and software components:

- Four MX Series 5G Universal Routing Platforms with MPC interfaces only, of which:
 - Two routers are configured as PE devices connected to a common multihomed customer site.
 - One router is configured as a remote PE device connected to a single-homed customer site.
 - One router is configured as the core provider device.
- Two CE devices, of which:
 - One CE device is multihomed to the two PE devices.
 - One CE device is single-homed to the remote PE device.
- Junos OS Release 17.1 or later running on all the PE devices.

Before you begin:

1. Configure the device interfaces and enable MPLS on all the interfaces of the PE devices.
2. Configure OSPF or any other IGP between the core provider device and the PE devices.
3. Configure MPLS and a label-switched path (LSP) from the ingress PE device to the remote egress PE device.
4. Configure an internal BGP session between the PE devices.

NOTE: We recommend that you configure Bidirectional Forwarding Detection (BFD) on the BGP session for faster detection of core isolation and better convergence.

Overview

IN THIS SECTION

- [Topology | 281](#)

Starting with Junos OS Release 17.1, an extra level of redundancy can be achieved in an EVPN active-active multihoming environment by configuring LACP on both the endpoints of the multihomed CE-PE link. The multihomed devices are configured with aggregated trunk links, where the link aggregation group (LAG) interfaces of the CE-PE link can either be in the active or in the standby state. When the LAG interface is in the active state, data traffic is transmitted over the CE-PE link. When the LAG interface is in the standby state, data traffic is blocked and only control traffic for communicating LAG interface state is transmitted over the link.

The LAG interface state is monitored and operated by LACP to ensure fast convergence on isolation of a multihomed PE device from the core. When there is a core failure, a null route can occur at the isolated PE device. However, with the support for LACP on the CE-PE link, at the time of core isolation, the CE-facing interface of the multihomed PE device is set to the standby state, thereby blocking data traffic transmission from and toward the multihomed CE device. After the core recovers from the failure, the interface state is switched back from standby to active.

The support for LACP for EVPN active-active multihoming is applicable to both EVPN with MPLS and EVPN with VXLAN.

When LACP is configured on the CE-PE link, isolation of the multihomed PE device from the core is handled as follows:

1. When the EVPN BGP peers are null for a multihomed PE device, the PE device is isolated from the core. The CE devices connected to the isolated PE device are notified about the core failure by the isolated PE device.
2. On learning about the core failure, data traffic is not forwarded from the CE device to the isolated multihomed PE device. Instead, the traffic is diverted to the other multihomed PE devices that belong to the same LAG. This helps in preventing traffic black holes at the isolated PE device.
3. If the multihomed CE device uses the LAG for load balancing traffic to multiple active multihomed PE devices, then the LACP configuration along with the same system ID configured on all the

multihomed PE devices for that given LAG, triggers an LACP out-of-sync to all the attached multihomed CE links.

When configuring LACP on the multihomed devices, be aware of the following considerations:

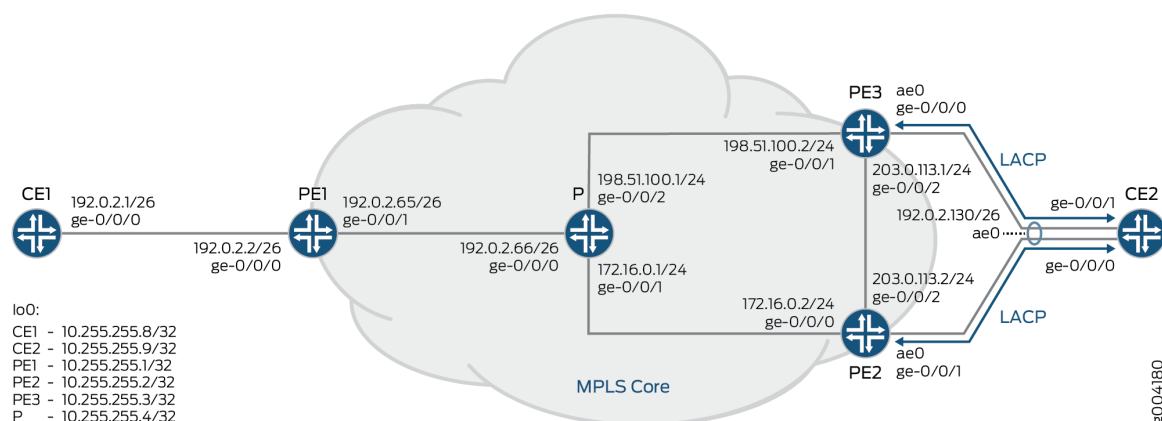
- The LAG link can operate either in the active or in the standby state regardless of the UP/DOWN operational state.
- On system reboot, the LACP link on the multihomed PE device is in the active state.
- When the control plane goes down or when the BGP-EVPN session is down, LACP is notified to run multiplexer state machine for the aggregation port and move the link from active to standby.
- An interface is not treated as up unless it operates in the active state and its operational state is also up.
- The following features are not supported with LACP on EVPN active-active multihoming:
 - Ethernet segment identifier (ESI) value auto-derivation from LACP system ID.
 - Redundancy coverage for non-LAG interfaces.
 - Redundancy coverage for core isolation with static LAGs.
 - Node isolation.
 - Access failures for LACP with EVPN active-active multihoming.

Topology

[Figure 13 on page 282](#) illustrates an EVPN active-active multihoming network with LACP configured on the multihomed CE and PE devices. Device CE1 is single-homed and is connected to a remote PE device, PE1. Device PE2 and Device PE3 connect to a common multihomed CE—CE2. Device P is the core device to which all the PE devices (PE1, PE2, and PE3) are connected.

The endpoints of the multihomed devices—Device CE2, Device PE2, and Device PE3—are configured with LACP on the CE-PE link.

Figure 13: LACP Support in EVPN Active-Active Multihoming



Core isolation of Device PE3 is handled as follows:

1. After LACP is configured on the multihomed CE-PE links, the LACP configuration and operational data is synchronized between the LACP peers to operate as a LAG.

The synchronization between the LACP peers happens with the exchange of control PDUs, and is required for the following reasons:

- To determine the state of the links in the Ethernet bundle—all-active or standby.
- To detect and handle CE device misconfiguration when LACP Port Key is configured on the PE device.
- To detect and handle miswiring between CE and PE devices when LACP Port Key is configured on the PE device.
- To detect and react to actor or partner churn when the LACP speakers are not able to converge.

2. After Device PE2 and Device PE3 establish BGP session with at least one peer, the state of the CE-PE link is set to unblocking mode by LACP.
3. When there is a core failure, and the BGP EVPN peers of Device PE2 becomes null, Device PE2 is isolated from the core. This can cause a null route at Device PE2. To prevent this situation, the LAG interface of Device PE2 that is facing Device CE2 is changed from the active state to the standby state by LACP.
4. An out-of-sync notification is sent from LACP on the attached multihomed CE2 link to block traffic transmission between Device CE2 and Device PE2.

5. When the control plane recovers, that is when Device PE2 establishes BGP session with other EVPN PEs, the LAG interface of Device PE2 is switched back from standby to active by LACP.

Configuration

IN THIS SECTION

- [CLI Quick Configuration | 283](#)
- [Procedure | 288](#)
- [Procedure | 294](#)

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the [edit] hierarchy level, and then enter `commit` from configuration mode.

Device CE1

```
set interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 mac 00:00:00:00:00:01
set interfaces ge-0/0/0 unit 0 vlan-id 100
set interfaces ge-0/0/0 unit 0 family inet address 192.0.2.1/26
set interfaces lo0 unit 0 family inet address 10.255.255.8/32
```

Device CE2

```
set chassis aggregated-devices ethernet device-count 1
set interfaces ge-0/0/0 gigether-options 802.3ad ae0
set interfaces ge-0/0/1 gigether-options 802.3ad ae0
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 mac 00:00:00:00:00:03
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 unit 0 vlan-id 100
set interfaces ae0 unit 0 family inet address 192.0.2.130/26
set interfaces lo0 unit 0 family inet address 10.255.255.9/32
```

Device PE1

```

set interfaces ge-0/0/0 unit 0 family inet address 192.0.2.2/26
set interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 encapsulation flexible-ethernet-services
set interfaces ge-0/0/0 unit 0 encapsulation vlan-bridge
set interfaces ge-0/0/0 unit 0 vlan-id 100
set interfaces ge-0/0/1 unit 0 family inet address 192.0.2.65/26
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.255.1/32
set routing-options router-id 10.255.255.1
set routing-options autonomous-system 100
set routing-options forwarding-table chained-composite-next-hop ingress evpn
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols mpls interface ge-0/0/1.0
set protocols bgp local-address 10.255.255.1
set protocols bgp family inet unicast
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 10.255.255.1
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 10.255.255.2
set protocols bgp group ibgp neighbor 10.255.255.3
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
set protocols ldp interface ge-0/0/1.0
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set protocols ldp interface lo0.0
set routing-instances ALPHA instance-type evpn
set routing-instances ALPHA vlan-id 100
set routing-instances ALPHA interface ge-0/0/0.0
set routing-instances ALPHA route-distinguisher 10.255.255.1:100
set routing-instances ALPHA vrf-target target:100:100
set routing-instances ALPHA protocols evpn label-allocation per-instance

```

Device PE2

```

set chassis aggregated-devices ethernet device-count 1
set interfaces ge-0/0/0 unit 0 family inet address 172.16.0.2/24
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 gigether-options 802.3ad ae0
set interfaces ge-0/0/2 unit 0 family inet address 203.0.113.2/24
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ae0 vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 esi 00:11:22:33:44:55:66:77:88:99
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp system-id 00:01:02:03:04:05
set interfaces ae0 unit 0 encapsulation vlan-bridge
set interfaces ae0 unit 0 vlan-id 100
set interfaces lo0 unit 0 family inet address 10.255.255.2/32
set routing-options router-id 10.255.255.2
set routing-options autonomous-system 100
set routing-options forwarding-table chained-composite-next-hop ingress evpn
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp local-address 10.255.255.2
set protocols bgp family inet unicast
set protocols bgp family l2vpn signaling
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 10.255.255.2
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 10.255.255.3
set protocols bgp group ibgp neighbor 10.255.255.1
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set protocols ldp interface lo0.0
set routing-instances ALPHA instance-type evpn
set routing-instances ALPHA vlan-id 100
set routing-instances ALPHA interface ae0.0

```



```
set routing-instances ALPHA route-distinguisher 10.255.255.2:100
set routing-instances ALPHA vrf-target target:100:100
```

Device PE3

```
set chassis aggregated-devices ethernet device-count 1
set interfaces ge-0/0/0 gigether-options 802.3ad ae0
set interfaces ge-0/0/1 unit 0 family inet address 198.51.100.2/24
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/2 unit 0 family inet address 203.0.113.1/24
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ae0 vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 esi 00:11:22:33:44:55:66:77:88:99
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp system-id 00:01:02:03:04:05
set interfaces ae0 unit 0 encapsulation vlan-bridge
set interfaces ae0 unit 0 vlan-id 100
set interfaces lo0 unit 0 family inet address 10.255.255.3/32
set routing-options router-id 10.255.255.3
set routing-options autonomous-system 100
set routing-options forwarding-table chained-composite-next-hop ingress evpn
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp local-address 10.255.255.3
set protocols bgp family inet unicast
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 10.255.255.3
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 10.255.255.2
set protocols bgp group ibgp neighbor 10.255.255.1
set protocols bgp group l2vpn type internal
set protocols bgp group l2vpn family l2vpn signaling
set protocols bgp group l2vpn neighbor 10.255.255.4
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
```

```

set protocols ldp interface lo0.0
set routing-instances ALPHA instance-type evpn
set routing-instances ALPHA vlan-id 100
set routing-instances ALPHA interface ae0.0
set routing-instances ALPHA route-distinguisher 10.255.255.3:100
set routing-instances ALPHA vrf-target target:100:100

```

Device P

```

set interfaces ge-0/0/0 unit 0 family inet address 192.0.2.66/26
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 family inet address 172.16.0.1/24
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/2 unit 0 family inet address 198.51.100.1/24
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.255.4/32
set routing-options router-id 10.255.255.4
set routing-options autonomous-system 100
set protocols rsvp interface all aggregate
set protocols mpls interface ge-0/0/0.0
set protocols mpls interface ge-0/0/1.0
set protocols mpls interface ge-0/0/2.0
set protocols bgp group l2vpn type internal
set protocols bgp group l2vpn local-address 10.255.255.4
set protocols bgp group l2vpn family l2vpn signaling
set protocols bgp group l2vpn neighbor 10.255.255.3
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0
set protocols ldp interface all
set routing-instances vpls1 instance-type vpls
set routing-instances vpls1 route-distinguisher 10.255.255.4:100
set routing-instances vpls1 vrf-target target:200:200
set routing-instances vpls1 protocols vpls no-tunnel-services
set routing-instances vpls1 protocols vpls site vpls-pe site-identifier 3
set routing-instances vpls1 protocols vpls site vpls-pe best-site

```

Procedure

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the [CLI User Guide](#).

To configure Device PE2:

NOTE: Repeat this procedure for other multihomed PE devices after modifying the appropriate interface names, addresses, and other parameters.

1. Specify the number of aggregated Ethernet interfaces to be created on Device PE2.

```
[edit chassis]
user@PE2# set aggregated-devices ethernet device-count 1
```

2. Configure Device PE2 interfaces that connect to Device P and Device PE3, respectively.

```
[edit interfaces]
user@PE2# set ge-0/0/0 unit 0 family inet address 172.16.0.2/24
user@PE2# set ge-0/0/0 unit 0 family iso
user@PE2# set ge-0/0/0 unit 0 family mpls
user@PE2# set ge-0/0/2 unit 0 family inet address 203.0.113.2/24
user@PE2# set ge-0/0/2 unit 0 family iso
user@PE2# set ge-0/0/2 unit 0 family mpls
```

3. Configure Device PE2 interfaces within the ae0 aggregated bundle toward the multihomed Device CE2.

```
[edit interfaces]
user@PE2# set ge-0/0/1 gigether-options 802.3ad ae0
user@PE2# set ae0 vlan-tagging
user@PE2# set ae0 encapsulation flexible-ethernet-services
user@PE2# set ae0 unit 0 encapsulation vlan-bridge
user@PE2# set ae0 unit 0 vlan-id 100
```

4. Configure active-active multihoming on the aggregated Ethernet interface.

```
[edit interfaces]
user@PE2# set ae0 esi 00:11:22:33:44:55:66:77:88:99
user@PE2# set ae0 esi all-active
```

5. Configure LACP on the CE-PE link of Device PE2.

```
[edit interfaces]
user@PE2# set ae0 aggregated-ether-options lacp active
user@PE2# set ae0 aggregated-ether-options lacp system-id 00:01:02:03:04:05
```

6. Configure the loopback interface of Device PE2.

```
[edit interfaces]
user@PE2# set lo0 unit 0 family inet address 10.255.255.2/32
```

7. Configure the router ID and autonomous system number for Device PE2.

```
[edit routing-options]
user@PE2# set router-id 10.255.255.2
user@PE2# set autonomous-system 100
```

8. Enable chained composite next hop for EVPN on Device PE2.

```
[edit routing-options]
user@PE2# set forwarding-table chained-composite-next-hop ingress evpn
```

9. Configure MPLS on all the interfaces of Device PE2, excluding the management interface.

```
[edit protocols]
user@PE2# set mpls interface all
user@PE2# set mpls interface fxp0.0 disable
```

10. Configure an internal BGP group for Device PE1 to peer with the other EVPN PE devices.

```
[edit protocols]
user@PE2# set bgp local-address 10.255.255.2
user@PE2# set bgp family inet unicast
user@PE2# set bgp family l2vpn signaling
user@PE2# set bgp group ibgp type internal
user@PE2# set bgp group ibgp local-address 10.255.255.2
user@PE2# set bgp group ibgp family evpn signaling
user@PE2# set bgp group ibgp neighbor 10.255.255.3
user@PE2# set bgp group ibgp neighbor 10.255.255.1
```

11. Configure OSPF and LDP on all the interfaces of Device PE2, excluding the management interface.

```
[edit protocols]
user@PE2# set ospf area 0.0.0.0 interface all
user@PE2# set ospf area 0.0.0.0 interface fxp0.0 disable
user@PE2# set ospf area 0.0.0.0 interface lo0.0 passive
user@PE2# set ldp interface all
user@PE2# set ldp interface fxp0.0 disable
user@PE2# set ldp interface lo0.0
```

12. Configure an EVPN routing instance on Device PE2 and assign the routing instance parameters.

```
[edit routing-instances]
user@PE2# set ALPHA instance-type evpn
user@PE2# set ALPHA vlan-id 100
user@PE2# set ALPHA interface ae0.0
user@PE2# set ALPHA route-distinguisher 10.255.255.2:100
user@PE2# set ALPHA vrf-target target:100:100
```

Results

From configuration mode, confirm your configuration by entering the `show chassis`, `show interfaces`, `show routing-options`, `show protocols`, and `show routing-instances` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE2# show chassis
aggregated-devices {
  ethernet {
    device-count 1;
  }
}
```

```
user@PE2# show interfaces
ge-0/0/0 {
  unit 0 {
    family inet {
      address 172.16.0.2/24;
    }
    family iso;
    family mpls;
  }
}
ge-0/0/1 {
  gigether-options {
    802.3ad ae0;
  }
}
ge-0/0/2 {
  unit 0 {
    family inet {
      address 203.0.113.2/24;
    }
    family iso;
    family mpls;
  }
}
ae0 {
  vlan-tagging;
  encapsulation flexible-ethernet-services;
  esi {
```

```

        00:11:22:33:44:55:66:77:88:99;
        all-active;
    }
    aggregated-ether-options {
        lacp {
            active;
            system-id 00:01:02:03:04:05;
        }
    }
    unit 0 {
        encapsulation vlan-bridge;
        vlan-id 100;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.255.255.2/32;
        }
    }
}
}

```

```

user@PE2# show routing-options
router-id 10.255.255.2;
autonomous-system 100;
forwarding-table {
    chained-composite-next-hop {
        ingress {
            evpn;
        }
    }
}
}

```

```

user@PE2# show protocols
mpls {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
}

```

```

bgp {
    local-address 10.255.255.2;
    family inet {
        unicast;
    }
    family l2vpn {
        signaling;
    }
    group ibgp {
        type internal;
        local-address 10.255.255.2;
        family evpn {
            signaling;
        }
        neighbor 10.255.255.3;
        neighbor 10.255.255.1;
    }
}

ospf {
    area 0.0.0.0 {
        interface all;
        interface fxp0.0 {
            disable;
        }
        interface lo0.0 {
            passive;
        }
    }
}

ldp {
    interface all;
    interface fxp0.0 {
        disable;
    }
    interface lo0.0;
}

```

```

user@PE2# show routing-instances
ALPHA {
    instance-type evpn;
    vlan-id 100;
}

```



```

interface ae0.0;
route-distinguisher 10.255.255.2:100;
vrf-target target:100:100;
protocols {
    evpn {
        traceoptions {
            file evpn-alpha.txt size 4294967295;
            flag all;
        }
    }
}
}

```

If you are done configuring the device, enter `commit` from configuration mode.

Procedure

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the [CLI User Guide](#).

To configure Device CE2:

NOTE: Repeat this procedure for other multihomed CE devices after modifying the appropriate interface names, addresses, and other parameters.

1. Specify the number of aggregated Ethernet interfaces to be created on Device CE2.

```

[edit chassis]
user@CE2# set aggregated-devices ethernet device-count 1

```

2. Configure Device CE2 interfaces within the ae0 aggregated bundle toward Device PE2 and Device PE3.

```

[edit interfaces]
user@CE2# set ge-0/0/0 gigether-options 802.3ad ae0
user@CE2# set ge-0/0/1 gigether-options 802.3ad ae0
user@CE2# set ae0 flexible-vlan-tagging

```

```

user@CE2# set ae0 encapsulation flexible-ethernet-services
user@E2# set ae0 mac 00:00:00:00:00:03
user@E2# set ae0 unit 0 vlan-id 100
user@CE2# set ae0 unit 0 family inet address 192.0.2.130/26

```

3. Configure LACP on the CE-PE links on Device CE2.

```

[edit interfaces]
user@CE2# set ae0 aggregated-ether-options lacp active

```

4. Configure the loopback interface of Device CE2.

```

[edit interfaces]
user@CE2# set interfaces lo0 unit 0 family inet address 10.255.255.9/32

```

Results

From configuration mode, confirm your configuration by entering the `show chassis`, `show interfaces`, `show routing-options`, `show protocols`, and `show routing-instances` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```

user@CE2# show chassis
aggregated-devices {
  ethernet {
    device-count 1;
  }
}

```

```

user@CE2# show interfaces
ge-0/0/0 {
  gigether-options {
    802.3ad ae0;
  }
}
ge-0/0/1 {
  gigether-options {
    802.3ad ae0;
  }
}

```

```

}
ae0 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    mac 00:00:00:00:00:03;
    aggregated-ether-options {
        lacp {
            active;
        }
    }
    unit 0 {
        vlan-id 100;
        family inet {
            address 192.0.2.130/26;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.255.255.9/32;
        }
    }
}
}

```

If you are done configuring the device, enter `commit` from configuration mode.

Verification

IN THIS SECTION

- [Verifying BGP Session Status | 297](#)
- [Verifying LACP Interface Status of Multihomed PE Device | 297](#)
- [Verifying LACP Interface State of Isolated PE Device | 298](#)
- [Verifying EVPN Routing Instance | 299](#)

Confirm that the configuration is working properly.

Verifying BGP Session Status

Purpose

Verify that Device PE2 has established BGP peering with other EVPN PE devices.

Action

From operational mode, run the **show bgp summary** command.

```

user@PE2> show bgp summary
Groups: 1 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State   Pending
inet.0
                0          0          0          0          0          0
bgp.l2vpn.0
                0          0          0          0          0          0
bgp.evpn.0
                5          5          0          0          0          0
Peer           AS      InPkt    OutPkt    OutQ    Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
10.255.255.1   100      6316     6333      0        0 1d 23:33:03 Establ
  bgp.evpn.0: 1/1/1/0
  ALPHA.evpn.0: 1/1/1/0
  __default_evpn__.evpn.0: 0/0/0/0
10.255.255.3   100      6318     6332      0        0 1d 23:32:56 Establ
  bgp.evpn.0: 4/4/4/0
  ALPHA.evpn.0: 3/3/3/0
  __default_evpn__.evpn.0: 1/1/1/0

```

Meaning

Device PE2 has established BGP peering with the other EVPN PE devices—Device PE1 and Device PE3.

Verifying LACP Interface Status of Multihomed PE Device

Purpose

Verify the LACP interface state on Device PE2.

Action

From operational mode, run the **show lacp interfaces** and **show interfaces ae* terse** commands.

```

user@PE2> show lacp interfaces
Aggregated interface: ae0
  LACP state:      Role  Exp  Def  Dist  Col  Syn  Aggr  Timeout  Activity
    ge-0/0/1      Actor  No   No   Yes  Yes  Yes  Yes    Fast    Active
    ge-0/0/1      Partner No   No   Yes  Yes  Yes  Yes    Fast    Active
  LACP protocol:   Receive State Transmit State      Mux State
    ge-0/0/1              Current  Fast periodic Collecting distributing

```

```

user@PE2> show interfaces ae* terse
Interface      Admin Link Proto  Local      Remote
ae0            up    up
ae0.0          up    up  bridge
ae0.32767      up    up multiservice

```

Meaning

The LACP LAG interface state is active when there is BGP peering with other EVPN PE devices. The physical interface state of the aggregated Ethernet interfaces are up and operational.

Verifying LACP Interface State of Isolated PE Device

Purpose

Verify the LACP interface state of Device PE2 when no BGP EVPN peers have been established.

Action

From operational mode, run the **show lacp interfaces** and **show interfaces ae* terse** commands.

```

user@PE2> show lacp interfaces
Aggregated interface: ae0
  LACP state:      Role  Exp  Def  Dist  Col  Syn  Aggr  Timeout  Activity
    ge-0/0/1      Actor  No   No   No   No   No   Yes    Fast    Active
    ge-0/0/1      Partner No   No   No   No   Yes  Yes    Fast    Active

```

LACP protocol:	Receive State	Transmit State	Mux State
ge-0/0/1	Current	Fast periodic	Waiting

```
user@PE2> show interfaces ae* terse
```

Interface	Admin	Link	Proto	Local	Remote
ae0	up	down			
ae0.0	up	down	bridge		
ae0.32767	up	down	multiservice		

Meaning

Because of core isolation, the LAG interface state is in standby, blocking traffic to-and-from Device CE2. As a result, the physical interface state of the aggregated Ethernet interface is also down until the LACP link state switches back to active on core failure recovery.

Verifying EVPN Routing Instance

Purpose

Verify the EVPN routing instance parameters on Device PE2.

Action

From operational mode, run the **show evpn instance extensive** command.

```
user@PE2> show evpn instance extensive
```

Instance: ALPHA

Route Distinguisher: 10.255.255.2:100

VLAN ID: 100

Per-instance MAC route label: 299776

MAC database status	Local	Remote
MAC advertisements:	0	0
MAC+IP advertisements:	0	0
Default gateway MAC advertisements:	0	0

Number of local interfaces: 1 (1 up)

Interface name	ESI	Mode	Status
ae0.0	00:11:22:33:44:55:66:77:88:99	all-active	Up

Number of IRB interfaces: 0 (0 up)

Number of bridge domains: 1

```

VLAN  Domain ID  Intfs / up  IRB intf  Mode          MAC sync  IM route label
100           1    1              Extended    Enabled    300048
Number of neighbors: 2
Address          MAC    MAC+IP      AD      IM      ES
10.255.255.1      0      0          0      1      0
10.255.255.3      0      0          2      1      0
Number of ethernet segments: 1
ESI: 00:11:22:33:44:55:66:77:88:99
Status: Resolved by IFL ae0.0
Local interface: ae0.0, Status: Up/Forwarding
Number of remote PEs connected: 1
Remote PE      MAC label  Aliasing label  Mode
10.255.255.3    0          299856          all-active
Designated forwarder: 10.255.255.2
Backup forwarder: 10.255.255.3
Last designated forwarder update: Nov 21 01:27:52
Advertised MAC label: 299856
Advertised aliasing label: 299856
Advertised split horizon label: 299968

Instance: __default_evpn__
Route Distinguisher: 10.255.255.2:0
Number of bridge domains: 0
Number of neighbors: 1
Address          MAC    MAC+IP      AD      IM      ES
10.255.255.3      0      0          0      0      1

```

Meaning

The output provides the following information:

- EVPN routing instance
- Mode of operation of each interface
- Neighbors of the routing instance
- Number of different routes received from each neighbor
- ESI attached to the routing instance
- Number of Ethernet segments on the routing instance
- Designated forwarder (DF) election roles for each ESI in an EVPN instance (EVI)

- VLAN ID and MAC labels for the routing instance

RELATED DOCUMENTATION

| [Example: Configuring EVPN Active-Active Multihoming](#) | 213

Example: Configuring LACP for EVPN VXLAN Active-Active Multihoming

IN THIS SECTION

- [Requirements](#) | 301
- [Overview](#) | 301
- [Configuration](#) | 304
- [Verification](#) | 327

This example shows how to configure the Link Aggregation Control Protocol (LACP) on multihomed customer edge (CE) and provider edge (PE) devices in an Ethernet VPN (EVPN) VXLAN active-active multihomed network.

Requirements

This example uses the following hardware and software components:

- Three QFX10002, QFX5100, QFX5110, QFX5200 switches, or QFX5100 Virtual Chassis configured as PE devices, and one QFX5100 switch configured as a CE device.
- Junos OS Release 17.1 or later running on all switches.

Overview

IN THIS SECTION

- [Topology](#) | 303

For another level of redundancy, you can configure EVPN VXLAN active-active multihoming by configuring LACP on both the endpoints of the multihomed CE-PE link. The multihomed devices are configured with aggregated trunk links, where the link aggregation group (LAG) interfaces of the CE-PE link can either be in the active or in the standby state. When the LAG interface is in the active state, data traffic is transmitted over the CE-PE link. When the LAG interface is in the standby state, data traffic is blocked and only control traffic for communicating LAG interface state is transmitted over the link.

LACP monitors and operates the LAG interface to ensure fast convergence on isolation of a multihomed PE device from the core. When there is a core failure, a null route can occur at the isolated PE device. However, with the support for LACP on the CE-PE link, at the time of core isolation, the CE-facing interface of the multihomed PE device is set to the standby state, thereby blocking data traffic transmission from and toward the multihomed PE device. After the core recovers from the failure, the interface state is switched back from standby to active.

NOTE: On QFX10002 and QFX10008 switches, only LACP for EVPN active-active multihoming with VXLAN is supported.

When you configure LACP on the CE-PE link, isolation of the multihomed PE device from the core is handled as follows:

1. The LACP peers synchronize the configuration and operational data.

The LACP peers synchronize by exchanging control PDUs, and is required for the following reasons:

- To determine the state of the links in the Ethernet bundle—all-active or standby.
 - To detect and handle CE device misconfiguration when LACP Port Key is configured on the PE device.
 - To detect and handle miswiring between CE and PE devices when LACP Port Key is configured on the PE device.
 - To detect and react to actor or partner churn when the LACP speakers are not able to converge.
2. When the peers are null for a multihomed PE device, the PE device is isolated from the core. In this case, the isolated PE device notifies the CE devices that are connected to the isolated PE device that there is a core failure.
 3. Data traffic is not forwarded from the CE device to the isolated multihomed PE device. Instead, the traffic is diverted to the other multihomed PE devices that belong to the same LAG. This prevents traffic black holes at the isolated PE device.
 4. If the multihomed CE device uses the LAG for load balancing traffic to multiple active multihomed PE devices, then the LACP configuration along with the same system ID configured on all the

multihomed PE devices for that given LAG, triggers an LACP out-of-sync to all the attached multihomed CE links.

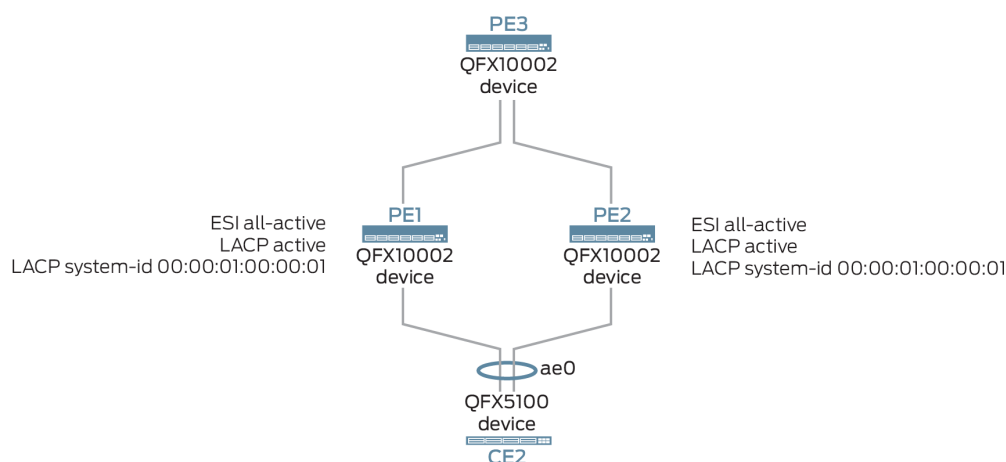
When configuring LACP on the multihomed devices, be aware of the following considerations:

- The LAG link can operate either in the active or in the standby state regardless of the UP/DOWN operational state.
- When you reboot the system, the LACP link on the multihomed PE device is in the active state.
- When the control plane goes down, LACP is notified to run multiplexer state machine for the aggregation port and move the link from active to standby.
- An interface is not treated as up unless it operates in the active state and its operational state is also up.

Topology

Figure 14 on page 303 illustrates an EVPN VXLAN active-active multihoming network with LACP configured on the multi-homed CE and PE devices. Device CE1 is single-homed and is connected to remote PE1 and PE2 devices. Device CE2 is multi-homed to PE1 and PE2 devices.

Figure 14: LACP Support in EVPN Active-Active Multihoming



Core isolation of Device PE1, for example, is handled as follows:

1. After PE2 and PE1 establish a BGP session, LACP sets the state of the CE-PE link to unblocking mode.
2. When there is a core failure, this can cause a null route at Device CE1.

To prevent this situation, the LAG interface that is facing Device CE2 is changed from the active state to the standby state by LACP.

3. LACP sends an out of-sync notification on the attached multihomed CE2 link to block traffic transmission between Device CE2 and Device PE1.
4. When the control plane recovers, Device PE2 is switched back from standby to active by LACP.

Configuration

IN THIS SECTION

- [CLI Quick Configuration | 304](#)
- [Configuring LACP for EVPN Active-Active Multihoming on PE3 | 307](#)
- [Configuring LACP for EVPN Active-Active Multihoming on PE1 | 312](#)
- [Configuring LACP for EVPN Active-Active Multihoming on PE2 | 318](#)
- [Configuring LACP for EVPN Active-Active Multihoming on CE2 | 325](#)

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the [edit] hierarchy level, and then enter `commit` from configuration mode.

Device PE3

```
set interfaces xe-0/0/0 unit 0 family inet address 10.10.10.1/24
set interfaces xe-0/0/2 unit 0 family inet address 10.12.12.1/24
set interfaces xe-0/0/4 unit 0 family inet address 10.14.14.1/24
set interfaces xe-0/0/8 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/8 unit 0 family ethernet-switching vlan members all
set interfaces lo0 unit 0 family inet address 10.2.3.1/32 primary
set vlans v100 vlan-id 100
set vlans v200 vlan-id 200
set vlans v100 vxlan vni 100
set vlans v200 vxlan vni 200
set routing-options router-id 10.2.3.1
set routing-options autonomous-system 11
set switch-options vtep-source-interface lo0.0
```

```

set switch-options route-distinguisher 10.2.3.1:1
set switch-options vrf-target target:1111:11
set protocols bgp group pe type internal
set protocols bgp group pe local-address 10.2.3.1
set protocols bgp group pe family evpn signaling
set protocols bgp group pe neighbor 10.2.3.3
set protocols bgp group pe neighbor 10.2.3.4
set protocols ospf area 0.0.0.0 interface xe-0/0/2
set protocols ospf area 0.0.0.0 interface lo0 passive
set protocols ospf area 0.0.0.0 interface xe-0/0/0
set protocols ospf area 0.0.0.0 interface xe-0/0/4
set protocols evpn encapsulation vxlan
set protocols evpn extended-vni-list all

```

Device PE1

```

set chassis aggregated-devices ethernet device-count 1
set interfaces xe-0/0/2 unit 0 family inet address 10.12.12.2/24
set interfaces xe-0/0/3 unit 0 family inet address 10.11.11.2/24
set interfaces xe-0/0/9 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/9 unit 0 family ethernet-switching vlan members v200
set interfaces ae0 unit 0 family ethernet-switching interface-mode trunk
set interfaces ae0 unit 0 family ethernet-switching vlan members all
set interfaces xe-0/0/55:0 ether-options 802.3ad ae0
set interfaces ae0 esi 00:03:03:03:03:03:03:03:03:03
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp system-id 00:00:01:00:00:01
set interfaces lo0 unit 0 family inet address 10.2.3.3/32 primary
set vlans v100 vlan-id 100
set vlans v200 vlan-id 200
set vlans v100 vxlan vni 100
set vlans v200 vxlan vni 200
set routing-options router-id 10.2.3.3
set routing-options autonomous-system 11
set switch-options vtep-source-interface lo0
set switch-options route-distinguisher 10.2.3.3:1
set switch-options vrf-target target:1111:11
set protocols bgp group pe type internal
set protocols bgp group pe local-address 10.2.3.3
set protocols bgp group pe family evpn signaling
set protocols bgp group pe neighbor 10.2.3.1

```

```

set protocols bgp group pe neighbor 10.2.3.4
set protocols ospf area 0.0.0.0 interface xe-0/0/2
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface xe-0/0/3
set protocols evpn encapsulation vxlan
set protocols evpn extended-vni-list all

```

Device PE2

```

set chassis aggregated-devices ethernet device-count 1
set interfaces xe-0/0/2 unit 0 family inet address 10.14.14.2/24
set interfaces xe-0/0/5 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/5 unit 0 family ethernet-switching vlan members all
set interfaces ae0 unit 0 family ethernet-switching interface-mode trunk
set interfaces ae0 unit 0 family ethernet-switching vlan members all
set interfaces xe-0/0/55:0 ether-options 802.3ad ae0
set interfaces ae0 esi 00:03:03:03:03:03:03:03:03
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp system-id 00:00:01:00:00:01
set interfaces lo0 unit 0 family inet address 10.2.3.4/32 primary
set vlans v100 vlan-id 100
set vlans v200 vlan-id 200
set vlans v100 vxlan vni 100
set vlans v200 vxlan vni 200
set routing-options router-id 10.2.3.4
set routing-options autonomous-system 11
set switch-options vtep-source-interface lo0
set switch-options route-distinguisher 10.2.3.4:1
set switch-options vrf-target target:1111:11
set protocols bgp group pe type internal
set protocols bgp group pe local-address 10.2.3.4
set protocols bgp group pe family evpn signaling
set protocols bgp group pe neighbor 10.2.3.1
set protocols bgp group pe neighbor 10.2.3.2
set protocols bgp group pe neighbor 10.2.3.3
set protocols ospf area 0.0.0.0 interface xe-0/0/2
set protocols ospf area 0.0.0.0 interface lo0 passive
set protocols evpn encapsulation vxlan
set protocols evpn extended-vni-list all

```

Device CE2

```
set vlans v100 vlan-id 100
set vlans v200 vlan-id 200
set interfaces xe-0/0/3 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/3 unit 0 family ethernet-switching vlan members all
set interfaces xe-0/0/5 ether-options 802.3ad ae0
set interfaces xe-0/0/0 ether-options 802.3ad ae0
set interfaces ae0 unit 0 family ethernet-switching interface-mode trunk
set interfaces ae0 unit 0 family ethernet-switching vlan members all
set interfaces ae0 aggregated-ether-options lacp active
```

Configuring LACP for EVPN Active-Active Multihoming on PE3

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the [CLI User Guide](#).

To configure Device PE3:

1. Configure uplink interfaces towards PE1 and PE2 devices.

```
[edit interfaces]
user@CE1# set xe-0/0/0 unit 0 family inet address 10.10.10.1/24
user@CE1# set xe-0/0/2 unit 0 family inet address 12.12.12.1/24
user@CE1# set xe-0/0/4 unit 0 family inet address 14.14.14.1/24
```

2. Configure xe-0/0/8 as a Layer 2 interface.

```
[edit interfaces]
user@CE1# set xe-0/0/8 unit 0 family ethernet-switching interface-mode trunk
user@CE1# set xe-0/0/8 unit 0 family ethernet-switching vlan members all
```

3. Configure a loopback interface.

```
[edit interfaces]
user@CE1# set lo0 unit 0 family inet address 10.2.3.1/32 primary
```

4. Create VLANs v100 and v200.

```
[edit vlans]
user@CE1# set v100 vlan-id 100
user@CE1# set v200 vlan-id 200
```

5. Map VLANs v100 and v200 to VNIs 100 and 200.

```
[edit vlans]
user@CE1# set v100 vxlan vni 100
user@CE1# set v200 vxlan vni 200
```

6. Configure the router ID and autonomous system number.

```
[edit routing-options]
user@CE1# set router-id 10.2.3.1
user@CE1# set autonomous-system 11
```

7. Specify the loopback interface as the source address for the VTEP tunnel.

```
[edit switch-options]
user@CE1# set vtep-source-interface lo0.0
```

8. Specify a route distinguisher to uniquely identify routes sent from this device.

```
[edit switch-options]
user@CE1# set route-distinguisher 10.2.3.1:1
```

9. Specify the global VRF export policy.

```
[edit switch-options]
user@CE1# set vrf-target target:1111:11
```

10. Configure an internal BGP group for PE3 to peer with PE1 and PE2.

```
[edit protocols]
user@CE1# set bgp group pe type internal
user@CE1# set bgp group pe local-address 10.2.3.1
user@CE1# set bgp group pe family evpn signaling
user@CE1# set bgp group pe neighbor 10.2.3.3
user@CE1# set bgp group pe neighbor 10.2.3.4
```

11. Configure an OSPF area.

```
[edit protocols]
user@CE1# set ospf area 0.0.0.0 interface xe-0/0/2
user@CE1# set ospf area 0.0.0.0 interface lo0 passive
user@CE1# set ospf area 0.0.0.0 interface xe-0/0/0
user@CE1# set ospf area 0.0.0.0 interface xe-0/0/4
```

12. Set VXLAN as the data plane encapsulation for EVPN.

```
[edit protocols]
user@CE1# set evpn encapsulation vxlan
```

13. Specify that all VNI(s) are advertised by EVPN.

```
[edit protocols]
user@CE1# set evpn extended-vni-list all
```

Results

From configuration mode, confirm your configuration by entering the `show chassis`, `show interfaces`, `show routing-options`, `show protocols`, and `show routing-instances` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE3# show interfaces
xe-0/0/0 {
  unit 0 {
    family inet {
```



```

        address 10.10.10.1/24;
    }
}
xe-0/0/2 {
    unit 0 {
        family inet {
            address 10.10.6.1/30;
            address 10.12.12.1/24;
        }
    }
}
xe-0/0/4 {
    unit 0 {
        family inet {
            address 10.14.14.1/24;
        }
    }
}
xe-0/0/8 {
    unit 0 {
        family ethernet-switching {
            interface-mode trunk;
            vlan {
                members all;
            }
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.10.0.1/32;
            address 10.2.3.1/32 {
                primary;
            }
        }
    }
}

```

```
    }
}
```

```
user@PE3# show vlans
```

```
v100 {
    vlan-id 100;
    vxlan {
        vni 100;
    }
}
v200 {
    vlan-id 200;
    vxlan {
        vni 200;
    }
}
```

```
user@PE3# show routing-options
```

```
router-id 10.2.3.1;
autonomous-system 11;
```

```
user@PE3# show switching-options
```

```
vtep-source-interface lo0;
route-distinguisher 10.2.3.1:1;
vrf-target target:1111:11;
```

```
user@PE3# show protocols bgp
```

```
group pe {
    type internal;
    local-address 10.2.3.1;
    family evpn {
        signaling;
    }
    neighbor 10.2.3.3;
```

```
neighbor 10.2.3.4;
}
```

```
user@PE3# show protocols ospf
area 0.0.0.0 {
    interface lo0.0 {
        passive;
    }
    interface xe-0/0/2;
    interface xe-0/0/0;
    interface xe-0/0/4;
}
```

If you are done configuring the device, enter `commit` from configuration mode.

Configuring LACP for EVPN Active-Active Multihoming on PE1

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the [CLI User Guide](#).

To configure Device PE1:

1. Specify the number of aggregated Ethernet interfaces to be created on Device PE1.

```
[edit chassis]
user@PE1# set aggregated-devices ethernet device-count 1
```

2. Configure the interfaces that connect to the CE device.

```
[edit interfaces]
user@PE1# set xe-0/0/2 unit 0 family inet address 10.12.12.2/24
user@PE1# set xe-0/0/2 unit 0 family inet address 10.11.11.2/24
```

3. Configure xe-0/0/9 as a Layer 2 interface.

```
[edit interfaces]
user@PE1# set xe-0/0/9 unit 0 family ethernet-switching interface-mode trunk
user@PE1# set xe-0/0/9 unit 0 family ethernet-switching vlan members v200
```

4. Configure ae0 as a Layer 2 interface.

```
[edit interfaces]
user@PE1# set ae0 unit 0 family ethernet-switching interface-mode trunk
user@PE1# set ae0 unit 0 family ethernet-switching vlan members all
```

5. Configure the interface towards the multihomed device, CE2.

Use the same ESI value on all PE devices where the CE2 is multihomed.

```
[edit interfaces]
user@PE1# set xe-0/0/55:0 ether-options 802.3ad ae0
user@PE1# set ae0 esi 00:03:03:03:03:03:03:03:03
user@PE1# set ae0 esi all-active
```

6. Configure LACP on the ae0.

Use the same system ID value on all PE devices where the CE2 is multihomed.

```
[edit interfaces]
user@PE1# set ae0 aggregated-ether-options lacp active
user@PE1# set ae0 aggregated-ether-options lacp system-id 00:00:01:00:00:01
```

7. Configure a loopback interface.

```
[edit interfaces]
user@PE1# set lo0 unit 0 family inet address 10.2.3.3/32 primary
```

8. Create VLANs v100 and v200.

```
[edit vlans]
user@PE1# set v100 vlan-id 100
user@PE1# set v200 vlan-id 200
```

9. Map VLANs v100 and v200 to VNIs 100 and 200.

```
[edit vlans]
user@PE1# set v100 vxlan vni 100
user@PE1# set v200 vxlan vni 200
```

10. Configure a router ID and autonomous system number.

```
[edit routing-options]
user@PE1# set router-id 10.2.3.3
user@PE1# set autonomous-system 11
```

11. Specify the loopback interface as the source address for the VTEP tunnel.

```
[edit switch-options]
user@PE1# set vtep-source-interface lo0.0
```

12. Specify a route distinguisher to uniquely identify routes sent from this device.

```
[edit switch-options]
user@PE1# set route-distinguisher 10.2.3.3:1
```

13. Specify the global VRF export policy.

```
[edit switch-options]
user@PE1# set vrf-target target:1111:11
```

14. Configure an internal BGP group for PE3 to peer with PE1 and PE2.

```
[edit protocols]
user@PE1# set bgp group pe type internal
user@PE1# set bgp group pe local-address 10.2.3.3
user@PE1# set bgp group pe family evpn signaling
user@PE1# set bgp group pe neighbor 10.2.3.1
user@PE1# set bgp group pe neighbor 10.2.3.4
```

15. Configure an OSPF area.

```
[edit protocols]
user@PE1# set ospf area 0.0.0.0 interface xe-0/0/2
user@PE1# set ospf area 0.0.0.0 interface lo0 passive
user@PE1# set ospf area 0.0.0.0 interface xe-0/0/3
```

16. Set VXLAN as the data plane encapsulation for EVPN.

```
[edit protocols]
user@PE1# set evpn encapsulation vxlan
```

17. Specify that all VNI(s) are advertised by EVPN.

```
[edit protocols]
user@PE1# set evpn extended-vni-list all
```

Results

From configuration mode, confirm your configuration by entering the `show chassis`, `show interfaces`, `show routing-options`, `show protocols`, and `show routing-instances` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show chassis
aggregated-devices {
  ethernet {
    device-count 1;
```

```

    }
}

```

```

user@PE1# show interfaces
xe-0/0/2 {
    unit 0 {
        family inet {
            address 10.10.6.1/30;
            address 10.12.12.1/24;
            address 10.12.12.2/24;
        }
    }
}
xe-0/0/3 {
    unit 0 {
        family inet {
            address 10.11.11.2/24;
        }
    }
}
xe-0/0/9 {
    unit 0 {
        family ethernet-switching {
            interface-mode trunk;
            vlan {
                members v200;
            }
        }
    }
}
xe-0/0/55:0 {
    ether-options {
        802.3ad ae0;
    }
}
ae0 {
    esi {
        00:03:03:03:03:03:03:03:03;
        all-active;
    }
    aggregated-ether-options {

```

```

        lacp {
            active;
            system-id 00:00:01:00:00:01;
        }
    }
    unit 0 {
        family ethernet-switching {
            interface-mode trunk;
            vlan {
                members all;
            }
        }
    }
}

```

```
user@PE1# show vlans
```

```

v100 {
    vlan-id 100;

    vxlan {
        vni 100;
    }
}
v200 {
    vlan-id 200;

    vxlan {
        vni 200;
    }
}

```

```
user@PE1# show routing-options
```

```

router-id 10.2.3.3;
autonomous-system 11;

```

```
user@PE1# show switching-options
```

```
vtep-source-interface lo0;
```



```
route-distinguisher 10.2.3.3:1;
vrf-target target:1111:11;
```

```
user@PE1# show protocols bgp
group pe {
    type internal;
    local-address 10.2.3.3;
    family evpn {
        signaling;
    }
    neighbor 10.2.3.3;
    neighbor 10.2.3.4;
    neighbor 10.2.3.1;
}
```

```
user@PE1# show protocols ospf
area 0.0.0.0 {
    interface lo0 {
        passive;
    }
    interface xe-0/0/2;
    interface xe-0/0/3;
}
```

```
user@PE1# show protocols evpn
encapsulation vxlan;
extended-vni-list all;
```

If you are done configuring the device, enter `commit` from configuration mode.

Configuring LACP for EVPN Active-Active Multihoming on PE2

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the [CLI User Guide](#).

To configure Device PE2:

1. Specify the number of aggregated Ethernet interfaces to be created on Device PE1.

```
[edit chassis]
user@PE2# set aggregated-devices ethernet device-count 1
```

2. Configure the interface that connects to the CE device.

```
[edit interfaces]
user@PE2# set xe-0/0/2 unit 0 family inet address 10.14.14.2/24
```

3. Configure xe-0/0/5 as a Layer 2 interface.

```
[edit interfaces]
user@PE2# set xe-0/0/5 unit 0 family ethernet-switching interface-mode trunk
user@PE2# set xe-0/0/5 unit 0 family ethernet-switching vlan members v200
```

4. Configure ae0 as a Layer 2 interface.

```
[edit interfaces]
user@PE2# set ae0 unit 0 family ethernet-switching interface-mode trunk
user@PE2# set ae0 unit 0 family ethernet-switching vlan members all
```

5. Configure the interface towards the multihomed device, CE2.

Use the same ESI value on all PE devices where the CE2 is multihomed.

```
[edit interfaces]
user@PE2# set xe-0/0/55:0 ether-options 802.3ad ae0
user@PE2# set ae0 esi 00:03:03:03:03:03:03:03:03
user@PE2# set ae0 esi all-active
```

6. Configure LACP on the ae0.

Use the same system ID value on all PE devices where the CE2 is multihomed.

```
[edit interfaces]
user@PE2# set ae0 aggregated-ether-options lacp active
user@PE2# set ae0 aggregated-ether-options lacp system-id 00:00:01:00:00:01
```

7. Configure a loopback interface.

```
[edit interfaces]
user@PE2# set lo0 unit 0 family inet address 10.2.3.4/32 primary
```

8. Create VLANs v100 and v200.

```
[edit vlans]
user@PE2# set v100 vlan-id 100
user@PE2# set v200 vlan-id 200
```

9. Map VLANs v100 and v200 to VNIs 100 and 200.

```
[edit vlans]
user@PE2# set v100 vxlan vni 100
user@PE2# set v200 vxlan vni 200
```

10. Configure a router ID and autonomous system number.

```
[edit routing-options]
user@PE2# set router-id 10.2.3.4
user@PE2# set autonomous-system 11
```

11. Specify the loopback interface as the source address for the VTEP tunnel.

```
[edit switch-options]
user@PE2# set vtep-source-interface lo0.0
```

12. Specify a route distinguisher to uniquely identify routes sent from this device.

```
[edit switch-options]
user@PE2# set route-distinguisher 10.2.3.4:1
```

13. Specify the global VRF export policy.

```
[edit switch-options]
user@PE2# set vrf-target target:1111:11
```

14. Configure an internal BGP group for PE3 to peer with PE1 and PE2.

```
[edit protocols]
user@PE2# set bgp group pe type internal
user@PE2# set bgp group pe local-address 10.2.3.4
user@PE2# set bgp group pe family evpn signaling
user@PE2# set bgp group pe neighbor 10.2.3.1
user@PE2# set bgp group pe neighbor 10.2.3.2
user@PE2# set bgp group pe neighbor 10.2.3.3
```

15. Configure an OSPF area.

```
[edit protocols]
user@PE2# set ospf area 0.0.0.0 interface xe-0/0/2
user@PE2# set ospf area 0.0.0.0 interface lo0 passive
```

16. Set VXLAN as the data plane encapsulation for EVPN.

```
[edit protocols]
user@PE2# set evpn encapsulation vxlan
```

17. Specify that all VNI(s) are advertised by EVPN.

```
[edit protocols]
user@PE2# set evpn extended-vni-list all
```

Results

From configuration mode, confirm your configuration by entering the `show chassis`, `show interfaces`, `show routing-options`, `show protocols`, and `show routing-instances` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE2# show chassis
aggregated-devices {
  ethernet {
    device-count 1;
  }
}
```

```
user@PE2# show interfaces
xe-0/0/2 {
  unit 0 {
    family inet {
      address 10.14.14.2/24;
    }
  }
}
xe-0/0/5 {
  unit 0 {
    family ethernet-switching {
      interface-mode trunk;
      vlan {
        members all;
      }
    }
  }
}
xe-0/0/55:0 {
  ether-options {
    802.3ad ae0;
  }
}
ae0 {
  esi {
    00:03:03:03:03:03:03:03:03;
    all-active;
  }
}
```

```

    aggregated-ether-options {
        lacp {
            active;
            system-id 00:00:01:00:00:01;
        }
    }
    unit 0 {
        family ethernet-switching {
            interface-mode trunk;
            vlan {
                members all;
            }
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.2.3.4/32 {
                primary;
            }
        }
    }
}
}

```

```

user@PE2# show vlans

```

```

v100 {
    vlan-id 100;

    vxlan {
        vni 100;
    }
}
v200 {
    vlan-id 200;

    vxlan {
        vni 200;
    }
}

```

```

    }
}

```

```

user@PE2# show routing-options
router-id 10.2.3.4;
autonomous-system 11;

```

```

user@PE2# show switching-options
vtep-source-interface lo0;
route-distinguisher 10.2.3.4:1;
vrf-target target:1111:11;

```

```

user@PE2# show protocols bgp
group pe {
    type internal;
    local-address 10.2.3.4;
    family evpn {
        signaling;
    }
    neighbor 10.2.3.3;
    neighbor 10.2.3.4;
    neighbor 10.2.3.1;
}

```

```

user@PE2# show protocols ospf
area 0.0.0.0 {
    interface lo0 {
        passive;
    }
    interface xe-0/0/2;
}

```

```

user@PE2# show protocols evpn
encapsulation vxlan;
extended-vni-list all;

```

If you are done configuring the device, enter `commit` from configuration mode.

Configuring LACP for EVPN Active-Active Multihoming on CE2

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the [CLI User Guide](#).

To configure Device CE1:

1. Configure VLANs v100 and v200.

```
[edit vlans]
user@CE2# set v100 vlan-id 100
user@CE2# set v200 vlan-id 200
```

2. Configure xe-0/0/3 as a Layer 2 interface.

```
[edit interfaces]
user@CE2# set xe-0/0/3 unit 0 family ethernet-switching interface-mode trunk
user@CE2# set xe-0/0/3 unit 0 family ethernet-switching vlan members all
```

3. Add member interfaces to ae0.

```
[edit interfaces]
user@CE2# set xe-0/0/0 ether-options 802.3ad ae0
user@CE2# set xe-0/0/5 ether-options 802.3ad ae0
```

4. Configure ae0 as a Layer 2 interface.

```
[edit interfaces]
user@CE2# set ae0 unit 0 family ethernet-switching interface-mode trunk
user@CE2# set ae0 unit 0 family ethernet-switching vlan members all
```


5. Configure LACP as active for ae0.

```
[edit interfaces]
user@CE2# set ae0 aggregated-ether-options lacp active
```

Results

From configuration mode, confirm your configuration by entering the `show chassis`, `show interfaces`, `show routing-options`, `show protocols`, and `show routing-instances` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@CE2# show interfaces
xe-0/0/0 {
  ether-options {
    802.3ad ae0;
  }
}
xe-0/0/3 {
  unit 0 {
    family ethernet-switching {
      interface-mode trunk;
      vlan {
        members all;
      }
    }
  }
}
```

```
user@CE2# show vlans
v100 {
  vlan-id 100;
  vxlan {
    vni 100;
  }
}
v200 {
  vlan-id 200;
  vxlan {
    vni 200;
```

```
}  
}
```

If you are done configuring the device, enter `commit` from configuration mode.

Verification

IN THIS SECTION

[Verifying LACP Interface Status of PE1 | 327](#)

[Verifying LACP Interface Status of PE2 | 328](#)

Confirm that the configuration is working properly.

Verifying LACP Interface Status of PE1

Purpose

Verify the LACP interface state on Device PE1.

Action

From operational mode, run the **show lacp interfaces** command.

```
user@PE1> show lacp interfaces  
Aggregated interface: ae0  
LACP state:      Role  Exp  Def  Dist  Col  Syn  Aggr  Timeout  Activity  
xe-0/0/55:0      Actor No   No   Yes  Yes  Yes  Yes   Fast   Active  
xe-0/0/55:0      Partner No   No   Yes  Yes  Yes  Yes   Fast   Active  
LACP protocol:   Receive State  Transmit State      Mux State  
xe-0/0/55:0      Current      Fast periodic      Collecting distributing
```

Meaning

The LACP LAG interface state is active.

NOTE: Core isolation state down (CDN) in LACP interface indicates that the LACP interface is down because all the eBGP sessions for Ethernet VPN (EVPN) are down.

Verifying LACP Interface Status of PE2

Purpose

Verify the LACP interface state on Device PE2.

Action

From operational mode, run the **show lacp interfaces** command.

```
user@PE2> show lacp interfaces
Aggregated interface: ae0
  LACP state:      Role  Exp  Def  Dist  Col  Syn  Aggr  Timeout  Activity
    xe-0/0/55:0    Actor No   No   Yes  Yes  Yes  Yes   Fast   Active
    xe-0/0/55:0    Partner No   No   Yes  Yes  Yes  Yes   Fast   Active
  LACP protocol:  Receive State  Transmit State      Mux State
    xe-0/0/55:0    Current      Fast periodic      Collecting distributing
```

Meaning

The LACP LAG interface state is active.

NOTE: Core isolation state down (CDN) in LACP interface indicates that the LACP interface is down because all the eBGP sessions for Ethernet VPN (EVPN) are down.

RELATED DOCUMENTATION

| [Understanding LACP for EVPN Active-Active Multihoming](#)

Understanding When to Disable EVPN-VXLAN Core Isolation

IN THIS SECTION

- [Use Case 1: Example of When to Use the Core Isolation Feature | 329](#)
- [Use Case 2: Example of When to Disable the Core Isolation Feature | 330](#)

By default, spine and leaf devices in an EVPN network implement the core isolation feature. If one of these devices loses all of its EVPN BGP peering sessions, the core isolation feature, working in conjunction with Link Aggregation Control Protocol (LACP), automatically brings down all Layer 2 Ethernet Segment Identifier (ESI) link aggregation group (LAG) interfaces on the device.

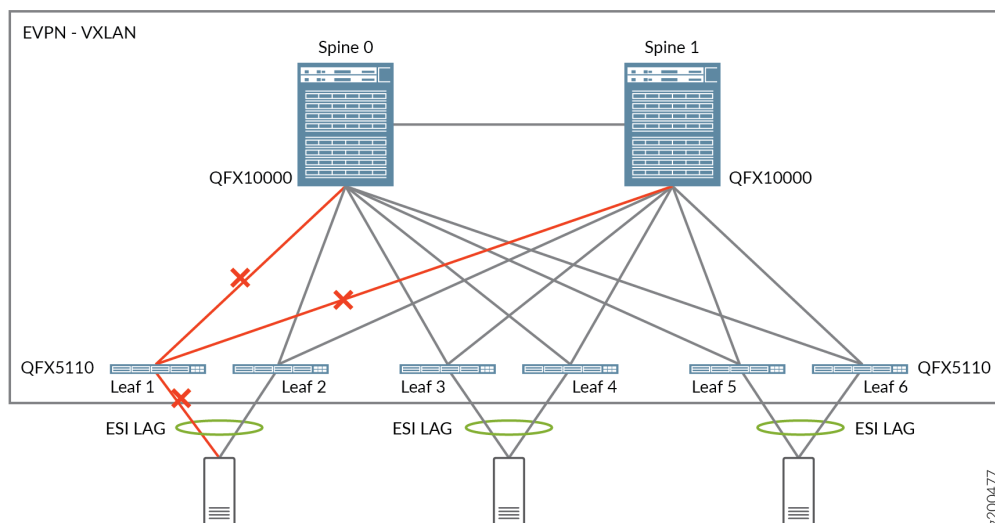
In some situations, the core isolation feature produces a favorable outcome. However, in other situations, the feature produces an undesired outcome, which you can prevent by disabling the feature. The next sections describe an example situation in each case.

Use Case 1: Example of When to Use the Core Isolation Feature

[Figure 15 on page 330](#) displays a topology in which two QFX10000 switches act as spine devices that form an EVPN-VXLAN core. In this topology, six QFX5110 switches that act as leaf devices are

multihomed in active-active mode to the spine devices, and in turn, each server is multihomed through ESI-LAG interfaces to two leaf devices.

Figure 15: EVPN-VXLAN Core Isolation Use Case



If the links between Leaf 1 and the two spine devices go down, the BGP peering sessions established over the links also go down. With the core isolation feature enabled by default, LACP sets the server-facing interface on Leaf 1 to standby mode, which blocks all traffic from the server. In this situation, the default implementation of the core isolation feature provides the following benefits:

- With the links from Leaf 1 to both spine devices down, it does not make sense for the server to continue forwarding traffic to Leaf 1.
- Traffic from the server is diverted to Leaf 2 until the links between Leaf 1 and the two spine devices are up again.

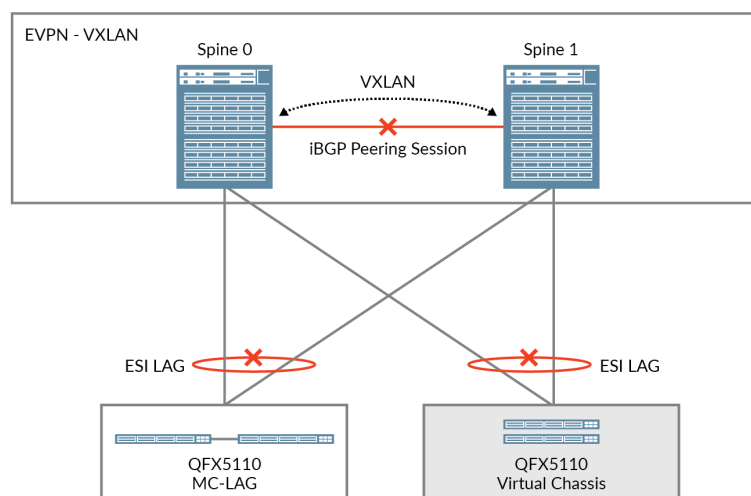
Use Case 2: Example of When to Disable the Core Isolation Feature

The topology shown in [Figure 16 on page 331](#) is migrating from multichassis link aggregation (MC-LAG) and Virtual Chassis environments to an EVPN-VXLAN environment. In this topology, the only EVPN-VXLAN components are two QFX10000 switches that act as spine devices. The QFX5110 switches that

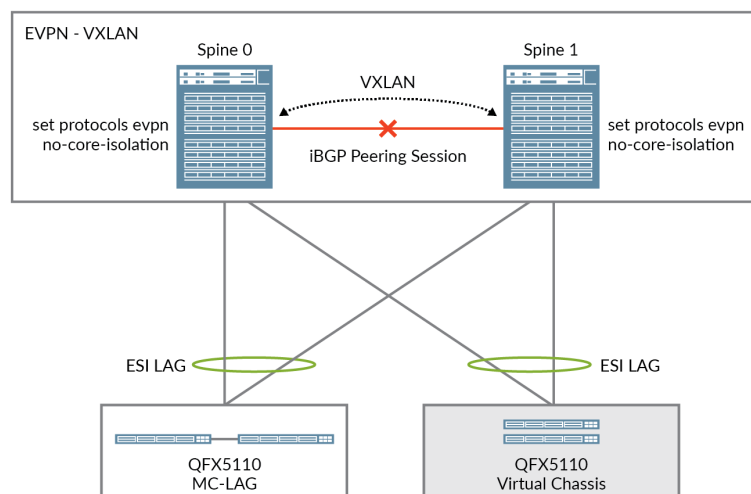
act as leaf (MC-LAG and Virtual Chassis) devices are multihomed in active-active mode through ESI-LAG interfaces to the spine devices.

Figure 16: EVPN No Core Isolation Use Case

BEFORE



AFTER



If the link between Spine 0 and Spine 1 goes down, the last established BGP peering session also goes down. With the core isolation feature enabled by default, LACP sets the leaf-facing interfaces on Spines 0 and 1 to standby mode, which causes data traffic to and from both leaf devices to be dropped. With the core isolation feature implemented at the leaf device level, traffic within the data center would essentially be halted, which is an undesired outcome.

In cases like this, you can set `no-core-isolation` at the `[edit protocols evpn]` configuration hierarchy level on each spine device to disable the core isolation feature. See the AFTER illustration in [Figure 16 on page 331](#). This statement is available only at the global level, so it applies to either all EVPN routing instances or the default switch instance on devices that don't have multiple routing instances.

Release History Table

Release	Description
17.3R3	Starting with Junos OS Release 17.3R3, you can set the <code>no-core-isolation</code> configuration statement at the <code>[edit protocols evpn]</code> hierarchy level on spine device s in the fabric to disable the core isolation feature.

RELATED DOCUMENTATION

[Example: Configuring LACP for EVPN VXLAN Active-Active Multihoming](#) | 301

Configuring Dynamic List Next Hop

The routing table on a remote PE has a next hop entry for Ethernet segment identifier (ESI) routes with multiple next-hop elements for multihomed PE devices. For EVPN active-active multihoming device, the ESI route points to two next hop elements. Prior to dynamic list next hop, the routing protocol process (rpd) removed the next-hop entry for the ESI route when the link between the CE device and a multihome PE device goes down. The rpd would then create a new next hop entry for the ESI causing mass MAC route withdrawals and additions.

Starting in Junos OS Release 17.4R1, Junos OS supports the dynamic list next-hop feature in an EVPN network. Now when the link between the CE device and a multihomed PE device goes down, rather than removing the entire next hop and creating a new next hop for the ESI , the rpd removes the affected next hop element from the dynamic list next-hop entry for the ESI route. Dynamic list next hop provides the benefit of reducing mass MAC route withdrawals, improving the device performance, and reducing network convergence time.

To enable the dynamic list next-hop feature, include the `dynamic-list-next-hop` statement in the `[edit routing-options forwarding-table]` hierarchy.

NOTE: If you are performing an unified in-service software upgrade (ISSU) to upgrade your device from a Junos OS release prior to Junos OS Release 17.4R1, you must upgrade both the

primary Routing Engine and the backup Routing Engine before enabling the dynamic list next-hop feature.

To disable the dynamic list next-hop feature when it is enabled, use the `delete routing-options forwarding-table dynamic-list-next-hop` statement.

To display the next-hop elements from the Routing Engine's forwarding table, use the `show route label` and `show route forwarding-table` commands.

The following sample output from the `show route label detail` command shows two indirect next hops for an ESI with the dynamic list next-hop feature enabled.

```
user@host> show route label 299952 detail
mpls.0: 14 destinations, 14 routes (14 active, 0 holddown, 0 hidden)
299952 (1 entry, 1 announced)
TSI:
KRT in-kernel 299952 /52 -> {Dyn list:indirect(1048577), indirect(1048574)}
    *EVPN    Preference: 7
            Next hop type: Dynamic List, Next hop index: 1048575
            Address: 0x13f497fc
            Next-hop reference count: 5
            Next hop: ELNH Address 0xb7a3d90 uflags EVPN data
                Next hop type: Indirect, Next hop index: 0
                Address: 0xb7a3d90
                Next-hop reference count: 3
                Protocol next hop: 10.255.255.2
                Label operation: Push 301344
                Indirect next hop: 0x135b5c00 1048577 INH Session ID: 0x181
                    Next hop type: Router, Next hop index: 619
                    Address: 0xb7a3d30
                    Next-hop reference count: 4
                    Next hop: 1.0.0.4 via ge-0/0/1.0
                    Label operation: Push 301344, Push 299792(top)
                    Label TTL action: no-prop-ttl, no-prop-ttl(top)
                    Load balance label: Label 301344: None; Label 299792: None;
                    Label element ptr: 0xb7a3cc0
                    Label parent element ptr: 0xb7a34e0
                    Label element references: 1
                    Label element child references: 0
                    Label element lsp id: 0
                Next hop: ELNH Address 0xb7a37f0 uflags EVPN data
```



```

Next hop type: Indirect, Next hop index: 0
Address: 0xb7a37f0
Next-hop reference count: 3
Protocol next hop: 10.255.255.3
Label operation: Push 301632
Indirect next hop: 0x135b5480 1048574 INH Session ID: 0x180
  Next hop type: Router, Next hop index: 600
  Address: 0xb7a3790
  Next-hop reference count: 4
  Next hop: 1.0.0.4 via ge-0/0/1.0
  Label operation: Push 301632, Push 299776(top)
  Label TTL action: no-prop-ttl, no-prop-ttl(top)
  Load balance label: Label 301632: None; Label 299776: None;
  Label element ptr: 0xb7a3720
  Label parent element ptr: 0xb7a3420
  Label element references: 1
  Label element child references: 0
  Label element lsp id: 0
State: <Active Int>
Age: 1:18
Validation State: unverified
Task: evpn global task
Announcement bits (2): 1-KRT 2-evpn global task
AS path: I
Routing Instance blue, Route Type Egress-MAC, ESI 00:11:22:33:44:55:66:77:88:99

```

The following sample output from the `show route forwarding table` command shows two next-hop entries for a destination with a multihomed route.

```

user@host> show route forwarding-table label 299952 extensive
MPLS:

Destination: 299952
Route type: user
Route reference: 0                               Route interface-index: 0
Multicast RPF nh index: 0
P2mpidx: 0
Flags: sent to PFE, rt nh decoupled
Next-hop type: indirect                          Index: 1048575 Reference: 2
Nexthop:
Next-hop type: composite                         Index: 601       Reference: 2
Next-hop type: indirect                         Index: 1048574 Reference: 3

```

```

Nexthop: 1.0.0.4
Next-hop type: Push 301632, Push 299776(top) Index: 600 Reference: 2
Load Balance Label: None
Next-hop interface: ge-0/0/1.0
Next-hop type: indirect          Index: 1048577 Reference: 3
Nexthop: 1.0.0.4
Next-hop type: Push 301344, Push 299792(top) Index: 619 Reference: 2
Load Balance Label: None
Next-hop interface: ge-0/0/1.0

```

The following sample shows the `show route forwarding-table` command output after one of the PE devices has been disabled. It shows one next-hop element and one empty next-hop element.

```

user@host> show route forwarding-table label 299952 extensive
Routing table: default.mpls [Index 0]
MPLS:

Destination: 299952
Route type: user
Route reference: 0          Route interface-index: 0
Multicast RPF nh index: 0
P2mpidx: 0
Flags: sent to PFE, rt nh decoupled
Next-hop type: indirect          Index: 1048575 Reference: 2
Nexthop:
Next-hop type: composite          Index: 601      Reference: 2
Next-hop type: indirect          Index: 1048577 Reference: 3
Nexthop: 1.0.0.4
Next-hop type: Push 301344, Push 299792(top) Index: 619 Reference: 2
Load Balance Label: None
Next-hop interface: ge-0/0/1.0

```

Example: Configuring an ESI on a Logical Interface With EVPN Multihoming

IN THIS SECTION

- [Requirements | 337](#)
- [Overview and Topology | 337](#)
- [EVPN Multihoming Active-Standby Configuration | 340](#)
- [Verification | 349](#)

When a customer edge (CE) device in an Ethernet VPN-Multiprotocol Label Switching (EVPN-MPLS) environment is multihomed to two or more provider edge (PE) devices, the set of Ethernet links that connect the devices comprise an Ethernet segment. An Ethernet segment identifier (ESI) is a 10-octet integer that identifies this segment. A sample ESI is 00:11:22:33:44:55:66:77:88:99.

In releases before Junos OS Release 15.1F6 and 16.1R4 for MX Series routers and in releases before Junos OS Release 17.3R1 for EX9200 switches, you can specify an ESI only on a physical or aggregated Ethernet interface, for example, `set interfaces ae0 esi 00:11:22:33:44:55:66:77:88:99`. If you specify an ESI on a physical or aggregated Ethernet interface, keep in mind that an ESI is a factor in the designated forwarder (DF) election process. For example, assume that you configure EVPN multihoming active-standby on aggregated Ethernet interface ae0, and given the ESI configured on ae0 and other determining factors, the DF election results in ae0 being in the down state. Further, all logical interfaces configured on ae0, for example, `set interfaces ae0 unit 1` and `set interfaces ae0 unit 2` are also in the down state, which renders logical interfaces ae0.1 and ae0.2 unable to provide services to their respective customer sites (VLANs).

Starting with Junos OS Releases 15.1F6 and 16.1R4 for MX Series routers and Junos OS Release 17.3R1 for EX9200 switches, you can specify an ESI on a logical interface. If you specify an ESI on a logical interface, the DF election process now occurs at the individual logical interface level, which enables you to better utilize logical interfaces. For example, assume that you configure logical interfaces ae0.1 and ae0.2 on aggregated Ethernet interface ae0. You configure EVPN multihoming active-standby on both logical interfaces and given the ESI configured on ae0.1 and other determining factors, the DF election results in ae0.1 being in the down state. Despite logical interface ae0.1 being down, logical interface ae0.2 and other logical interfaces configured on ae0 can be in the up state and provide services to their respective customer sites (VLANs).

This topic shows how to configure an ESI on logical interfaces in both EVPN multihoming active-standby and active-active modes.

Requirements

Both EVPN multihoming active-standby and multihoming active-active examples use the following hardware and software components:

- An EX9200 switch running Junos OS Release 17.3R1 or later (PE1)
- An MX Series router running Junos OS Release 15.1F6 or later, or Junos OS Release 17.1R1 or later (PE2)

Overview and Topology

EVPN Multihoming Active-Standby

Figure 17 on page 337 shows an EVPN-MPLS topology in which CE1 is multihomed to PE1 and PE2 to provide redundant paths to CE2. On CE1, the connections to PE1 and PE2 are configured as separate aggregated Ethernet interfaces. Table 8 on page 338 shows how the connections with CE1 are configured on PE1 and PE2. Note that the EVPN multihoming mode, ESI, and VLAN ID are actually configured on logical interfaces ae0.1 on each PE device.

Figure 17: EVPN-MPLS Topology with Multihoming Active-Standby

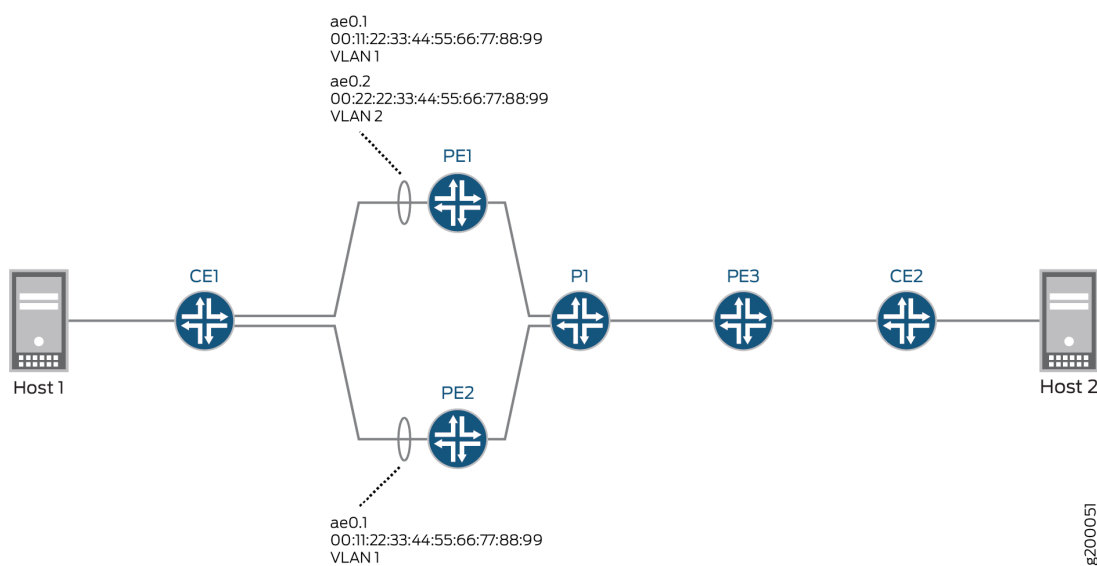


Table 8: EVPN Multihoming Active-Standby: Configuring the Connection with CE1 on PE1 and PE2

Device	Physical Interface	Aggregated Ethernet Interface	Logical Interface	EVPN Multihoming Mode	ESI	VLAN ID
PE1	xe-2/0/0	ae0	ae0.1	single-active	00:11:22:33:44:55:66:77:88:99	1
PE2	xe-3/0/2	ae0	ae0.1	single-active	00:11:22:33:44:55:66:77:88:99	1

Based on the DF election, logical interface ae0.1 on PE2 is up, and logical interface ae0.1 on PE1 is down.

[Table 9 on page 338](#) also shows the configuration for logical interface ae0.2 on PE1. Note that logical interface ae0.2 provides services for a different VLAN and is configured with a different ESI than logical interface ae0.1, which is configured on the same aggregated Ethernet interface. As a result, logical interface ae0.2 is up and providing services to VLAN 2 despite the fact that logical interface ae0.1 is in the down state.

Table 9: Multihoming Active-Standby: Configuring Logical Interface on PE1 that Provides Services to a Different VLAN

Device	Physical Interface	Aggregated Ethernet Interface	Logical Interface	EVPN Multihoming Mode	ESI	VLAN ID
PE1	xe-2/0/0	ae0	ae0.2	single-active	00:22:22:33:44:55:66:77:88:99	2

EVPN Multihoming Active-Active

[Figure 18 on page 339](#) shows an EVPN-MPLS topology in which CE1 is multihomed to PE1 and PE2 to provide redundant paths to CE2. On CE1, the connections to PE1 and PE2 are configured as one aggregated Ethernet interface. [Table 10 on page 339](#) shows how the connections with CE1 are

configured on PE1 and PE2. Note that the EVPN multihoming mode, ESI, and VLAN ID are actually configured on logical interfaces ae0.1 on each device.

Figure 18: EVPN-MPLS Topology with Multihoming Active-Active

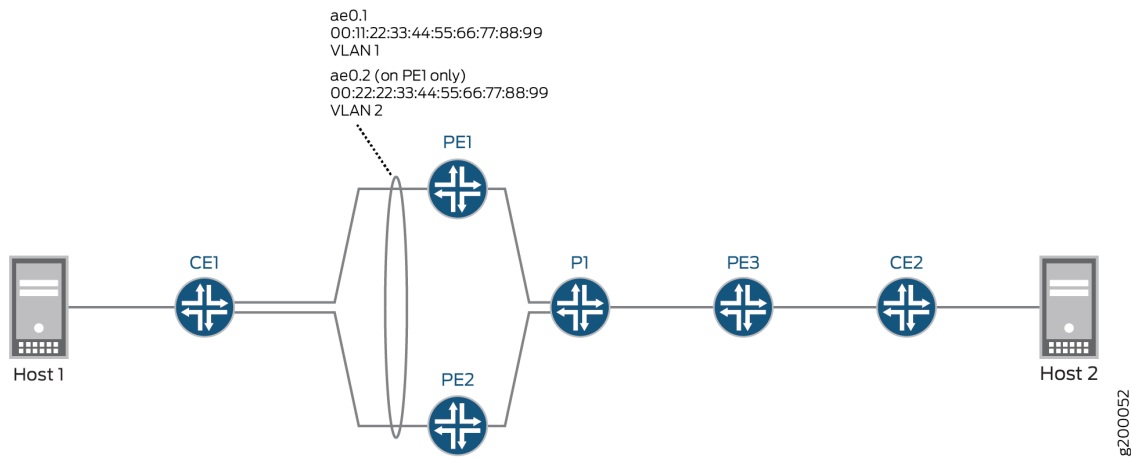


Table 10: Multihoming Active-Active: Configuring the Connection with CE1 on PE1 and PE2

Device	Physical Interface	Aggregated Ethernet Interface	Logical Interface	EVPN Multihoming Mode	ESI	VLAN ID
PE1	xe-2/0/0	ae0	ae0.1	all-active	00:11:22:33:44:55:66:77:88:99	1
PE2	xe-3/0/2	ae0	ae0.1	all-active	00:11:22:33:44:55:66:77:88:99	1

Based on the DF election, logical interface ae0.1 on PE1 is in the up state, and logical interface ae0.1 on PE2 is in the down state.

[Table 11 on page 340](#) also shows the configuration for logical interface ae0.2 on PE1. Note that logical interface ae0.2 provides services for a different VLAN and is configured with a different ESI than logical interface ae0.1, which is in the same aggregated Ethernet interface. As a result, logical interface ae0.2 is down and unable to provide services to VLAN 2 despite the fact that logical interface ae0.1 is in the up state.

Table 11: EVPN Multihoming Active-Active: Configuring Interface on PE1 that Provides Services to a Different VLAN

Device	Physical Interface	Aggregated Ethernet Interface	Logical Interface	EVPN Multihoming Mode	ESI	VLAN ID
PE1	xe-2/0/0	ae0	ae0.2	all-active	00:22:22:33:44:55:66:77:88:99	2

EVPN Multihoming Active-Standby Configuration

IN THIS SECTION

- [CLI Quick Configuration | 340](#)
- [Procedure | 342](#)
- [EVPN Multihoming Active-Active Configuration | 345](#)

NOTE: The configurations for PE1 (EX9200) and PE2 (MX Series router) focus on configuring EVPN multihoming active-standby and ESIs on logical interfaces. The configurations do not include all EVPN-related configurations for physical interfaces, aggregated Ethernet interfaces, logical interfaces, and routing instances. For a more comprehensive configuration of EVPN multihoming active-standby in an EVPN-MPLS environment, see ["Example: Configuring EVPN Active-Standby Multihoming" on page 154](#). Note that the referenced example shows how to configure ESIs on physical and aggregated Ethernet interfaces.

CLI Quick Configuration

PE1

```
set interfaces xe-2/0/0 gigether-options 802.3ad ae0
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 unit 1 encapsulation vlan-bridge
set interfaces ae0 unit 1 vlan-id 1
set interfaces ae0 unit 1 esi 00:11:22:33:44:55:66:77:88:99
```

```

set interfaces ae0 unit 1 esi single-active
set interfaces ae0 unit 2 encapsulation vlan-bridge
set interfaces ae0 unit 2 vlan-id 2
set interfaces ae0 unit 2 esi 00:22:22:33:44:55:66:77:88:99
set interfaces ae0 unit 2 esi single-active
set interfaces irb unit 1 family inet address 192.0.2.1/24
set interfaces irb unit 2 family inet address 192.0.2.2/24
set routing-instances blue instance-type evpn
set routing-instances blue vlan-id 1
set routing-instances blue interface ae0.1
set routing-instances blue l3-interface irb.1
...
set routing-instances blue protocols evpn interface ae0.1
set routing-instances green instance-type evpn
set routing-instances green vlan-id 2
set routing-instances green interface ae0.2
set routing-instances green l3-interface irb.2
...
set routing-instances green protocols evpn interface ae0.2
set routing-instances vrf instance-type vrf
set routing-instances vrf interface irb.1
set routing-instances vrf interface irb.2
...

```

PE2

```

set interfaces xe-3/0/2 gigether-options 802.3ad ae0
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 unit 1 encapsulation vlan-bridge
set interfaces ae0 unit 1 vlan-id 1
set interfaces ae0 unit 1 esi 00:11:22:33:44:55:66:77:88:99
set interfaces ae0 unit 1 esi single-active
set interfaces irb unit 1 family inet address 192.0.2.1/24
set routing-instances blue instance-type evpn
set routing-instances blue vlan-id 1
set routing-instances blue interface ae0.1
set routing-instances blue routing-interface irb.1
...
set routing-instances blue protocols evpn interface ae0.1
set routing-instances vrf instance-type vrf

```



```
set routing-instances vrf interface irb.1
...
```

Procedure

Step-by-Step Procedure

To configure EVPN multihoming active-standby on PE1:

1. Specify an Ethernet interface as a member of aggregated Ethernet interface ae0.

```
[edit interfaces]
user@switch# set xe-2/0/0 gigether-options 802.3ad ae0
```

2. Configure aggregated Ethernet interface ae0 to simultaneously transmit 802.1Q VLAN single-tag and dual-tag frames and to support different types of Ethernet encapsulation at the logical interface level.

```
[edit interfaces]
user@switch# set ae0 flexible-vlan-tagging
user@switch# set ae0 encapsulation flexible-ethernet-services
```

3. On aggregated Ethernet interface ae0, configure logical interfaces ae0.1 and ae0.2. Configure the logical interfaces to use VLAN bridge encapsulation, and map the logical interfaces to VLANs 1 and 2, respectively. Also, assign an ESI to the logical interfaces, and enable EVPN multihoming active-standby.

```
[edit interfaces]
user@switch# set ae0 unit 1 encapsulation vlan-bridge
user@switch# set ae0 unit 1 vlan-id 1
user@switch# set ae0 unit 1 esi 00:11:22:33:44:55:66:77:88:99
user@switch# set ae0 unit 1 esi single-active
user@switch# set ae0 unit 2 encapsulation vlan-bridge
user@switch# set ae0 unit 2 vlan-id 2
user@switch# set ae0 unit 2 esi 00:22:22:33:44:55:66:77:88:99
user@switch# set ae0 unit 2 esi single-active
```

4. Configure IRB interfaces irb.1 and irb.2, and assign an IP address to each interface.

```
[edit interfaces]
set interfaces irb unit 1 family inet address 192.0.2.1/24
set interfaces irb unit 2 family inet address 192.0.2.2/24
```

5. Configure an EVPN routing instance named blue. Map the routing instance to VLAN 1, logical interface ae0.1, and IRB interface irb.1. Configure EVPN logical interface ae0.1 for the EVPN routing instance.

```
[edit routing-instances]
user@switch# set blue instance-type evpn
user@switch# set blue vlan-id 1
user@switch# set blue interface ae0.1
user@switch# set blue l3-interface irb.1
user@switch# set blue protocols evpn interface ae0.1
```

6. Configure an EVPN routing instance named green. Map the routing instance to VLAN 2, logical interface ae0.2, and IRB interface irb.2. Configure logical interface ae0.2 for the EVPN routing instance.

```
[edit routing-instances]
user@switch# set green instance-type evpn
user@switch# set green vlan-id 2
user@switch# set green interface ae0.2
user@switch# set green l3-interface irb.2
user@switch# set green protocols evpn interface ae0.2
```

7. Configure a VRF routing instance, and add IRB interfaces irb.1 and irb.2 to the routing instance.

```
[edit routing-instances]
set vrf instance-type vrf
set vrf interface irb.1
set vrf interface irb.2
```

Step-by-Step Procedure

To configure EVPN multihoming active-standby on PE2:

1. Specify an Ethernet interface as a member of aggregated Ethernet interface ae0.

```
[edit interfaces]
user@router# set xe-3/0/2 gigether-options 802.3ad ae0
```

2. Configure aggregated Ethernet interface ae0 to simultaneously transmit 802.1Q VLAN single-tag and dual-tag frames and to support different types of Ethernet encapsulation at the logical interface level.

```
[edit interfaces]
user@router# set ae0 flexible-vlan-tagging
user@router# set ae0 encapsulation flexible-ethernet-services
```

3. On aggregated Ethernet interface ae0, configure logical interface ae0.1. Configure the logical interface to use VLAN bridge encapsulation, and map the logical interface to VLAN 1 and 2. Also, assign an ESI to the logical interface, and enable EVPN multihoming active-standby.

```
[edit interfaces]
user@router# set ae0 unit 1 encapsulation vlan-bridge
user@router# set ae0 unit 1 vlan-id 1
user@router# set ae0 unit 1 esi 00:11:22:33:44:55:66:77:88:99
user@router# set ae0 unit 1 esi single-active
```

4. Configure IRB interface irb.1, and assign an IP address to the interface.

```
[edit interfaces]
user@router# set irb unit 1 family inet address 192.0.2.1/24
```

5. Configure an EVPN routing instance named blue. Map the routing instance to VLAN 1, logical interface ae0.1, and IRB interface irb.1. Configure EVPN logical interface ae0.1 for the EVPN routing instance.

```
[edit routing-instances]
user@router# set blue instance-type evpn
user@router# set blue vlan-id 1
user@router# set blue interface ae0.1
user@router# set blue routing-interface irb.1
user@router# set blue protocols evpn interface ae0.1
```

6. Configure a VRF routing instance, and add IRB interface irb.1 to the routing instance.

```
[edit routing-instances]
user@router# set vrf instance-type vrf
user@router# set vrf interface irb.1
```

EVPN Multihoming Active-Active Configuration

CLI Quick Configuration

NOTE: The configurations for PE1 (EX9200) and PE2 (MX Series router) focus on configuring EVPN multihoming active-active and ESIs on logical interfaces. The configurations do not include all EVPN-related configurations for physical interfaces, aggregated Ethernet interfaces, logical interfaces, and routing instances. For a more comprehensive configuration of EVPN multihoming active-active, see ["Example: Configuring EVPN Active-Active Multihoming" on page 213](#). Note that the referenced example shows how to configure ESIs on physical and aggregated Ethernet interfaces.

PE1

```
set interfaces xe-2/0/0 gigether-options 802.3ad ae0
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 unit 1 encapsulation vlan-bridge
set interfaces ae0 unit 1 vlan-id 1
set interfaces ae0 unit 1 esi 00:11:22:33:44:55:66:77:88:99
set interfaces ae0 unit 1 esi all-active
set interfaces ae0 unit 2 encapsulation vlan-bridge
set interfaces ae0 unit 2 vlan-id 2
set interfaces ae0 unit 2 esi 00:22:22:33:44:55:66:77:88:99
set interfaces ae0 unit 2 esi all-active
set interfaces irb unit 1 family inet address 192.0.2.1/24
set interfaces irb unit 2 family inet address 192.0.2.2/24
set routing-instances blue instance-type evpn
set routing-instances blue vlan-id 1
set routing-instances blue interface ae0.1
set routing-instances blue l3-interface irb.1
...
set routing-instances blue protocols evpn interface ae0.1
```

```

set routing-instances green instance-type evpn
set routing-instances green vlan-id 2
set routing-instances green interface ae0.2
set routing-instances green l3-interface irb.2
...
set routing-instances green protocols evpn interface ae0.2
set routing-instances vrf instance-type vrf
set routing-instances vrf interface irb.1
set routing-instances vrf interface irb.2
...

```

PE2

```

set interfaces xe-3/0/2 gigether-options 802.3ad ae0
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 unit 1 encapsulation vlan-bridge
set interfaces ae0 unit 1 vlan-id 1
set interfaces ae0 unit 1 esi 00:11:22:33:44:55:66:77:88:99
set interfaces ae0 unit 1 esi all-active
set interfaces irb unit 1 family inet address 192.0.2.1/24
set routing-instances blue instance-type evpn
set routing-instances blue vlan-id 1
set routing-instances blue interface ae0.1
set routing-instances blue routing-interface irb.1
...
set routing-instances blue protocols evpn interface ae0.1
set routing-instances vrf instance-type vrf
set routing-instances vrf interface irb.1
...

```

Step-by-Step Procedure

To configure EVPN multihoming active-active on PE1:

1. Specify an Ethernet interface as a member of aggregated Ethernet interface ae0.

```

[edit interfaces]
user@switch# set xe-2/0/0 gigether-options 802.3ad ae0

```

2. Configure aggregated Ethernet interface ae0 to simultaneously transmit 802.1Q VLAN single-tag and dual-tag frames and to support different types of Ethernet encapsulation at the logical interface level.

```
[edit interfaces]
user@switch# set ae0 flexible-vlan-tagging
user@switch# set ae0 encapsulation flexible-ethernet-services
```

3. On aggregated Ethernet interface ae0, configure logical interfaces ae0.1 and ae0.2. Configure the logical interfaces to use VLAN bridge encapsulation, and map the logical interfaces to VLANs 1 and 2, respectively. Also, assign an ESI to the logical interfaces, and enable EVPN multihoming active-active.

```
[edit interfaces]
user@switch# set ae0 unit 1 encapsulation vlan-bridge
user@switch# set ae0 unit 1 vlan-id 1
user@switch# set ae0 unit 1 esi 00:11:22:33:44:55:66:77:88:99
user@switch# set ae0 unit 1 esi all-active
user@switch# set ae0 unit 2 encapsulation vlan-bridge
user@switch# set ae0 unit 2 vlan-id 2
user@switch# set ae0 unit 2 esi 00:22:22:33:44:55:66:77:88:99
user@switch# set ae0 unit 2 esi all-active
```

4. Configure IRB interfaces irb.1 and irb.2, and assign an IP address to each interface.

```
[edit interfaces]
set interfaces irb unit 1 family inet address 192.0.2.1/24
set interfaces irb unit 2 family inet address 192.0.2.2/24
```

5. Configure an EVPN routing instance named blue. Map the routing instance to VLAN 1, logical interface ae0.1, and IRB interface irb.1. Configure logical interface ae0.1 for the EVPN routing instance.

```
[edit routing-instances]
user@switch# set blue instance-type evpn
user@switch# set blue vlan-id 1
user@switch# set blue interface ae0.1
user@switch# set blue l3-interface irb.1
user@switch# set blue protocols evpn interface ae0.1
```

6. Configure an EVPN routing instance named green. Map the routing instance to VLAN 2, logical interface ae0.2, and IRB interface irb.2. Configure logical interface ae0.2 for the EVPN routing instance.

```
[edit routing-instances]
user@switch# set green instance-type evpn
user@switch# set green vlan-id 2
user@switch# set green interface ae0.2
user@switch# set green l3-interface irb.2
user@switch# set green protocols evpn interface ae0.2
```

7. Configure a VRF routing instance, and add IRB interfaces irb.1 and irb.2 to the routing instance.

```
[edit routing-instances]
set vrf instance-type vrf
set vrf interface irb.1
set vrf interface irb.2
```

Step-by-Step Procedure

To configure EVPN multihoming active-active on PE2:

1. Specify Ethernet interface xe-3/0/2 as a member of aggregated Ethernet interface ae0.

```
[edit interfaces]
user@router# set xe-3/0/2 gigether-options 802.3ad ae0
```

2. Configure aggregated Ethernet interface ae0 to simultaneously transmit 802.1Q VLAN single-tag and dual-tag frames and to support different types of Ethernet encapsulation at the logical interface level.

```
[edit interfaces]
user@router# set ae0 flexible-vlan-tagging
user@router# set ae0 encapsulation flexible-ethernet-services
```

3. On aggregated Ethernet interface ae0, configure logical interface ae0.1. Configure the logical interface to use VLAN bridge encapsulation, and map the logical interface to VLANs 1. Also, assign an ESI to the logical interface, and enable EVPN multihoming active-active.

```
[edit interfaces]
user@router# set ae0 unit 1 encapsulation vlan-bridge
user@router# set ae0 unit 1 vlan-id 1
user@router# set ae0 unit 1 esi 00:11:22:33:44:55:66:77:88:99
user@router# set ae0 unit 1 esi all-active
```

4. Configure IRB interface irb.1, and assign an IP address to the interface.

```
[edit interfaces]
user@router# set irb unit 1 family inet address 192.0.2.1/24
```

5. Configure an EVPN routing instance named blue. Map the routing instance to VLAN 1, logical interface ae0.1, and IRB interface irb.1. Configure logical interface ae0.1 for the EVPN routing instance.

```
[edit routing-instances]
user@router# set blue instance-type evpn
user@router# set blue vlan-id 1
user@router# set blue interface ae0.1
user@router# set blue routing-interface irb.1
user@router# set blue protocols evpn interface ae0.1
```

6. Configure a VRF routing instance, and add IRB interface irb.1 to the routing instance.

```
[edit routing-instances]
user@router# set vrf instance-type vrf
user@router# set vrf interface irb.1
```

Verification

IN THIS SECTION

 [Verifying that a Logical Interface Has Correct ESI and EVPN Multihoming Mode | 350](#)

● Verifying the EVPN Routing Instance Status | 350

Verifying that a Logical Interface Has Correct ESI and EVPN Multihoming Mode

Purpose

Verify that logical interface ae0.1 is configured with the correct ESI and EVPN multihoming mode.

Action

From operational mode, enter the `show interfaces ae0.1` command.

```
user@switch> show interfaces ae0.1
  Logical interface ae0.1 (Index 332) (SNMP ifIndex 706)
  Flags: Up SNMP-Traps 0x20004000 VLAN-Tag [ 0x8100.100 ] Encapsulation: VLAN-Bridge
  Input packets : 0
  Output packets: 0
  Protocol bridge, MTU: 1522
  Flags: Is-Primary
  Ethernet segment value: 00:11:22:33:44:55:66:77:88:99, Mode: Single-active
```

Meaning

The output shows that logical interface ae0.1 is configured with ESI 00:11:22:33:44:55:66:77:88:99 and the EVPN multihoming mode is single-active for multihoming active-standby mode. For a scenario with multihoming active-active mode, the output would display all-active.

Verifying the EVPN Routing Instance Status

Purpose

Verify the status of the various elements configured in an EVPN routing instance.

Action

From operational mode, enter the `show evpn instance extensive` command.

```
user@switch> show evpn instance extensive
Instance: blue
...
Number of local interfaces: (1 up)
  Interface name  ESI                               Mode           Status    AC-Role
  ae0.1           00:11:22:33:44:55:066:77:88:99  single-active   Up        Root
Number of IRB interfaces: 1 (1 up)
  Interface name  VLAN ID  Status  L3 context
  irb.1           1        Up      vrf
Number of protect interfaces: 0
Number of bridge domains: 1
  VLAN  Domain ID  Intfs / up  IRB intf  Mode           MAC sync  IM route label  SG
sync  IM core nexthop
  1           0      0           Local switching
...
Number of ethernet segments: 1
ESI: 00:11:22:33:44:55:066:77:88:99
Status: Resolved by IFL ae0.1
Local interface: ae0.1, Status: Up/Forwarding
Number of remote PEs connected: 1
  Remote PE      MAC label  Aliasing label  Mode
  10.255.0.1     0          300928          all-active
DF Election Algorithm: MOD based
Designated forwarder: 10.255.0.1
Backup forwarder: 10.255.0.2
Last designated forwarder update: Jul 28 13:04:29
Advertised split horizon label: 300128
```

Meaning

The output shows that for the EVPN routing instance named `blue`, the logical interface `ae0.1` and IRB interface `irb.1` are up and running. It also shows that VLAN 1 and Ethernet segment `00:11:22:33:44:55:066:77:88:99` are properly mapped to the routing instance. It also shows the status of the DF election.

Release History Table

Release	Description
15.1F6	Starting with Junos OS Releases 15.1F6 and 16.1R4 for MX Series routers and Junos OS Release 17.3R1 for EX9200 switches, you can specify an ESI on a logical interface.

RELATED DOCUMENTATION

[EVPN Multihoming Overview](#) | 96

Understanding Automatically Generated ESIs in EVPN Networks

IN THIS SECTION

- [Benefits](#) | 353
- [Automatic ESI Configuration](#) | 353
- [Method 1 Sample Configuration—Automatic ESI on An Aggregated Ethernet Interface](#) | 354
- [Method 2 Sample Configuration—Automatic ESI On Aggregated Ethernet Logical Interfaces](#) | 356
- [Method 3 Sample Configuration—Manual ESI on Aggregated Ethernet Interface and Automatic ESI on Logical Interfaces](#) | 357
- [ESI Value Format](#) | 359

Starting with Junos OS Release 18.4R1, you can configure aggregated Ethernet interfaces and aggregated Ethernet logical interfaces to automatically derive Ethernet segment identifiers (ESIs) from the Link Aggregation Control Protocol (LACP) configuration. We support this feature in the following environments:

- On Juniper Networks devices that support this feature and are multihomed in active-active mode in an EVPN-VXLAN overlay network.
- On Juniper Networks devices that support this feature and are multihomed in active-standby or active-active mode in an EVPN-MPLS overlay network.

This topic includes the following information:

Benefits

- In large EVPN-VXLAN and EVPN-MPLS overlay networks, frees you from manually configuring ESIs.
- Eliminates the possibility of inadvertently configuring the same ESI for multiple Ethernet segments.

Automatic ESI Configuration

In general, you can configure ESIs on aggregated Ethernet interfaces and aggregated Ethernet logical interfaces using the following methods:

- **Method 1**—You can configure automatic ESI on an aggregated Ethernet interface on which LACP is enabled. In this case, an ESI is generated, and that particular ESI is assigned to all logical interfaces on the aggregated Ethernet interface.
- **Method 2**—You can configure automatic ESI on one or more logical interfaces of an aggregated Ethernet interface on which LACP is configured. In this case, an ESI is generated for each logical interface on which the feature is enabled and assigned to that particular logical interface.
- **Method 3**—On an aggregated Ethernet interface on which LACP is enabled, you can manually configure an ESI using the `esi identifier` configuration statement at the `[edit interfaces aeX]` hierarchy level. On one or more logical interfaces on that particular aggregated Ethernet interface, you can configure automatic ESI. In this case, an ESI is generated for each logical interface on which the feature is enabled and assigned to that particular logical interface.

[Table 12 on page 353](#) outlines the automatic ESI configuration options, how to configure each option, and how the ESI is derived for each option.

Table 12: Automatic ESI Configuration Options

Configuration Options	How to Configure Automatic ESI	How ESI Is Derived
Configure automatic ESI on an aggregated Ethernet interface on which LACP is enabled.	Include the <code>auto-derive</code> and <code>lacp</code> configuration statements at the <code>[edit interfaces aeX esi]</code> hierarchy level.	The ESI is derived from the configured values for the <code>system-id</code> and <code>admin-key</code> configuration statements at the <code>[edit interfaces aeX aggregated-ether-options lacp]</code> hierarchy level.

Table 12: Automatic ESI Configuration Options (Continued)

Configuration Options	How to Configure Automatic ESI	How ESI Is Derived
Configure automatic ESI on an aggregated Ethernet logical interface. LACP is enabled on the parent aggregated Ethernet interface.	Include the auto-derive and lacp configuration statements at the [edit interfaces aeX unit <i>logical-unit-number</i> esi] hierarchy level.	The ESI is derived from the configured values for the system-id configuration statement at the [edit interfaces aeX aggregated-ether-options lacp] hierarchy level and the vlan-id configuration statement at the [edit interfaces aeX unit <i>logical-unit-number</i>] hierarchy level. If a logical interface is configured as a trunk interface (interface-mode trunk) and has a VLAN ID list associated with it, the lowest VLAN ID value is used.

When implementing the automatic ESI feature, keep the following in mind:

- In your EVPN-VXLAN or EVPN-MPLS overlay network, you can configure automatic ESI using a mix of method 1, 2, and 3 configuration use cases.
- If a local device is multihomed to two remote devices, we recommend that the aggregated Ethernet and aggregated Ethernet logical interfaces by which the three devices are multihomed have the automatic ESI feature enabled. If the automatic ESI feature is not enabled on one of the interfaces, that interface is not considered during the designated forwarder (DF) election process.
- The automatically generated ESI is supported in both modulo operation- and preference-based DF election processes.
- If you enable the automatic ESI feature and manually configure an ESI on a particular aggregated Ethernet interface or aggregated Ethernet logical interface, you will receive an error when you try to commit the configuration.
- If you enable the automatic ESI feature on an aggregated Ethernet interface and one or more of the logical interfaces on that particular aggregated Ethernet interface, you will receive an error when you try to commit the configuration.

Method 1 Sample Configuration—Automatic ESI on An Aggregated Ethernet Interface

The following example shows the configuration of automatic ESI on aggregated Ethernet interface ae0, which is multihomed in active-active mode. This configuration results in an ESI that is automatically

generated based on the LACP configuration and assigned to logical interfaces ae0.0, ae0.100, ae0.101, and ae0.102.

```

user@mx240> show configuration interfaces
ae0
flexible-vlan-tagging;
encapsulation flexible-ethernet-services;
esi {
    auto-derive { ### Automatic ESI configuration.###
        lacp; ### Automatic ESI configuration.###
    }
    all-active;
}
aggregated-ether-options {
    lacp {
        active;
        system-id 00:00:00:00:11:01; ### ESI derived from this value.###
        admin-key 40; ### ESI derived from this value.###
    }
}
unit 0 {
    encapsulation vlan-bridge;
    vlan-id 10;
}
unit 100 {
    family bridge {
        interface-mode trunk;
        vlan-id-list 100;
    }
}
unit 101 {
    family bridge {
        interface-mode trunk;
        vlan-id-list 101;
    }
}
unit 102 {
    family bridge {
        interface-mode trunk;
        vlan-id-list 102;
    }
}

```

```
}
...
```

Method 2 Sample Configuration—Automatic ESI On Aggregated Ethernet Logical Interfaces

The following example shows the configuration of automatic ESI on aggregated Ethernet logical interfaces ae0.0, ae0.100, ae0.101, and ae0.102, all of which are multihomed in active-active mode. This configuration results in ESIs that are automatically generated based on the LACP and VLAN ID configurations and assigned to each respective logical interface.

```
user@mx240> show configuration interfaces
ae0
flexible-vlan-tagging;
encapsulation flexible-ethernet-services;
aggregated-ether-options {
    lacp {
        active;
        system-id 00:00:00:00:11:01; ### ESI derived from this value.###
    }
}
unit 0 {
    encapsulation vlan-bridge;
    vlan-id 10; ### ESI derived from this value.###
    esi {
        auto-derive { ### Automatic ESI configuration.###
            lacp; ### Automatic ESI configuration.###
        }
        all-active;
    }
}
unit 100 {
    esi {
        auto-derive { ### Automatic ESI configuration.###
            lacp; ### Automatic ESI configuration.###
        }
        all-active;
    }
}
family bridge {
    interface-mode trunk;
    vlan-id-list 100; ### ESI derived from this value.###
}
```

```

    }
}
unit 101 {
    esi {
        auto-derive { ### Automatic ESI configuration.###
            lacp; ### Automatic ESI configuration.###
        }
        all-active;
    }
    family bridge {
        interface-mode trunk;
        vlan-id-list 101; ### ESI derived from this value.###
    }
}
unit 102 {
    esi {
        auto-derive { ### Automatic ESI configuration.###
            lacp; ### Automatic ESI configuration.###
        }
        all-active;
    }
    family bridge {
        interface-mode trunk;
        vlan-id-list 102; ### ESI derived from this value.###
    }
}
...

```

Method 3 Sample Configuration—Manual ESI on Aggregated Ethernet Interface and Automatic ESI on Logical Interfaces

The following example shows the manual configuration of an ESI on aggregated Ethernet interface ae0, and the configuration of automatic ESI on logical interfaces ae0.0, ae0.100, ae0.101, and ae0.102. All interfaces are multihomed in active-active mode. This configuration results in ESI 00:11:22:33:44:55:66:77:88:99 being assigned to ae0, and ESIs that are automatically generated based on the LACP and VLAN ID configurations and assigned to the respective logical interfaces.

```

user@mx240> show configuration interfaces
ae0
flexible-vlan-tagging;
encapsulation flexible-ethernet-services;

```



```

esi 00:11:22:33:44:55:66:77:88:99; ### Manual ESI configuration.###
aggregated-ether-options {
    lacp {
        active;
        system-id 00:00:00:00:11:01; ### Logical interface ESI derived from this value.###
    }
}
unit 0 {
    encapsulation vlan-bridge;
    vlan-id 10; ### Logical interface ESI derived from this value.###
    esi {
        auto-derive { ### Automatic ESI configuration.###
            lacp; ### Automatic ESI configuration.###
        }
        all-active;
    }
}
unit 100 {
    esi {
        auto-derive { ### Automatic ESI configuration.###
            lacp; ### Automatic ESI configuration.###
        }
        all-active;
    }
    family bridge {
        interface-mode trunk;
        vlan-id-list 100; ### Logical interface ESI derived from this value.###
    }
}
unit 101 {
    esi {
        auto-derive { ### Automatic ESI configuration.###
            lacp; ### Automatic ESI configuration.###
        }
        all-active;
    }
    family bridge {
        interface-mode trunk;
        vlan-id-list 101; ### Logical interface ESI derived from this value.###
    }
}
unit 102 {
    esi {

```

```

    auto-derive { ### Automatic ESI configuration.###
        lacp; ### Automatic ESI configuration.###
    }
    all-active;
}
family bridge {
    interface-mode trunk;
    vlan-id-list 102; ### Logical interface ESI derived from this value.###
}
}
...

```

ESI Value Format

When the automatic ESI feature is configured, the aggregated Ethernet and aggregated Ethernet logical interfaces derive the ESIs from various configurations on the aggregated Ethernet interface. The 10-byte format in which the ESI value is stored and propagated is shown in [Figure 19 on page 359](#).

Figure 19: ESI Value Format



The fields of the ESI format are as follows:

- **T**—This field is encoded as 0x01, or Type 1, which indicates that the following 9-octet ESI value is automatically generated from the LACP configuration on the interface.
- **CE LACP System MAC address**—This 6-octet field includes the system MAC address, which is derived from the value of the system-id configuration statement at the [edit interfaces aeX aggregated-ether-options lacp] hierarchy level.
- **CE LACP Port Key**—This 2-octet field includes one of the following:
 - If automatic ESI is configured on an aggregated Ethernet interface, the port key, which is derived from the value of the admin-key configuration statement at the [edit interfaces aeX aggregated-ether-options lacp] hierarchy level.
 - If automatic ESI is configured on an aggregated Ethernet logical interface, the value of the vlan-id configuration statement at the [edit interfaces aeX unit logical-unit-number] hierarchy level.
- **0**—This 1-octet field is encoded as 0x00.

Release History Table

Release	Description
18.4R1	Starting with Junos OS Release 18.4R1, you can configure aggregated Ethernet interfaces and aggregated Ethernet logical interfaces to automatically derive Ethernet segment identifiers (ESIs) from the Link Aggregation Control Protocol (LACP) configuration.

RELATED DOCUMENTATION

EVPN Multihoming Overview 96
auto-derive 1549

EVPN Proxy ARP and ARP Suppression, and NDP and NDP Suppression

IN THIS CHAPTER

- [EVPN Proxy ARP and ARP Suppression, and Proxy NDP and NDP Suppression | 361](#)
- [ARP and NDP Request with a proxy MAC address | 364](#)

EVPN Proxy ARP and ARP Suppression, and Proxy NDP and NDP Suppression

Proxy Address Resolution Protocol (ARP) and ARP suppression, and proxy Neighbor Discovery Protocol (NDP) and NDP suppression are supported as follows:

- MX Series routers and EX9200 switches
 - Starting with Junos OS Release 17.2R1, MX Series routers and EX9200 switches that function as provider edge (PE) devices in an Ethernet VPN-MPLS (EVPN-MPLS) or Layer 3 VXLAN gateways in an Ethernet VPN-Virtual Extensible LAN (EVPN-VXLAN) environment support proxy ARP and ARP suppression, and proxy NDP and NDP suppression on an integrated and routing (IRB) interface.
 - Starting with Junos OS Release 17.4R2, MX Series routers and EX9200 switches support proxy ARP and ARP suppression, and proxy NDP and NDP suppression on non-IRB interfaces. Junos OS Release 17.4R2 also introduces the ability to limit the number of media-access-control (MAC)-IP address bindings that can be learned on these Juniper Networks devices.
- QFX10000 switches
 - Starting with Junos OS Release 17.3R1, QFX10000 switches that function as Layer 3 VXLAN gateways in an EVPN-VXLAN environment support proxy ARP and ARP suppression, and proxy NDP and NDP suppression on an IRB interface.
 - Starting with Junos OS Release 19.1R1, QFX10000 switches that function as Layer 2 VXLAN gateways in an EVPN-VXLAN environment support proxy ARP and ARP suppression, and proxy

NDP and NDP suppression on a non-IRB interface. You can also limit the number of MAC-IP address bindings that can be learned on these switches.

- QFX5100, QFX5200, and QFX5110 switches—Starting with Junos OS Release 18.1R1, QFX5100 and QFX5200 switches that function as Layer 2 VXLAN gateways in an EVPN-VXLAN environment support proxy ARP and ARP suppression, and proxy NDP and NDP suppression on non-IRB interfaces. QFX5110 switches that function as Layer 2 or 3 VXLAN gateways in an EVPN-VXLAN environment support proxy ARP and ARP suppression, and proxy NDP and NDP suppression on IRB interfaces and non-IRB interfaces. You can also limit the number of MAC-IP address bindings that can be learned on these switches.

This feature reduces the flooding of ARP and NDP messages in the EVPN network, resulting in a more efficient use of core bandwidth. By default, proxy ARP and ARP suppression and proxy NDP and NDP suppression are enabled.

When limiting the number of MAC-IP address bindings that can be learned, the limit can be configured globally or for a specific routing instance, bridge domain, VLAN, or interface. After the specified limit is reached, no additional entries are added to the MAC-IP binding database. You can also specify a timeout interval for MAC-IP address bindings.

NOTE: To avoid synchronization issues with MAC and MAC-IP binding entries on QFX5100, QFX5110, and QFX5200 switches in an EVPN-VXLAN environment, Juniper Networks recommends specifying greater MAC aging timer values than ARP aging timer values. For example:

```
set protocols l2-learning global-mac-ip-table-aging-time 600
set protocols l2-learning global-mac-table-aging-time 1200
set system arp aging-timer 10
```

In the sample configuration above, note that the MAC aging timer values are in seconds, and the ARP aging timer value is in minutes.

Proxy ARP and NDP snooping are enabled by default for all EVPN-MPLS or EVPN-VXLAN bridge domains and VLANs. ARP or NDP packets generated from a local customer edge (CE) device or Layer 2 VXLAN gateway are snooped. ARP and NDP packets generated from a remote PE device or Layer 3 VXLAN gateway through core-facing interfaces, however, are not snooped.

Both IRB and non-IRB interfaces configured on a PE device or a Layer 2 or 3 VXLAN gateway deliver ARP requests and NDP requests. When one of these devices receives an ARP request or NDP request, the device searches its MAC-IP address bindings database for the requested IP address. If the device finds the MAC-IP address binding in its database, it responds to the request. If the device does not find the MAC-IP address binding, it takes the following action:

- If the device is running Junos OS Releases 17.2Rx or 17.3Rx, the device swaps the source MAC address with the MAC address of the interface on which the request was received and sends the request to all interfaces.
- If the device is running Junos OS Releases 17.4R1 or later, the device leaves the source MAC address as is and sends the request to all interfaces.

Even when a PE device or a Layer 2 or 3 VXLAN gateway responds to an ARP request or NDP request, ARP packets and NDP might still be flooded across the WAN. ARP suppression and NDP suppression prevent this flooding from occurring.

You can disable the suppression of ARP packets and NDP packets by specifying the `no-arp-suppression` configuration statement. However, if you do so, be aware of the following implications:

- ARP and NDP packets will be flooded.
- The PE device or Layer 2 or 3 VXLAN gateway does not respond to ARP or NDP requests.
- Then PE device or Layer 2 or 3 VXLAN gateway does not learn the IP address from the ARP or NDP request.

Therefore, we recommend that ARP suppression and NDP suppression remain enabled.

Proxy ARP and ARP suppression and proxy NDP and NDP suppression are supported in the following scenarios:

- Single-homed devices in active mode—EVPN-MPLS and EVPN-VXLAN
- Multihomed devices in active-active mode—EVPN-MPLS and EVPN-VXLAN
- Multihomed devices in single-active mode—EVPN-MPLS only

In a multihoming active-active scenario, the database of MAC-IP address bindings are synchronized between the PE device or Layer 2 or 3 VXLAN gateway that act as the designated forwarder (DF) and non-designated forwarder (non-DF).

Release History Table

Release	Description
18.2R1	Starting with Junos OS Release 19.1R1, QFX10000 switches that function as Layer 2 VXLAN gateways in an EVPN-VXLAN environment support proxy ARP and ARP suppression, and proxy NDP and NDP suppression on a non-IRB interface. You can also limit the number of MAC-IP address bindings that can be learned on these switches.

18.1R1	Starting with Junos OS Release 18.1R1, QFX5100 and QFX5200 switches that function as Layer 2 VXLAN gateways in an EVPN-VXLAN environment support proxy ARP and ARP suppression, and proxy NDP and NDP suppression on non-IRB interfaces. QFX5110 switches that function as Layer 2 or 3 VXLAN gateways in an EVPN-VXLAN environment support proxy ARP and ARP suppression, and proxy NDP and NDP suppression on IRB interfaces and non-IRB interfaces. You can also limit the number of MAC-IP address bindings that can be learned on these switches.
17.4R2	Starting with Junos OS Release 17.4R2, MX Series routers and EX9200 switches support proxy ARP and ARP suppression, and proxy NDP and NDP suppression on non-IRB interfaces.
17.4R2	Junos OS Release 17.4R2 also introduces the ability to limit the number of media-access-control (MAC)-IP address bindings that can be learned on these Juniper Networks devices.
17.3R1	Starting with Junos OS Release 17.3R1, QFX10000 switches that function as Layer 3 VXLAN gateways in an EVPN-VXLAN environment support proxy ARP and ARP suppression, and proxy NDP and NDP suppression on an IRB interface.
17.2R1	Starting with Junos OS Release 17.2R1, MX Series routers and EX9200 switches that function as provider edge (PE) devices in an Ethernet VPN-MPLS (EVPN-MPLS) or Layer 3 VXLAN gateways in an Ethernet VPN-Virtual Extensible LAN (EVPN-VXLAN) environment support proxy ARP and ARP suppression, and proxy NDP and NDP suppression on an integrated and routing (IRB) interface.

RELATED DOCUMENTATION

[global-mac-ip-limit](#) | 1582

[interface-mac-ip-limit](#) | 1600

[mac-ip-table-size](#) | 1619

[no-arp-suppression](#) | 1630

ARP and NDP Request with a proxy MAC address

IN THIS SECTION

- [Benefits of using a virtual gateway or IRB MAC address in an ARP request](#) | 369

Starting in Junos OS Release 19.1R1, PE devices support the substitution of a source MAC address with a proxy MAC address in the ARP or NDP reply. When the PE device receives an ARP or NDP request, the PE device searches the MAC-IP address binding database and if there is an entry, it replaces the source MAC address with the proxy MAC address in the ARP reply.

For EVPN networks with an edge-routed bridging overlay, you can enable proxy MAC address using the `irb` option. Junos OS uses the virtual gateway MAC address as the proxy MAC address on each leaf device. When virtual gateway is not configured, Junos uses the IRB MAC address as the proxy MAC address on the leaf device.

If the entry is not found in the MAC-IP address binding database in an edge-routed bridging EVPN network, the source MAC and IP address in the ARP request are replaced with the proxy MAC and IP address of the IRB interface and the ARP request is flooded.

In edge-routed bridging, Junos specifies the proxy MAC address using the following precedence order:

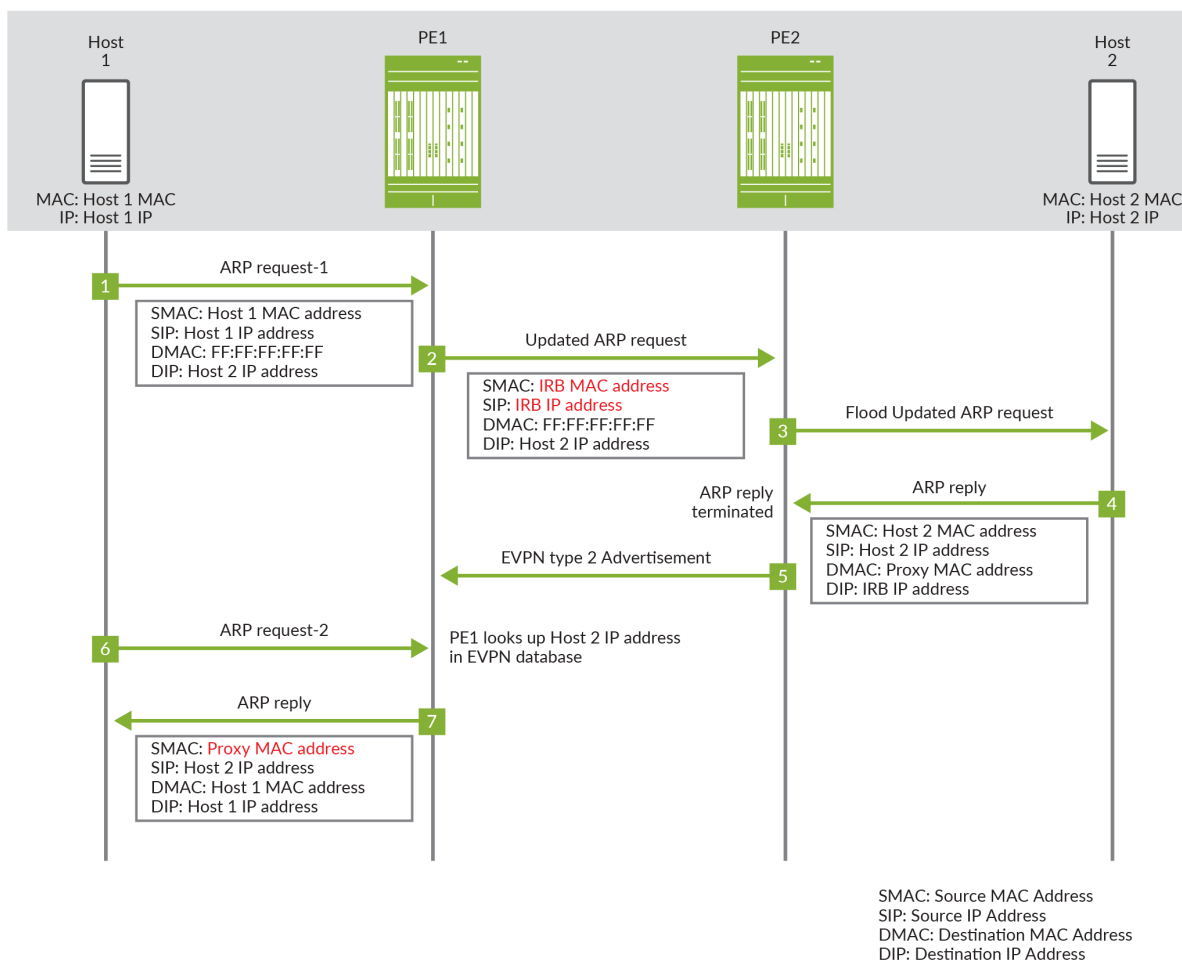
1. A configured virtual gateway MAC address.
2. An automatically derived virtual gateway MAC address.
3. A configured IRB MAC address.
4. Automatically assigned MAC address based on the physical interface when neither the virtual gateway MAC address nor the IRB MAC address is configured.

Figure 20 on page 366 illustrates how proxy MAC address is applied in an edge-routed bridging EVPN network. The sequence of events for an ARP request is as follows:

1. Host 1 sends the first ARP request message to locate Host 2. The ARP request message has the MAC and IP address of host 1 as the source MAC and IP address.
2. Device PE1 receives the ARP request message. Since there are no matching entries in its MAC-IP address binding database, PE1 appropriates the ARP request message and replaces the MAC and IP address of Host 1 with the MAC and IP address of the IRB interface.
3. Device PE2 forwards the ARP request message.
4. Host 2 sends the ARP response with its MAC and IP address.
5. Device PE2 adds an entry for Host 2 to its MAC-IP address binding database and sends an EVPN type 2 advertisement message for Host 2. When PE1 receives the EVPN type 2 message, it adds an entry for Host 2 to its MAC-IP address binding database.
6. ARP request 1 times out and host 1 sends another ARP request message.

7. Device PE1 receives the second ARP request message. Since there is an entry in the MAC-IP address binding database, it responds as the ARP proxy with the proxy MAC address.

Figure 20: Proxy MAC ARP Request in an Edge-Routed Bridging EVPN Network

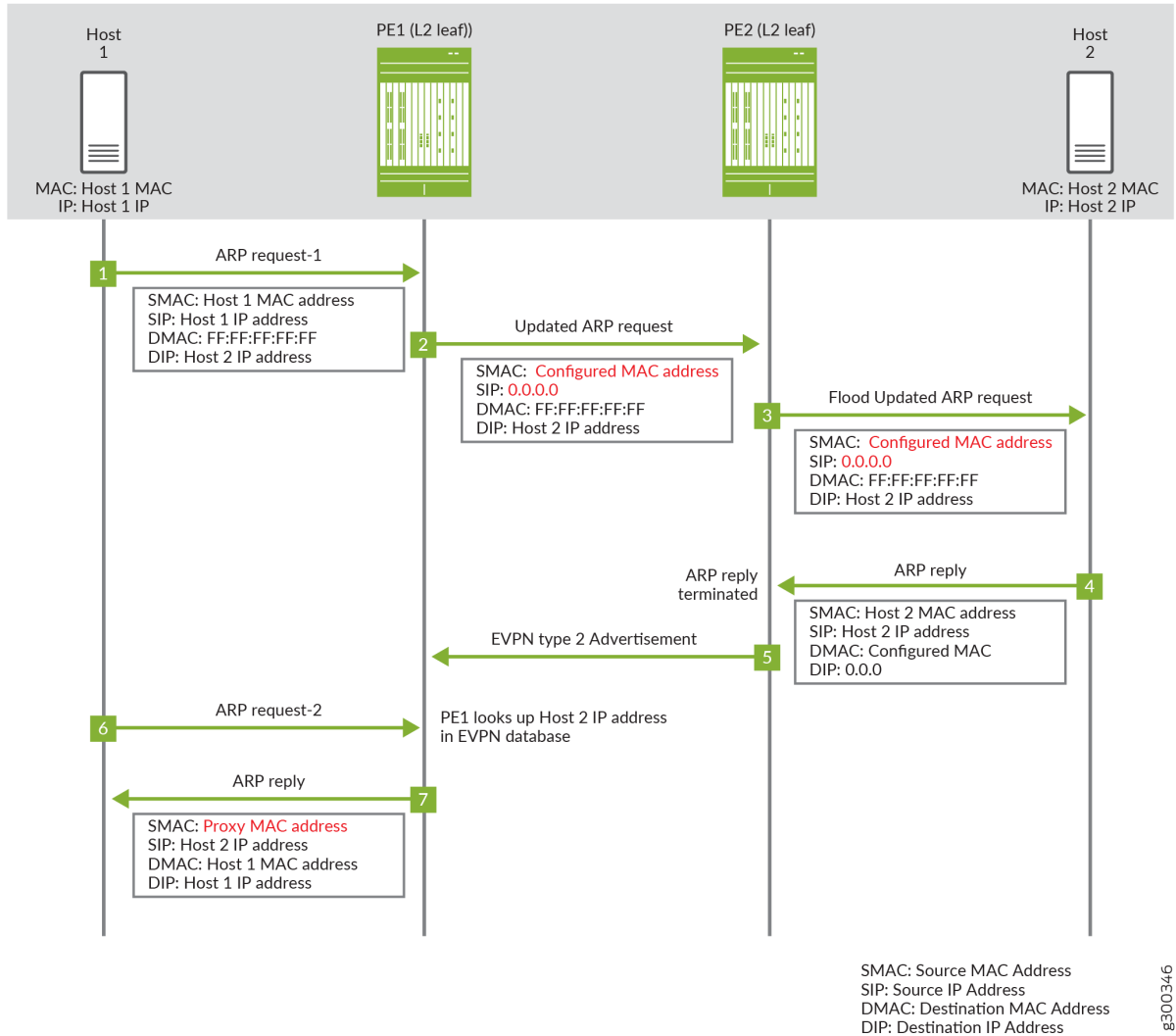


For EVPN networks with a centrally-routed bridging overlay, you assign a configured MAC address as the proxy MAC address. The configured proxy MAC address on the leaf should be the virtual gateway MAC address of the IRB on spine device or gateway. When the virtual gateway MAC address is not configured, configure the proxy MAC address to use the IRB MAC address on the centralized gateway.

If the entry is not found in the MAC-IP address binding database in a centrally-routed bridging EVPN network, the source MAC is replaced with the configured proxy MAC and the source IP address is set to 0.0.0.0. in the ARP request. [Figure 21 on page 367](#) illustrates how proxy MAC address is applied in a

centrally-routed bridging EVPN Network where Host 1 and Host 2 are connected to leaf devices PE1 and PE2 respectively. The sequence of events for the ARP request is as follows:

Figure 21: Proxy MAC ARP Request in a Centrally-routed Bridging EVPN Network



1. Host 1 sends the first ARP request message to locate Host 2. The ARP request message has the MAC and IP address of Host 1 as the source MAC and IP address.
2. Leaf device PE1 receives the ARP request message. Since there are no matching entries in its MAC-IP address binding database, PE1 floods the ARP request message and replaces the source MAC address of Host 1 with the configured proxy MAC address and a source IP address to 0.0.0.0.
3. Leaf device PE2 receives and forwards the ARP request message.

4. Host 2 sends the ARP response with its MAC and IP address.
5. Device PE2 adds an entry for Host 2 to its MAC-IP address binding database and sends an EVPN type 2 advertisement message for Host 2. When PE1 receives the EVPN type 2 message, it adds an entry for Host 2 to its MAC-IP address binding database.
6. ARP request 1 times out and host 1 sends another ARP request message.
7. Device PE1 receives the second ARP request message. Since there is an entry in the MAC-IP address binding database, it responds as the ARP proxy with the proxy MAC address.

This feature ensures all inter-VLAN and intra-VLAN traffic are forwarded as Layer 3 traffic to the configured gateway and that the traffic can be routed.

To enable this feature, include the `proxy-mac (irb | proxy-mac-address)` statement in the `[edit routing-instances routing-instance-name protocols evpn]` hierarchy for the evpn instance type or in the `[edit routing-instances routing-instance-name bridge-domains domain_name]` hierarchy for the virtual-switch instance type.

In an EVPN network with edge-routed bridging overlay where CE devices are directly connected to a L3 gateway device, configure `proxy-mac` on the gateway devices with the `irb` option.

In an EVPN network with a centrally-routed bridging overlay where CE devices are connected to an L3 gateway device via layer 2 only leaf devices, configure `proxy-mac` on the leaf devices with the `proxy-mac-address` option using the virtual or physical gateway MAC address on the centralized L3 gateway IRB interface as the MAC address.

The following is a sample configuration for bridge domain with support for proxy MAC for an edge-routed bridging EVPN network.

```
routing-instances {
  VS-1 {
    instance-type virtual-switch;
    bridge-domains {
      bd-110 {
        interface ae0.0;
        vlan-id 110;
        routing-interface irb.5;
        proxy-mac {
          irb;
        }
      }
    }
  }
}
```

Benefits of using a virtual gateway or IRB MAC address in an ARP request

With this feature, all inter-VLAN and intra-VLAN traffic for all configured routing instances will be routed to an IRB interface. This allows the following:

- Communication between private VLANs.
- Layer 3 security filtering at the gateway for both intra-VLAN and inter-VLAN traffic.
- MAC address privacy. Host do not learn the MAC addresses of other hosts.

RELATED DOCUMENTATION

[EVPN Proxy ARP and ARP Suppression, and Proxy NDP and NDP Suppression](#) | 361

[proxy-mac](#) | 1644

Configuring DHCP Relay Agents

IN THIS CHAPTER

- [DHCP Relay Agent in EVPN-MPLS Network | 370](#)
- [DHCP Relay Agent over EVPN-VXLAN | 373](#)

DHCP Relay Agent in EVPN-MPLS Network

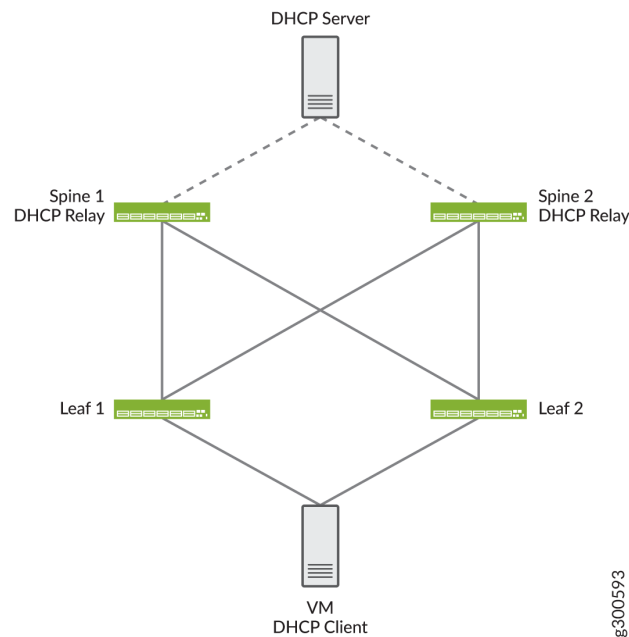
IN THIS SECTION

- [Benefits of DHCP Relay Agent | 372](#)
- [Configuring DHCP Relay Agents in EVPN-MPLS | 372](#)

Dynamic Host Configuration Protocol (DHCP) is a protocol that allows a DHCP server to dynamically allocate IP addresses to DHCP clients. The DHCP relay agent forward DHCP messages between DHCP clients and DHCP servers when they are on different networks. Junos supports configuring the DHCP relay agent on the spine devices in a centrally bridging overlay on an EVPN-MPLS network.

Figure 22 on page 371 shows a centrally-routed bridging EVPN-MPLS network with a virtual machine on a datacenter server multihomed to leaf 1 and leaf 2. The leaf devices are connected to spine devices that function as DHCP relays and forwards the DHCP messages to the DHCP server.

Figure 22: DHCP Relay Agent on Centrally-Routed Bridging Overlay



The following sequence describes the interaction between the DHCP client, DHCP relay agent, and DHCP server.

1. The DHCP client (VM) initiates a request by sending a DHCP discovery message to find a DHCP server and obtain a lease for an IP address. DHCPv6 clients send DHCPv6 solicit messages.
2. Leaf 1 broadcasts the new locally-learned MAC address and DHCP discovery (or DHCPv6 solicit) message to the other devices in the EVPN core.
3. Spine 1 forwards the DHCP discovery (or DHCPv6 solicit) message to DHCP server.
4. Upon receipt of the DHCP discovery (or DHCPv6 solicit) message, the DHCP server issues a DHCP offer message (or DHCPv6 advertisement) message with the IP address back to Spine 1.
5. Spine 1 forwards the DHCP offer (or DHCPv6 advertisement) message to Leaf 1 where the message is sent back to the client.

Subsequent DHCP request (or DHCPv6 request) and DHCP acknowledgment (or DHCPv6 reply) messages are similarly forwarded. For DHCP renew and DHCP release messages, the DHCP client sends

unicast messages to renew or release their IP address. The DHCP V6 clients send multicast messages to renew or release their IP address.

Benefits of DHCP Relay Agent

DHCP allows you to manage IP addresses and other network configurations easily. The DHCP relay on the spine devices of a centrally routed bridging overlay allows you to simplify and centralize the routing of the DHCP messages.

Configuring DHCP Relay Agents in EVPN-MPLS

To configure the DHCP relay agent on the Junos OS device, include the `dhcp-relay` statement at the `[edit forwarding-options]` hierarchy level for either DHCP or DHCPv6 services.

To configure the DHCP server, create a server group with at least one DHCP server IP address on the DHCP relay agent and apply the server-group as a member of an active-server-group.

To configure a group of DHCP server addresses and define them as an active server group:

1. Specify the name of the server group.

```
[edit forwarding-options dhcp-relay]
user@host# set server-group server-group-name
```

2. Add the IP addresses of the DHCP servers belonging to the group.

```
[edit forwarding-options dhcp-relay server-group server-group-name]
user@host# set ip-address1
user@host# set ip-address2
```

3. Define the server group as an active server group.

```
[edit forwarding-options dhcp-relay]
user@host# set active-server-group server-group-name
```

The following configuration snippet shows a sample DHCP relay configuration.

```
forwarding-options {
  dhcp-relay {
    server-group {
```

```

        dhcp-server {
            10.0.0.3;
        }
    }
}
active-server-group dhcp-server;
group v4-relay {
    interface irb.0;
}
}

```

NOTE: DHCP relay agent on EVPN-MPLS does not support active leasequery (ALQ).

RELATED DOCUMENTATION

[active-server-group](#)

[dhcp-relay](#)

[server-group](#)

DHCP Relay Agent over EVPN-VXLAN

IN THIS SECTION

- [DHCP Relay on a Centrally-routed Bridging Overlay | 374](#)
- [DHCP Relay on an Edge-routed Bridging Overlay | 375](#)

Dynamic Host Configuration Protocol (DHCP) is a protocol that enables a DHCP server to dynamically allocate IP addresses to DHCP clients. This allows you to manage IP addresses and other network configurations easily. The DHCP relay agent forward DHCP messages between DHCP clients and DHCP servers when they are on different networks. Junos supports configuring the DHCP relay agent in an EVPN-VXLAN fabric.

In an EVPN-VXLAN fabric, the DHCP server and the DHCP clients connect into the network using access interfaces on leaf devices. The DHCP server and clients can communicate with each other over the existing network without further configuration when the DHCP client and server are in the same VLAN. When a DHCP client and server are in different VLANs, DHCP traffic between the client and server is forwarded between the VLANs through integrated routing and bridging (IRB) interfaces.

You can configure DHCP relay on a centrally-routed bridging overlay or on an edge-routed bridging overlay in an EVPN-VXLAN fabric. The main difference between the overlay models for DHCP relay configuration is the IRB interfaces. In a centrally-routed bridging overlay, these IRB interfaces are on the spine devices, while in an edge-routed bridging overlay, they are on the leaf devices.

NOTE: DHCP relay is supported in EVPN-VXLAN fabrics with IPv6 underlay. For information on IPv6 underlay, see ["EVPN-VXLAN with an IPv6 Underlay" on page 643](#).

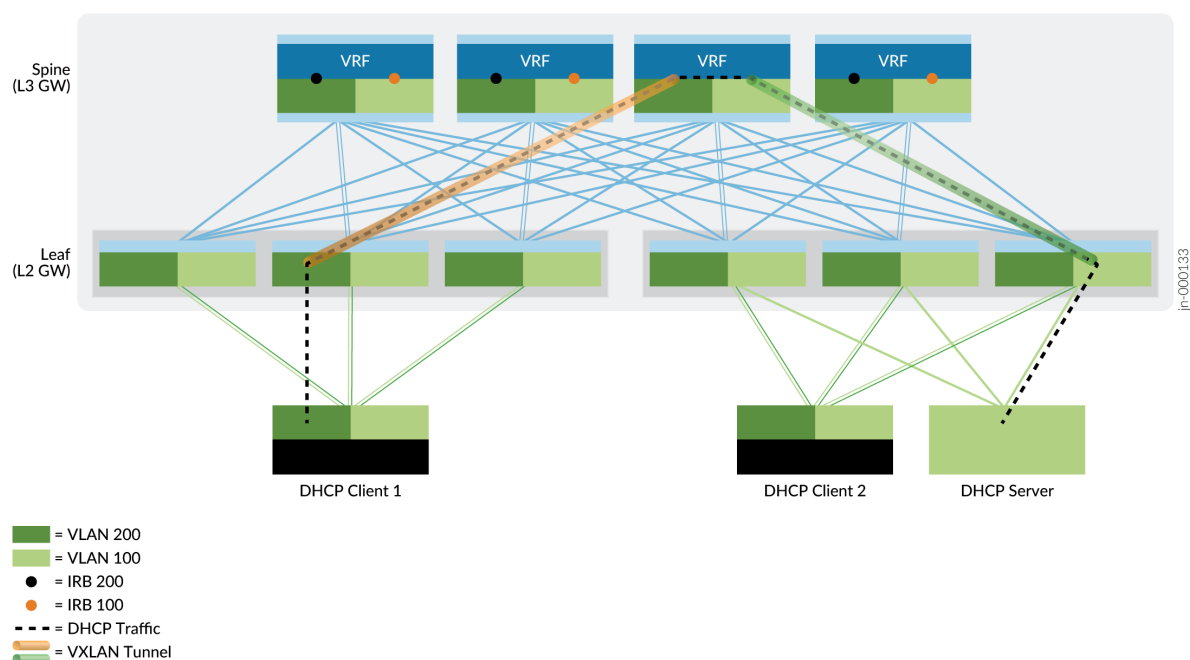
DHCP Relay on a Centrally-routed Bridging Overlay

In a centrally-routed bridging overlay, the spine devices function as Layer 3 gateways that handle traffic between VXLANs while the leaf devices function as Layer 2 gateways that handle traffic within a VXLAN. DHCP clients are connected to pure Layer 2 EVPN leaf devices. DHCP packets are sent over the EVPN core towards spine devices, which function as DHCP relay agents, forwarding the packets to the servers. Configuring DHCP relay on the spine devices of a centrally-routed bridging overlay allows you to simplify and centralize the routing of the DHCP messages.

[Figure 23 on page 375](#) shows how DHCP traffic is forwarded in a centrally-routed bridging overlay where the DHCP client and server connect to access interfaces on different leaf devices in different

VLANs. For information on how to configure DHCP relay in a centrally-routed bridging overlay, see [DHCP Relay Design and Implementation](#).

Figure 23: DHCP Relay Agent on Centrally-Routed Bridging Overlay



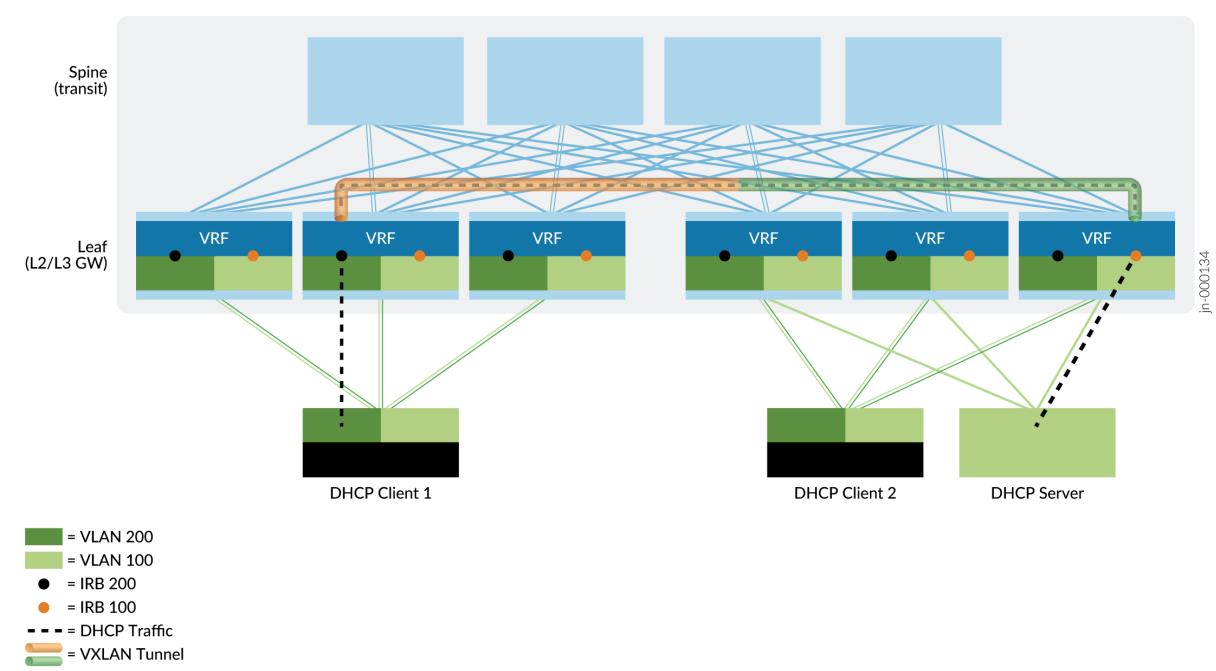
DHCP Relay on an Edge-routed Bridging Overlay

In an edge-routed bridging overlay, the leaf devices serve as both Layer 2 and Layer 3 gateways. All leaf devices support IRB functionality and function as VTEPs to establish VXLAN tunnels. Spine devices are configured to handle only IP traffic, which removes the need to extend the bridging overlays to the spine devices.

[Figure 24 on page 376](#) shows how DHCP traffic is forwarded in an edge-routed bridging overlay where the DHCP client and server connect to access interfaces on different leaf devices in different VLANs.

For more information on configuring DHCP relay in an edge-routed bridging overlay, see [Configure a DHCP Relay in EVPN-VXLAN Fabric Architecture](#).

Figure 24: DHCP Relay Agent on Edge-Routed Bridging Overlay



Configuring MAC Pinning

IN THIS CHAPTER

- [EVPN MAC Pinning Overview | 377](#)
- [Configuring EVPN MAC Pinning | 379](#)
- [Creating an Exclusion List for MAC Pinning | 379](#)

EVPN MAC Pinning Overview

Starting in Release 16.2, Junos OS enables MAC address pinning for Ethernet VPN (EVPN), including customer edge (CE) interfaces and EVPN over MPLS core interfaces, in both all-active mode or active-standby mode.

When you configure an interface with MAC pinning, the l2ald process adds an 8-octet extension to the address (which implements aspects of Section 7.7 of RFC-7432: MAC Mobility Extended Community). The low-order bit in the flag octet of this structure is the Sticky/static flag. Configuring MAC pinning sets the flag to 1 and designates the address is static.

If you configure MAC pinning on a CE interface, that MAC address cannot be moved to any other CE interface. Similarly, if you configure MAC pinning on an MPLS core interface on a PE device, that MAC address cannot be moved to a different interface on the MPLS core.

CE devices advertise MAC pinned addresses through the l2ald process to remote PE devices. When a PE device learns a MAC-pinned interface address from a CE device, the PE device synchronizes the address, through the control plane, with remote peer PE devices in the EVPN network. Thereafter, if the PE device receives traffic, or an advertised route, from any CE device in the EVPN network that bears the same source MAC address, the receiving device drops the traffic.

A PE device that learns a MAC pinned address via its control plane prefers that address over an address bearing the identical MAC address that is learned from a remote peer PE device, or locally by way of its l2ald from a CE interface.

Before the introduction of MAC address pinning for EVPN, a MAC address on a remote PE device that was learned locally could be aged out. For EVPN MAC pinning, a pinned MAC address does not age out

on the remote PE device unless the routing protocol process removes it from the routing table. Similarly, a pinned MAC address persists for the control plane of the remote PE device.

This static interface address cannot be moved unless it is deleted from the routing table of the device on which it is configured.

A MAC address might be considered moved as a result of:

- Misconfiguration
- Configuration on a different Ethernet segment
- Physical movement of a device within a network topology

NOTE: If EVPN is configured in all-active multihoming mode, you must either enable or disable MAC pinning on the multihoming PE device interfaces in the broadcast domain. Also, either enable or disable MAC pinning on all CE device interfaces to avoid inconsistent MAC learning in the EVPN broadcast domain.

If EVPN is configured in active-standby multihoming mode, a MAC pinned address received by the active PE device can be moved to a CE interface on the standby PE device in response to a switchover, if traffic has been running continuously on the CE interface.



CAUTION: Do not enable or disable MAC pinning on an interface while traffic is running.

Release History Table

Release	Description
16.2	Starting in Release 16.2, Junos OS enables MAC address pinning for Ethernet VPN (EVPN), including customer edge (CE) interfaces and EVPN over MPLS core interfaces, in both all-active mode or active-standby mode.

RELATED DOCUMENTATION

[EVPN Overview](#) | 874

[Configuring EVPN Routing Instances](#) | 10

Understanding MAC Pinning

[mac-pinning \(EVPN Routing Instances\)](#) | 1620

Configuring EVPN MAC Pinning

The EVPN MAC pinning feature enables you to apply MAC pinning to CE/VPLS interfaces in a bridge domain and to PE interfaces in the core of a service-provider network.

The EVPN MAC pinning feature enables you to make the MAC address associated with a specific interface static. That is, the MAC-pinned address cannot be moved to any other interface in the bridge domain. The MAC-pinned addresses on the interface ages out at the same interval as the default MAC table timeout interval for an MX Series router.

Traffic transmitted with pinned MAC address as source MAC, from any interface in the EVPN domain other than the one on which the MAC address is pinned, will be discarded.

Pinning a MAC address to an interface for EVPN does not require new CLI. Existing CLI enables pinning MAC addresses to CE-device and PE-device interfaces. You can enable MAC Pinning on an interface in an EVPN network with the following command:

```
User@CE1# set switch-options interface interface-name mac-pinning
```

RELATED DOCUMENTATION

[EVPN Overview | 874](#)

[Configuring EVPN Routing Instances | 10](#)

[EVPN MAC Pinning Overview | 377](#)

[mac-pinning \(EVPN Routing Instances\) | 1620](#)

[Configuring the MAC Table Timeout Interval](#)

Creating an Exclusion List for MAC Pinning

SUMMARY

Use an exclusion list to exclude MAC addresses from being pinned in an EVPN network.

IN THIS SECTION

- [Benefits of Using an Exclusion List for MAC pinning | 381](#)

MAC pinning allows you to control the movement of MAC addresses and prevents the creation of network loops by pinning a virtual machine's MAC address to an interface. When you enable MAC

pinning on an interface in an EVPN network, the MAC addresses that are learned on the interface are identified as pinned MAC addresses on that interface and in the MAC advertisement message. This prevents virtual machines (MAC addresses) from being moved to another interface in the EVPN network. However, in some cases, you might not want to pin *all* the MAC addresses for an interface; instead, you might want to exclude a few MAC addresses. For example, the Virtual Router Redundancy Protocol (VRRP) provides redundancy with the primary and backup router sharing a virtual MAC address. The network needs to know when the VRRP virtual MAC address has moved from the primary VRRP router to the backup VRRP router, so in this case, you would want to exclude the VRRP virtual MAC address from being pinned.

While MAC pinning is enabled separately on individual interfaces, the exclusion list is configured globally on the device. When you configure an exclusion list, the l2ald process verifies the newly learned addresses on the interface against the MAC addresses on the exclusion list. Addresses that are not on the exclusion list are identified as pinned MAC addresses. Addresses that are in the exclusion list are identified as dynamically learned MAC addresses. When the device sends the MAC IP advertisement route message to other devices, the pinned MAC addresses will be identified with the static flag in the extended community set to 1.

When you add a MAC address to the exclusion list that was previously identified as a pinned address, the l2ald process removes the pinned MAC address from the MAC address tables, adds it back in as a dynamic nonpinned MAC address, and sends an updated MAC route advertisement messages to the other devices. A similar process happens when you remove a MAC address from the exclusion list.

To configure an exclusion list, include a list of MAC addresses with the `exclusive-mac` parameter at the `[edit protocols l2-learning global-mac-move]` hierarchy level.

For example, if you want to set an exclusion list for MAC addresses 00:00:5E:00:01:01 and 00:00:5E:00:01:20, you include the following configuration. The output for `show bridge mac-table` displays the following

```
User@PE1# set protocols l2-learning global-mac-move exclusive-mac
00:00:5E:00:01:01
User@PE1# set protocols l2-learning global-mac-move exclusive-mac
00:00:5E:00:01:20
```

To remove a MAC address from the exclusion list, use the `delete` configuration mode command at the `[edit protocols l2-learning global-mac-move]` hierarchy. For example, `delete protocols l2-learning global-mac-move exclusive-mac 00:00:5E:00:01:01`.

The following show bridge mac-table output shows how MAC addresses are learned by other PE devices and identifies the excluded MAC addresses and pinned MAC addresses.

```
User@PE1> show bridge mac-table
MAC flags      (S -static MAC, D -dynamic MAC, L -locally learned, C -Control MAC
               O -OVSDB MAC, SE -Statistics enabled, NM -Non configured MAC, R -Remote PE MAC, P -Pinned
               MAC)

Routing instance : EVPN
Bridging domain : bd100, VLAN : 100

  MAC          MAC      Logical      NH      MAC      active
  address      flags    interface  Index  property  source
  00:00:5E:00:01:01  DL      ge-0/0/1.0                #Excluded MAC
Address
  00:50:56:93:f8:ff  DC                1048575      10.1.1.3
  00:50:56:93:c2:60  DC                1048575
10.1.1.3
  00:50:56:93:d9:fb  DP      ge-0/0/1.0                #Pinned MAC Address
```

The following features supports exclusion lists for EVPN MAC pinning:

- EVPN-MPLS, EVPN-VXLAN, EVPN, ELAN, and EVPN E-tree.
- EVPN routing instances and virtual-switch routing instances.
- All-active and single-active EVPN routing instances.
- MAC mobility extended community support for EVPN Type 5 routes.
- Static MAC addresses.
- MC-LAG.

Benefits of Using an Exclusion List for MAC pinning

Exclusion lists allows you to have more flexibility and more control in configuring devices and interfaces on your network.

RELATED DOCUMENTATION

[EVPN MAC Pinning Overview](#) | 377

Configuring EVPN MAC Pinning | **379**

exclusive-mac | **1570**

global-mac-move | **1585**

mac-pinning (EVPN Routing Instances) | **1620**

High Availability in EVPN

IN THIS CHAPTER

- [NSR and Unified ISSU Support for EVPN | 383](#)
- [Preventing Traffic Loss in an EVPN/VXLAN Environment With GRES and NSR | 386](#)
- [Graceful Restart in EVPN | 387](#)

NSR and Unified ISSU Support for EVPN

IN THIS SECTION

- [Supported Features on EVPN | 385](#)

Starting in Release 16.2, Junos OS ensures minimal loss of traffic when a Routing Engine switchover occurs with nonstop active routing (NSR) and graceful Routing Engine switchover (GRES) enabled. The forwarding state of the Packet Forwarding Engine (PFE) remains intact during switchover. The signaling state on the primary Routing Engine and on the standby Routing Engine are built in parallel.

EVPN reproduces dynamically generated data (such as labels and sequence numbers), and data obtained from peers on the primary Routing Engine, and on the standby Routing Engine. EVPN also monitors BGP ingress and egress routing table messages on the standby Routing Engine to populate its signaling plane data structures. Local MAC addresses are obtained by the Layer 2 address learning process (l2ald), which transfers the data to the EVPN module in the route processing software. In the network layer reachability information (NLRI) fields of its packets, BGP transfers the MAC addresses to peers in the network.

In earlier releases, the l2ald did not run on the standby Routing Engine. Consequently, the l2ald did not inform the routing protocol process on the standby Routing Engine about locally learned MAC addresses. With this feature, the routing protocol process learns of newly acquired MAC addresses by listening to the BGP ingress and egress routing table messages and then populates its data structures.

Following a switchover, when the I2ald becomes active on the new primary Routing Engine, it sends all locally-learned MAC addresses to the routing protocol process, which can then verify the MAC addresses learned from the I2ald with MAC addresses derived from the BGP routing table egress messages.

NOTE: Expect a traffic loss pertaining to a topology change if the topology change occurs during a switchover.

This feature mirrors the following data to the standby Routing Engine:

- MAC route labels—EVPN allocates a MAC route label per EVPN instance (EVI) and per Ethernet segment Identifier (ESI). MAC labels, including the EVI and ESI are mirrored to the standby Routing Engine.
- Inclusive multicast (IM) route labels—EVPN allocates an IM label per VLAN. An IM label, including the EVI name and VLAN ID are mirrored to the standby Routing Engine.
- Split horizon labels—EVPN mirrors a split horizon label and the EVI name to the standby Routing Engine.
- Aliasing labels—EVPN mirrors an aliasing label and EVI name to the standby Routing Engine.
- Dummy labels—EVPN creates a dummy label with which it adds the egress route to the mpls.0 table. EVPN mirrors a dummy label and the EVI name to the standby Routing Engine so the mpls.0 table is identical on the primary and standby Routing Engines.

For EVPN ETREE, Junos mirrors the local EVPN ETREE leaf label that is advertised to other PE as part of the ETREE extended community on the standby Routing Engine.

For EVPN P2MP, Junos mirrors the LDP/RSVP transport label on the standby Routing Engine.

NSR Data Flow Pre-Switchover

The EVPN configuration on the standby Routing Engine directs it to operate identically to the primary Routing Engine except that it does not allocate any labels. Label information is updated on the standby Routing Engine when it receives the information from its label information base mirror (libmirror).

BGP updates remote MAC addresses in the instance.EVPN table on the standby Routing Engine. Just as EVPN operates on the primary Routing Engine, the standby Routing Engine derives information from its flash drive to update its data structures.

To gather local MAC addresses and update data structures, EVPN on the primary Routing Engine syncs its MAC address information with the standby routing engine. Following a switchover, when the I2ald becomes active, if it sends MAC addresses that are identical to those marked stale, the addresses are

retained. If the I2ald does not send the same MAC addresses learned from BGP RIB egress messages, it deletes the addresses.

NSR Data Flow Post Switchover

Following a switchover, when the standby Routing Engine becomes the primary Routing Engine, it establishes connection with the I2ald. When the I2ald becomes active, it sends local MAC addresses to the routing protocol process. The routing protocol process removes the stale bit from the MAC addresses it receives from the I2ald.

MAC addresses that were not derived from BGP RIB egress messages, but were sent to the routing protocol process by the I2ald, are treated as newly learned MAC addresses. BGP advertises such MAC addresses.

When the I2ald sends the end marker of the MAC address to the routing protocol process, all local MAC addresses marked as stale are deleted.

Unified ISSU Support

Starting in Junos OS Release 17.2R1, unified in-service software upgrade is supported for VXLAN on MX Series routers. ISSU enables you to upgrade your Junos OS software on your MX Series router with no disruption on the control plane and with minimal disruption of traffic. Unified ISSU is supported only on dual Routing Engine platforms. In addition, the graceful Routing Engine switchover (GRES) and nonstop active routing (NSR) features must be enabled. Unified ISSU allows you to eliminate network downtime, reduce operating costs, and deliver higher levels of services. See [Getting Started with Unified In-Service Software Upgrade](#).

To enable GRES, include the `graceful-switchover` statement at the `[edit chassis redundancy]` hierarchy level.

To enable NSR, include the `nonstop-routing` statement at the `[edit routing-options]` hierarchy level and the `commit synchronize` statement at the `[edit system]` hierarchy level.

Supported Features on EVPN

Junos OS supports NSR/ISSU on the following features for EVPN:

Feature	Initial Junos OS Release
EVPN with ingress replication for BUM traffic	16.2R1
EVPN-ETREE	17.2R1
EVPN-VPWS	17.2R1

(Continued)

Feature	Initial Junos OS Release
EVPN -VXLAN	17.2R1
PBB-EVPN	17.2R1
EVPN with P2MP mLDP replication for BUM traffic	18.2R1

Release History Table

Release	Description
17.2R1	Starting in Junos OS Release 17.2R1, unified in-service software upgrade is supported for VXLAN on MX Series routers.
16.2	Starting in Release 16.2, Junos OS ensures minimal loss of traffic when a Routing Engine switchover occurs with nonstop active routing (NSR) and graceful Routing Engine switchover (GRES) enabled.

RELATED DOCUMENTATION[show evpn instance | 1834](#)[show evpn database | 1819](#)[Tracing EVPN Traffic and Operations | 42](#)[Understanding Graceful Routing Engine Switchover](#)**Preventing Traffic Loss in an EVPN/VXLAN Environment With GRES and NSR**

The graceful Routing Engine switchover (GRES) feature in Junos OS enables QFX10000 switches to continue forwarding packets, even if one Routing Engine fails. To preserve routing during a switchover, GRES is combined with nonstop active routing (NSR). Although GRES preserves interface and kernel information, in an EVPN/VXLAN environment, a QFX10000 switch could experience Layer 3 traffic loss if traffic is flowing during the switchover or reboot of the Routing Engine.

To prevent traffic flows from being dropped in such a scenario, configure a hold-time interval that prevents IP phantom addresses from being added to the routing tables. During this interval, the routes are maintained in the kernel. After the interval expires, the phantom routes are added to the appropriate routing tables before they are deleted from the kernel.

NOTE: The recommended hold-time value in an EVPN/VXLAN environment is 300 seconds (5 minutes).

- Enable GRES
- Enable NSR

To specify a hold-time interval that prevents phantom routes from being added to the routing table during a switchover until the interval expires:

Specify a hold-time interval of 300.

```
[edit routing-options]
user@swtich# set nsr-phantom-holdtime seconds 300.
```

NOTE: The hold-time interval range is 0 through 10000.

RELATED DOCUMENTATION

[Understanding Graceful Routing Engine Switchover](#)

Graceful Restart in EVPN

Starting in Junos OS 18.4R1 graceful restart is supported on devices in an EVPN-VXLAN network. Graceful restart allows a routing engine undergoing a switchover without NSR to inform its adjacent neighbors and peers of its condition. When you enable graceful-restart, the routing protocol process learns of newly acquired MAC addresses by listening to the BGP ingress and egress routing table messages. It then verifies MAC addresses learned from the I2ald, with MAC addresses derived from the BGP routing table egress messages. Graceful restart enables the device to reduce the overall network traffic loss while the device is passing through the intermediate convergence states that occurs during a restart.

To enable graceful restart, include the `graceful-restart` statement at the `[edit routing-options]` hierarchy level.

RELATED DOCUMENTATION

| [Graceful Restart Concepts](#)

Monitoring EVPN Networks

IN THIS CHAPTER

- [Connectivity Fault Management Support for EVPN and Layer 2 VPN Overview | 389](#)
- [Configuring a MEP to Generate and Respond to CFM Protocol Messages | 391](#)

Connectivity Fault Management Support for EVPN and Layer 2 VPN Overview

IN THIS SECTION

- [Limitations of CFM on layer 2 VPN and EVPNs | 390](#)

The IEEE 802.1ag specification provides for Ethernet connectivity fault management (CFM). The goal of CFM is to monitor an Ethernet network that consists of one or more service instances through the use of CFM protocol messages. CFM partitions the service network into various administrative domains. Each administrative domain is mapped into a maintenance domain. A maintenance association end point (MEP) refers to the boundary of a domain. A MEP generates and responds to CFM protocol messages. You can configure multiple up (PE to PE) MEPs or down (PE to CE) MEPs for a single instance of a maintenance domain identifier and a maintenance association name to monitor services in a VPN.

For Layer 2 VPNs and EVPN networks, you can configure multiple up MEPs for a single combination of maintenance association ID and maintenance domain ID on a routing instance on the logical interfaces (IFLs), regardless of whether the logical interface is composed of physical interfaces on the same device or on different devices. The devices must be in enhanced IP network services mode.

In an EVPN network, the following CFM features are supported:

- Monitoring the connectivity between two provider edge (PE) routers in an active-active or active-standby multihomed configuration.

- Delay measurement and Synthetic Loss measurement. This feature is not supported when multiple MEPs are configured on multiple logical interfaces (IFLs) on the same physical interface (IFD).
- CFM monitoring between PE devices and customer edge (CE) devices. When the customer edge device is not a Juniper Networks device, you can enable CFM monitoring by using either the remote defect indication (RDI) bit or the Interface Status TLV. For more information, see [Understanding CFM Monitoring between CE and PE Devices](#)
- Starting with 18.3R1, Junos OS supports CFM configuration in attachment circuits (AC) on EVPN with ETREE services. The AC is a physical or virtual circuit that connects a CE device to a PE device. You can configure multiple Up MEPs on the MD or MA to monitor each AC between the CE and PE.
- Starting with 18.3R1, Junos OS supports maintenance intermediate points (MIPs) on Attachment Circuits on EVPN with ETREE and EVPN with ELAN services. When you configure MIP using the named bridge domain, all the interfaces will be enabled except for the EVPN core interface. For more information on MIPS, see [Configuring Maintenance Intermediate Points \(MIPs\)](#).
- Starting with 19.2R1, Junos OS supports MEPs and MIPs on ACs in an EVPN-VPWS network. CFM monitoring on EVPN-VPWS supports continuity check messages (CCM), delay measurements, synthetic loss measurement, loopback and link trace messages on single-active multihomed networks.
- Starting in Junos OS Release 20.2R1, Junos OS supports inline performance monitoring on an EVPN network. When inline performance monitoring is enabled, the router offloads performance monitoring packet processing from the CPU to other hardware, thereby decreasing the load on the CPU and allowing an increase in the number of performance monitoring session. Inline performance monitor only applies to delay measurements (DM) and synthetic loss measurements (SLM).

For more information on inline performance monitoring, see [Enabling Inline Mode Of Performance Monitoring To Achieve Maximum Scaling](#).

For more information on configuring delay measurement, see [Configuring Ethernet Frame Delay Measurement Sessions](#).

For more information on configuring synthetic loss measurement, see [Configuring Ethernet Synthetic Loss Measurements](#).

Limitations of CFM on layer 2 VPN and EVPNs

- In a circuit cross-connect (ccc) layer 2 VPN or local switch with MEPs and maintenance intermediate points (MIPs), the counter for link trace messages (LTMs) received on the MAC address of the up MEP does not get incremented when the MIP in the path is configured at the same level. The MIP traps the LTM packet, while the LTR message is sent. This leads to a discrepancy between the number of LTMs received and the number of LTRs sent.

- CFM up MEP on an EVPN network does not support the use of action profiles for interface down. In other words, you can configure an action profile, but no action is taken.
- CFM up MEP is supported on EVPN with ELAN and EVPN with ETREE services.
- CFM monitoring between leaf nodes on EVPN with ETREE services is not supported. CFM monitors MEP session from a leaf node to a root node and from a root node to another root node.
- CFM monitors the AC connectivity from between PE devices, learning about local adjacencies. In EVPN with E-TREE services, performance monitoring on local adjacencies is not supported.

For more information on CFM, see [IEEE 802.1ag OAM Connectivity Fault Management Overview](#).

Release History Table

Release	Description
19.2R1	Starting with 19.2R1, Junos OS supports MEPs and MIPs on ACs in an EVPN-VPWS network.
18.3R1	Starting with 18.3R1, Junos OS supports CFM configuration in attachment circuits (AC) on EVPN with ETREE services.
18.3R1	Starting with 18.3R1, Junos OS supports maintenance intermediate points (MIPs) on Attachment Circuits on EVPN with ETREE and EVPN with ELAN services.

RELATED DOCUMENTATION

IEEE 802.1ag OAM Connectivity Fault Management Overview
Creating a Maintenance Domain
Creating a Maintenance Association
Configuring a MEP to Generate and Respond to CFM Protocol Messages
show oam ethernet connectivity-fault-management interfaces

Configuring a MEP to Generate and Respond to CFM Protocol Messages

IN THIS SECTION

- [Configuring a Maintenance Association End Point \(MEP\) | 392](#)

● Configuring a remote Maintenance Association End Point (MEP) | 394

A maintenance association end point (MEP) refers to the boundary of a domain. A MEP generates and responds to connectivity fault management (CFM) protocol messages. You can configure multiple up MEPs for a single combination of maintenance association ID and maintenance domain ID for interfaces belonging to a particular VPLS service or a bridge domain. You can configure multiple down MEPs for a single instance of maintenance domain identifier and maintenance association name to monitor services provided by Virtual Private LAN service (VPLS), bridge domain, circuit cross-connect (CCC), or IPv4 domains.

For layer 2 VPNs routing instances (local switching) and EVPN routing instances, you can also configure multiple up MEPs for a single combination of maintenance association ID and maintenance domain ID on logical interfaces. The logical interface can be configured on different devices or on the same device. To support multiple up MEPs on two IFLs, enhanced IP network services must be configured for the chassis.

You can enable automatic discovery of a MEP. With automatic discovery a MEP is enabled to accept continuity check messages (CCMs) from all remote MEPs of the same maintenance association. If automatic discovery is not enabled, the remote MEPs must be configured. If the remote MEP is not configured, the CCMs from the remote MEP are treated as errors.

Continuity measurement is provided by an existing continuity check protocol. The continuity for every remote MEP is measured as the percentage of time that remote MEP was operationally up over the total administratively enabled time. Here, the operational uptime is the total time during which the CCM adjacency is active for a particular remote MEP and the administrative enabled time is the total time during which the local MEP is active. You can also restart the continuity measurement by clearing the currently measured operational uptime and the administrative enabled time.

Configuring a Maintenance Association End Point (MEP)

To configure a maintenance association end point:

1. Specify an ID for the MEP at the [edit protocols oam ethernet connectivity-fault-management maintenance-domain *domain-name* maintenance-association *ma-name*]. You can specify any value from 1 through 8191.

```
[edit protocols oam ethernet connectivity-fault-management maintenance-domain domain-name
maintenance-association ma-name]
user@host# set mep mep-id
```

2. Enable maintenance end point automatic discovery so the MEP can accept continuity check messages (CCMs) from all remote MEPs of the same maintenance association.

```
[edit protocols oam ethernet connectivity-fault-management maintenance-domain domain-name
maintenance-association ma-name mep mep-id]
user@host# set auto-discovery
```

3. Specify the direction in which the CCM packets are transmitted for the MEP. You can specify up or down. If you specify the direction as up, CCMs are transmitted out of every logical interface that is part of the same bridging or VPLS instance except for the interface configured on the MEP. If you specify the direction as down, CCMs are transmitted only out of the interface configured on the MEP.

NOTE: Ports in the Spanning Tree Protocol (STP) blocking state do not block CFM packets destined to a down MEP. Ports in an STP blocking state without the continuity check protocol configured do block CFM packets.

```
[edit protocols oam ethernet connectivity-fault-management maintenance-domain domain-name
maintenance-association ma-name mep mep-id]
user@host# set direction down
```

NOTE: Starting with Junos OS Release 12.3, for all interfaces configured on Modular Port Concentrators (MPCs) on MX Series 5G Universal Routing Platforms, you no longer need to configure the `no-control-word` statement for all Layer 2 VPNs and Layer 2 circuits over which you are running CFM MEPs. For all other interfaces on MX Series routers and on all other routers and switches, you must continue to configure the `no-control-word` statement at the `[edit routing-instances routing-instance-name protocols l2vpn]` or `[edit protocols l2circuit neighbor neighbor-id interface interface-name]` hierarchy level when you configure CFM MEPs. Otherwise, the CFM packets are not transmitted, and the `show oam ethernet connectivity-fault-management mep-database` command does not display any remote MEPs.

4. Specify the interface to which the MEP is attached. It can be a physical interface, logical interface, or trunk interface. On MX Series routers, the MEP can be attached to a specific VLAN of a trunk interface.

```
[edit protocols oam ethernet connectivity-fault-management maintenance-domain domain-name
maintenance-association ma-name mep mep-id]
user@host# set interface interface-name
```

5. Specify the IEEE 802.1 priority bits that are used by continuity check and link trace messages. You can specify a value from through 7 as the priority.

```
[edit protocols oam ethernet connectivity-fault-management maintenance-domain domain-name
maintenance-association ma-name mep mep-id]
user@host# set priority number
```

6. Specify the lowest priority defect that generates a fault alarm whenever CFM detects a defect. Possible values include: all -defects, err-xcon, mac-rem-err-xcon, no-defect, rem-err-xcon, and xcon.

```
[edit protocols oam ethernet connectivity-fault-management maintenance-domain domain-name
maintenance-association ma-name mep mep-id]
user@host# set lowest-priority-defect mac-rem-err-xcon
```

7. Specify the ID of the remote MEP at the [edit protocols oam ethernet connectivity-fault-management maintenance-domain *domain-name* maintenance-association *ma-name* mep *mep-id*]. You can specify any value from 1 through 8191.

```
[edit protocols oam ethernet connectivity-fault-management maintenance-domain domain-name
maintenance-association ma-name mep mep-id]
user@host# set remote-mep mep-id
```

SEE ALSO

[priority](#)

Configuring a remote Maintenance Association End Point (MEP)

To configure a remote maintenance association end point:

1. Configure the remote MEP by specifying the MEP ID at the [edit protocols oam ethernet connectivity-fault-management maintenance-domain *domain-name* maintenance-association *ma-name* mep *mep-id*]. You can specify any value from 1 through 8191.

```
[edit protocols oam ethernet connectivity-fault-management maintenance-domain domain-name
maintenance-association ma-name mep mep-id]
user@host# edit remote-mep mep-id
```

2. Specify the name of the action profile to be used for the remote MEP by including the action-profile *profile-name* statement at the [edit protocols oam ethernet connectivity-fault-management maintenance-domain

domain-name maintenance-association *ma-name* mep *mep-id* remote-mep *remote-mep-id*]. The profile must be defined at the [edit protocols oam ethernet connectivity-fault-management] hierarchy level.

```
[edit protocols oam ethernet connectivity-fault-management maintenance-domain domain-name
maintenance-association ma-namemep mep-id remote-mep remote-mep-id]
user@host# set action-profile profile-name
```

- 3. Configure the remote MEP to detect initial loss of connectivity. By default, the MEP does not generate loss-of-continuity (LOC) defect messages. When you configure the detect-loc statement, a loss-of-continuity (LOC) defect is detected if no continuity check message is received from the remote MEP within a period equal to 3.5 times the continuity check interval configured for the maintenance association. If a LOC defect is detected, a syslog error message is generated.

NOTE: When you configure connectivity-fault management (CFM) along with detect-loc, any action-profile configured to bring down the interface is executed if continuity check message is not received . However, the action-profile is not executed if you have not configured detect-loc and continuity check message is not received.

```
[edit protocols oam ethernet connectivity-fault-management maintenance-domain domain-name
maintenance-association ma-namemep mep-id remote-mep remote-mep-id]
user@host# set detect-loc
```

SEE ALSO

| [remote-mep](#)

Release History Table

Release	Description
12.3	Starting with Junos OS Release 12.3, for all interfaces configured on Modular Port Concentrators (MPCs) on MX Series 5G Universal Routing Platforms, you no longer need to configure the no-control-word statement for all Layer 2 VPNs and Layer 2 circuits over which you are running CFM MEPs.

RELATED DOCUMENTATION

| [action-profile](#)

auto-discovery

connectivity-fault-management

detect-loc

direction

lowest-priority-defect

2

PART

EVPN-VXLAN

[Overview | 398](#)

[Configuring EVPN-VXLAN Interfaces | 465](#)

[Configuring VLAN-Aware Bundle Services, VLAN-Based Services, and Virtual Switch Support | 497](#)

[Setting Up a Layer 3 VXLAN Gateway | 508](#)

[Configuring an EVPN-VXLAN Centrally-Routed Bridged Overlay | 580](#)

[Configuring an EVPN-VXLAN Edge-Routed Bridging Overlay | 617](#)

[IPv6 Underlay for VXLAN Overlays | 643](#)

[Multicast Features with EVPN-VXLAN | 668](#)

[Configuring the Tunneling of Q-in-Q Traffic | 809](#)

[Tunnel Traffic Inspection on SRX Series Devices | 834](#)

CHAPTER 11

Overview

IN THIS CHAPTER

- [Understanding EVPN with VXLAN Data Plane Encapsulation | 398](#)
- [EVPN-over-VXLAN Supported Functionality | 408](#)
- [Understanding VXLANs | 414](#)
- [VXLAN Constraints on QFX Series and EX Series Switches | 423](#)
- [EVPN Over VXLAN Encapsulation Configuration Overview for QFX Series and EX4600 Switches | 430](#)
- [Implementing EVPN-VXLAN for Data Centers | 432](#)
- [PIM NSR and Unified ISSU Support for VXLAN Overview | 436](#)
- [Routing IPv6 Data Traffic through an EVPN-VXLAN Network with an IPv4 Underlay | 438](#)
- [Understanding How to Configure VXLANs and Layer 3 Logical Interfaces to Interoperate | 453](#)
- [Dynamic Load Balancing in an EVPN-VXLAN Network | 455](#)
- [MAC-VRF Routing Instance Type Overview | 459](#)

Understanding EVPN with VXLAN Data Plane Encapsulation

IN THIS SECTION

- [Understanding EVPN | 399](#)
- [Understanding VXLAN | 402](#)
- [EVPN-VXLAN Integration Overview | 402](#)
- [Firewall Filtering and Policing Support for EVPN-VXLAN | 404](#)
- [Understanding Contrail Virtual Networks Use with EVPN-VXLAN | 405](#)
- [EVPN-VXLAN Support for VXLAN Underlays on MX Series Routers and the EX9200 Line of Switches | 406](#)
- [EVPN-VXLAN Support for VXLAN Underlays on QFX Series Switches | 406](#)
- [EVPN-VXLAN Packet Format | 407](#)

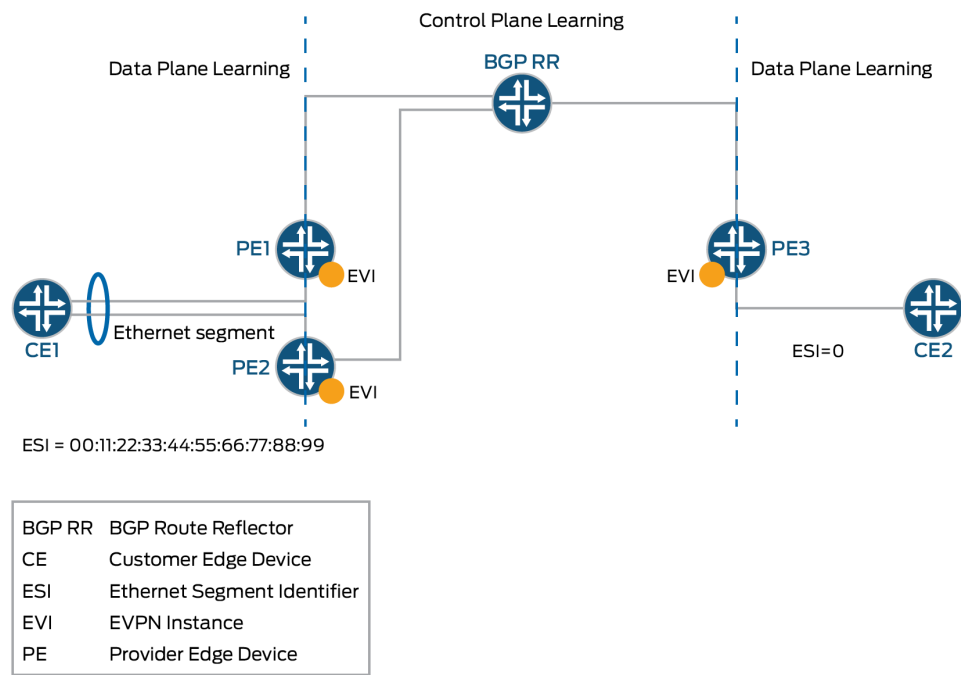
Ethernet VPNs (EVPNs) enable you to connect groups of dispersed customer sites using Layer 2 virtual bridges, and Virtual Extensible LANs (VXLANs) allow you to stretch Layer 2 connectivity over an intervening Layer 3 network, while providing network segmentation like a VLAN, but without the scaling limitation of traditional VLANs. EVPN with VXLAN encapsulation handles Layer 2 connectivity at the scale required by cloud server providers and replaces limiting protocols like Spanning Tree Protocol (STP), freeing up your Layer 3 network to use more robust routing protocols. EVPN with VXLAN data plane encapsulation can be used with and without Juniper Networks Contrail virtualization software—use Contrail with EVPN VXLAN data plane encapsulation when you have an environment that includes both virtual and bare-metal devices.

Understanding EVPN

Ethernet VPN (EVPN) is a standards-based technology that provides virtual multipoint bridged connectivity between different Layer 2 domains over an IP or IP/MPLS backbone network. Like other VPN technologies, such as IP VPN and virtual private LAN service (VPLS), EVPN instances are configured on provider edge (PE) routers to maintain logical service separation between customers. The PE routers connect to customer edge (CE) devices, which can be routers, switches, or hosts. The PE routers then exchange reachability information using Multiprotocol BGP (MP-BGP) and encapsulated traffic is forwarded between PE routers. Because elements of the architecture are common with other

VPN technologies, you can seamlessly introduce and integrate EVPN into existing service environments, as shown in [Figure 25 on page 400](#).

Figure 25: EVPN Overview



The EVPN is used as a Layer 2 overlay solution to provide Layer 2 connection over an IP underlay for the endpoints within a virtual network whenever Layer 2 connectivity is required by an end station such as bare-metal server (BMS). Otherwise, Layer 3 routing is used through VRF tables between Contrail vRouters and MX Series routers. EVPN technology offers multitenancy, flexible services that can be extended on demand, frequently using compute resources of different physical data centers for a single service (Layer 2 extension).

EVPN's MP-BGP control plane enables you to dynamically move live virtual machines from one data center to another, also known as virtual machine (VM) motion. After you move a VM to a destination server or hypervisor, it transmits a gratuitous ARP, which updates the Layer 2 forwarding table of the PE device at the destination data center. The PE device then transmits a MAC route update to all remote PE devices which in turn update their forwarding tables. An EVPN tracks the movement of the VM, which is also known as MAC mobility.

EVPN also has mechanisms that detect and stop MAC flapping, and prevent the looping of broadcast, unknown unicast, and multicast (BUM) traffic in an all-active multi-homed topology.

The EVPN technology, similar to Layer 3 MPLS VPN, includes the concept of routing MAC addresses using IP/MPLS core. EVPN provides the following benefits:

- Ability to have an active multihomed edge device
- Aliasing
- Fast convergence
- Load balancing across dual-active links
- MAC address mobility
- Multitenancy

In addition, EVPN uses these techniques:

- *Multihoming* provides redundancy in the event that an access link or one of the PE routing devices fails. In either case, traffic flows from the CE device towards the PE router, using the remaining active links. For traffic in the other direction, the remote PE router updates its forwarding table to send traffic to the remaining active PE routers connected to the multihomed Ethernet segment. EVPN provides a fast convergence mechanism, which reduces traffic restoration time so that the time it takes to make this adjustment is independent of the number of media access control (MAC) addresses learned by the PE router. All-active multihoming enables a CE device to connect to two or more PE routers such that traffic is forwarded using all of the links between the devices. This multihoming enables the CE device to load-balance traffic to multiple PE routers. More importantly, multihoming enables a remote PE router to load-balance traffic to the multihomed PE routers across the core network. This load balancing of traffic flows between data centers is known as *aliasing*, which causes different signals to become indistinguishable—they become aliases of one another. Aliasing is used with digital audio and digital images.
- *Split horizon* prevents the looping of broadcast, unknown unicast, and multicast (BUM) traffic in a network. The split horizon basic principle is simple: Information about the routing for a particular packet is never sent back in the direction from which it was received.
- *Local link bias* conserves bandwidth by using local links to forward unicast traffic exiting a Virtual Chassis or Virtual Chassis Fabric (VCF) that has a link aggregation group (LAG) bundle composed of member links on different member switches in the same Virtual Chassis or VCF. A local link is a member link in the LAG bundle that is on the member switch that received the traffic.
- EVPN with VXLAN encapsulation is used for Layer 2 connectivity between virtual machines and a top-of-rack (TOR) switch, for example, a QFX5100 switch, within a Layer 2 domain.

You can use Contrail to provision an MX Series router as a Layer 2 or Layer 3 VXLAN gateway. MX Series routers implement the NETCONF XML management protocol, which is an XML-based protocol that client applications use to request and change configuration information on routing, switching, and security devices. The NETCONF XML management protocol uses an XML based data encoding for the configuration data and remote procedure calls. The NETCONF protocol defines basic operations that are equivalent to configuration mode commands in the command-line interface (CLI). Applications use the

protocol operations to display, edit, and commit configuration statements similarly to how administrators use CLI configuration mode commands to perform those same operations.

Understanding VXLAN

Virtual extensible LANs (VXLANs) introduced an overlay scheme that expands the Layer 2 network address space from 4K to 16 million, largely solving the scaling issues seen in VLAN-based environments.

Network overlays are created by encapsulating traffic and tunneling the traffic over a physical network. You can use a number of tunneling protocols in the data center to create network overlays—the most common protocol is VXLAN. VXLAN tunneling protocol encapsulates Layer 2 Ethernet frames in Layer 3 UDP packets. This encapsulation enables you to create virtual Layer 2 subnets or segments that can span physical Layer 3 networks.

In a VXLAN overlay network, a VXLAN network identifier (VNI) uniquely identifies each Layer 2 subnet or segment. A VNI segments traffic the same way that an IEEE 802.1Q VLAN ID segments traffic. As is the case with VLAN, virtual machines on the same VNI can communicate directly with each other, whereas virtual machines on different VNIs need a router to communicate with each other.

The entity that performs the encapsulation and de-encapsulation is called a VXLAN tunnel endpoint (VTEP). In the physical network, a Juniper Networks device that functions as a Layer 2 or Layer 3 VXLAN gateway can encapsulate and de-encapsulate data packets. This type of VTEP is known as a hardware VTEP. In the virtual network, VTEPs can reside in hypervisor hosts, such as kernel-based virtual machine (KVM) hosts. This type of VTEP is known as a software VTEP.

Each VTEP has two interfaces.

- One interface is a switching interface that faces the virtual machines in the host and provides communication between VMs on the local LAN segment.
- The other is an IP interface that faces the Layer 3 network.

Each VTEP has a unique IP address that is used for routing the UDP packets between VTEPs. For example, when VTEP1 receives an Ethernet frame from VM1 addressed to VM3, it uses the VNI and the destination MAC to look up in its forwarding table which VTEP sends the packet to. It then adds a VXLAN header that contains the VNI to the Ethernet frame and encapsulates the frame in a Layer 3 UDP packet and routes the packet to VTEP2 over the Layer 3 network. VTEP2 de-encapsulates the original Ethernet frame and forwards it to VM3. VM1 and VM3 cannot detect the VXLAN tunnel and the Layer 3 network between them.

EVPN-VXLAN Integration Overview

VXLAN defines a tunneling scheme to overlay Layer 2 networks on top of Layer 3 networks. This tunneling scheme allows for optimal forwarding of Ethernet frames with support for multipathing of

unicast and multicast traffic with the use of UDP/IP encapsulation for tunneling, and is mainly used for the intra-data center site connectivity.

A unique characteristic of EVPN is that MAC address learning between PE routers occurs in the control plane. The local PE router detects a new MAC address from a CE device and then, using MP-BGP, advertises the address to all the remote PE routers. This method differs from existing Layer 2 VPN solutions such as VPLS, which learn by flooding unknown unicast in the data plane. This control plane MAC learning method is the key enabler of the many useful features that EVPN provides.

Because MAC learning is handled in the control plane, EVPN has the flexibility to support different data plane encapsulation technologies between PE routers. This flexibility is important because not every backbone network might be running MPLS, especially in enterprise networks.

EVPN addresses many of the challenges faced by network operators building data centers to offer cloud and virtualization services. The main application of EVPN is Data Center Interconnect (DCI), which refers to the ability to extend Layer 2 connectivity between different data centers that are deployed to improve the performance of delivering application traffic to end users and for disaster recovery.

Although various DCI technologies are available, EVPN has an advantage over the other MPLS technologies because of its unique features, such as active/active redundancy, aliasing, and mass MAC withdrawal. As a result, to provide a solution for DCI, VXLAN is integrated with EVPN.

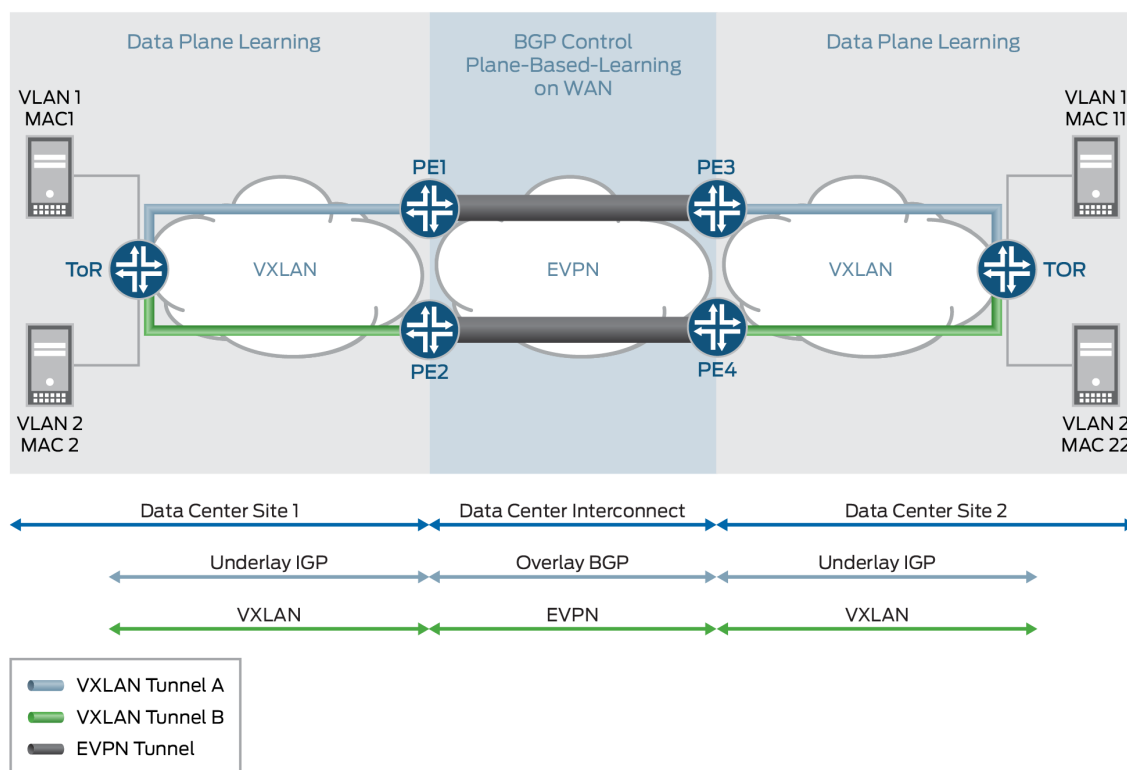
As shown in [Figure 26 on page 404](#), each VXLAN, which is connected to the MPLS or IP core, runs an independent instance of the interior gateway protocol (IGP) control plane. Each PE router participates in the IGP control plane instance of its VXLAN. Each customer is a data center, so each has its own virtual router for VXLAN underlay.

Each PE node can terminate the VXLAN data plane encapsulation where the VXLAN network identifier (VNI) is mapped to a bridge domain or VLAN. The PE router performs data plane learning on the traffic received from the VXLAN.

Each PE node implements EVPN to distribute the client MAC addresses learned over the VXLAN tunnel into BGP. Each PE node encapsulates the VXLAN or Ethernet frames with MPLS when sending the

packets over the MPLS core and with the VXLAN tunnel header when sending the packets over the VXLAN network.

Figure 26: EVPN-VXLAN Integration Overview



Firewall Filtering and Policing Support for EVPN-VXLAN

For each firewall filter that you apply to a VXLAN, specify `family ethernet-switching` to filter Layer 2 (Ethernet) packets, or `family inet` to filter on IRB interfaces. The IRB interface acts as a Layer 3 routing interface to connect the VXLANs in one-layer or two-layer IP fabric topologies. The following limitations apply:

- Filtering and policing are not supported for VXLAN transit traffic.
- Firewall filtering on VNI at the egress VTEP device is not supported.
- Policing on VNI at the egress VTEP device is not supported.
- Match conditions against VXLAN header fields are not supported.

NOTE: EVPN-VXLAN firewall filters are configured on the interface after the VXLAN header is stripped by the VXLAN tunnel endpoint (VTEP).

For more information on configuring firewall filters, match conditions, and actions, see:

- [Configuring Firewall Filters](#)
- [Firewall Filter Match Conditions and Actions \(QFX and EX Series Switches\)](#)
- [Firewall Filter Match Conditions and Actions \(QFX10000 Switches\)](#)
- [Firewall Filters for EX Series Switches Overview](#)

Understanding Contrail Virtual Networks Use with EVPN-VXLAN

Juniper Networks Contrail virtualization software is a software-defined networking (SDN) solution that automates and orchestrates the creation of highly scalable virtual networks. These virtual networks enable you to harness the power of the cloud—for new services, increased business agility, and revenue growth. MX Series routers can use EVPN-VXLAN to provide both Layer 2 and Layer 3 connectivity for end stations within a Contrail virtual network (VN).

The Contrail software for virtual networks provides both Layer 2 and Layer 3 connectivity. With Contrail, Layer 3 routing is preferred over Layer 2 bridging whenever possible. Layer 3 routing is used through virtual routing and forwarding (VRF) tables between Contrail vRouters and physical MX Series routers. MX Series routers provide Layer 3 gateway functionality between virtual networks.

Contrail enables you to use EVPN-VXLAN when your network includes both virtual and bare-metal devices.

Two types of encapsulation methods are used in virtual networks.

- MPLS-over-GRE (generic routing encapsulation) is used for Layer 3 routing between Contrail and MX Series routers.
- EVPN-VXLAN is used for Layer 2 connectivity between virtual machines and top-of-rack (TOR) switches, for example, QFX5100 switches, within a Layer 2 domain. For Layer 2 connectivity, traffic load balancing in the core is achieved by using the multihoming all-active feature provided by EVPN. Starting with Junos OS Release 17.3R1, EX9200 switches also support EVPN-VXLAN with Contrail.

NOTE: MPLS core is not supported on switches—only MX Series routers support this feature.

You cannot simultaneously mix EVPN-VXLAN with Open vSwitch Database (OVSDB)-VXLAN on QFX Series switches. After a switch is set to OVSDB-managed, the controller treats all ports as managed by OVSDB.

EVPN-VXLAN Support for VXLAN Underlays on MX Series Routers and the EX9200 Line of Switches

MX Series routers and the EX92xx line of switches support Virtual Extensible LAN (VXLAN) gateways. Each VXLAN gateway supports the following functionalities:

- Switching functionality with traditional Layer 2 networks and VPLS networks
- Inter-VXLAN routing and VXLAN-only bridging domain with IRB
- Virtual switches
- VXLAN with VRF functionality
- Configurable load balancing
- Statistics for remote VTEP

Starting in Junos OS Release 17.3R1, EVPN-VXLAN support on MX Series routers is extended to VXLAN gateway implementation using an IPv6 underlay. We support EVPN Type 1, Type 2, Type 3, and Type 4 routes with an IPv6 underlay on MX Series routers.

We support the following service types with the IPv6 underlay support:

- VLAN-based service
- VLAN-bundle service
- Port-based service
- VLAN-aware service

Both IPv4 and IPv6 EVPN-VXLAN underlays support EVPN Type 2 MAC addresses with IP address advertisement and proxy MAC addresses with IP address advertisement.

EVPN-VXLAN Support for VXLAN Underlays on QFX Series Switches

QFX Series switches support VXLAN gateways in an EVPN-VXLAN network. All devices that support EVPN-VXLAN can use an IPv4 underlay for the VXLAN overlay.

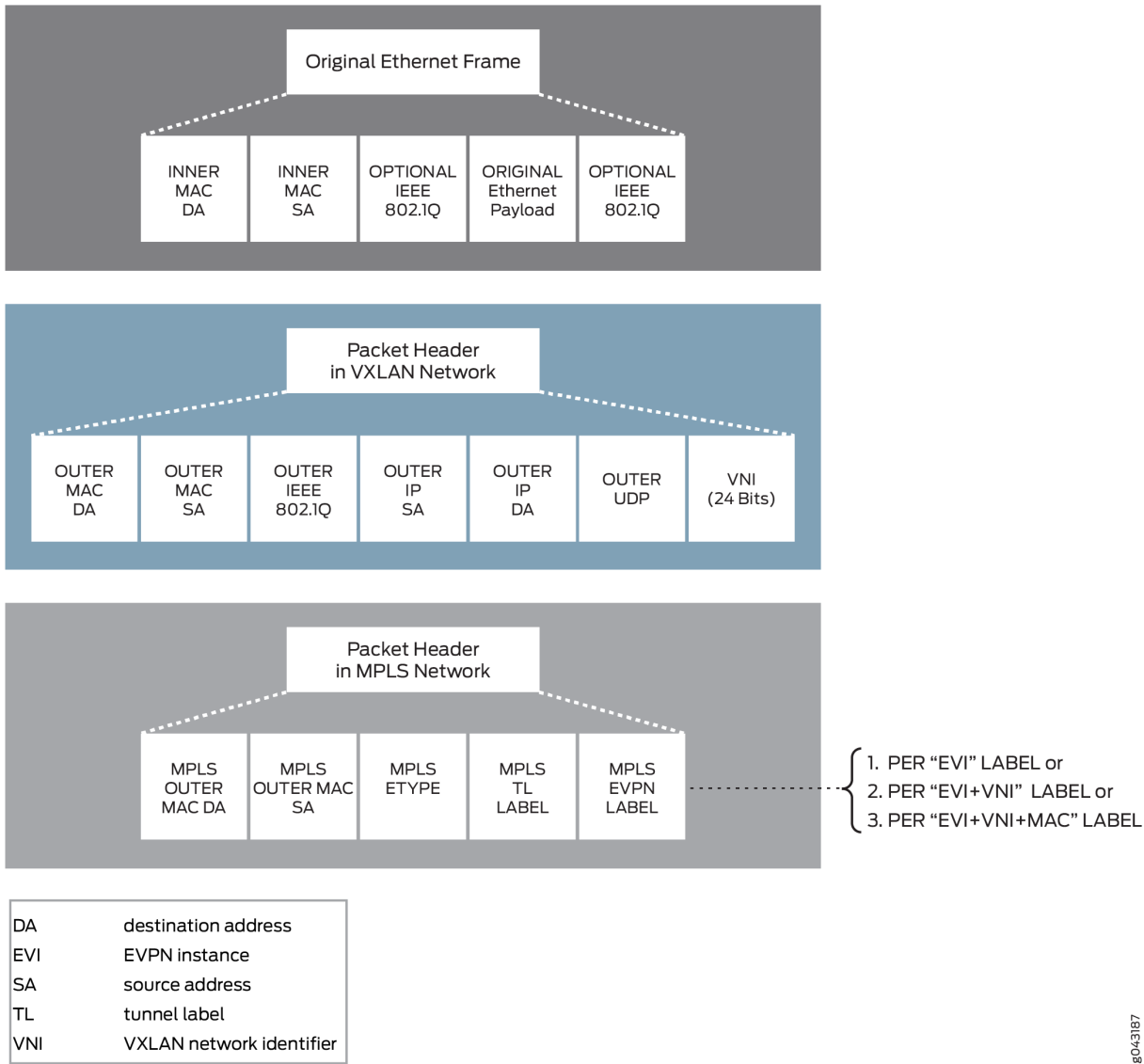
Starting in Junos OS Releases 21.2R2 and 21.4R1, the QFX10000 line of switches and most models of QFX5120 switches support configuring an IPv6 underlay for the VXLAN overlay in an EVPN-VXLAN network. You can only configure an IPv6 underlay using MAC-VRF EVPN instances. With an IPv6

underlay, the outer IP header in the VXLAN packet is an IPv6 header, and you configure the VTEP source address as an IPv6 address. See ["EVPN-VXLAN with an IPv6 Underlay" on page 643](#) for more about IPv6 underlay support and how to configure a VXLAN gateway device to use an IPv6 underlay.

EVPN-VXLAN Packet Format

The EVPN-VXLAN packet format is shown in [Figure 27 on page 407](#).

Figure 27: EVPN-VXLAN Packet Format



Release History Table

Release	Description
21.4R1	Starting in Junos OS Releases 21.2R2 and 21.4R1, the QFX10000 line of switches and most QFX5120 switches support configuring an IPv6 underlay for the VXLAN overlay in an EVPN-VXLAN network.
17.3R1	Starting with Junos OS Release 17.3R1, EX9200 switches also support EVPN-VXLAN with Contrail.
17.3R1	Starting in Junos OS Release 17.3R1, EVPN-VXLAN support on MX Series is extended to VXLAN gateway implementation using an IPv6 underlay.

RELATED DOCUMENTATION

[EVPN Multihoming Overview | 96](#)

[Understanding EVPN Pure Type 5 Routes | 22](#)

[Understanding VXLANs | 414](#)

[EVPN-over-VXLAN Supported Functionality | 408](#)

[EVPN Over VXLAN Encapsulation Configuration Overview for QFX Series and EX4600 Switches | 430](#)

EVPN-over-VXLAN Supported Functionality

IN THIS SECTION

- [VXLAN Encapsulation | 409](#)
- [EVPN BGP Routes and Attributes | 409](#)
- [EVPN Multihoming Procedure | 410](#)
- [Next-Hop Forwarding | 411](#)
- [Control Plane MAC Learning Method | 411](#)
- [Contrail vRouters and the L3-VRF Table | 412](#)
- [Designated Forwarder Election | 413](#)

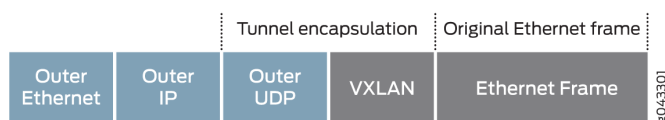
The following functionality is supported for EVPN-over-VXLAN data plane encapsulation:

VXLAN Encapsulation

EVPN supports the VXLAN data plane encapsulation type within the same EVPN instance. To support the VXLAN encapsulation type, all EVPN PE devices specify the tunnel type as VXLAN in the BGP encapsulation extended community. The ingress PE determines which encapsulation types to use based on its own encapsulation capability and the one its EVPN remote PE device advertises.

When EVPN is used as overlay solution for the underlay IP network with VXLAN encapsulation, the data packet is encapsulated with a VXLAN header at the ingress PE device, and the data packet is de-encapsulated from the VXLAN header at the egress PE device. [Figure 28 on page 409](#) shows the packet format when a packet is forwarded in the core through the underlay IP network with VXLAN encapsulation:

Figure 28: Underlay IP Network with VXLAN Encapsulation



If the underlay network uses the IPv4 protocol and IPv4 addressing, the outer IP header in the VXLAN packet is an IPv4 header. All platforms that support EVPN-VXLAN work with an IPv4 underlay network.

On some platforms, you can configure the underlay to use the IPv6 protocol and IPv6 addressing. In that case, the outer IP header in the VXLAN packet is an IPv6 header, and you configure the VTEP source address as an IPv6 address. See ["EVPN-VXLAN with an IPv6 Underlay" on page 643](#) for details on using an IPv6 underlay.

EVPN BGP Routes and Attributes

EVPN BGP routes and attributes support EVPN-over-VXLAN data plane encapsulation and are affected as follows:

- By attaching the BGP encapsulation extended community attribute with tunnel encapsulation type VXLAN in the EVPN MAC routes, the egress PE device advertises the EVPN inclusive multicast route and EVPN per EVI route autodiscovery.
- The Ethernet tag fields for all EVPN BGP routes are set to zero to support VXLAN network identifier (VNI)-based mode only.
- The VNI, also referred to as the VNI field in the EVPN overlay, is placed in the MPLS fields for the EVPN MAC route, the EVPN inclusive multicast route, and per EVPN instance autodiscovery route.

- The Ethernet segment identifier label field is set to zero for the EVPN per Ethernet segment autodiscovery route because no Ethernet segment identifier label exists for the supported VXLAN encapsulation type.
- All corresponding EVPN routes from the remote PE device are resolved over the inet.0 or :vxlan.inet.0 table instead of the inet.3 table for IPv4. When you use an IPv6 underlay for EVPN-VXLAN tunneling, the same is true for the inet6.0 and :vxlan.inet6.0 tables.

EVPN Multihoming Procedure

You can multihome a bare-metal server (BMS) to a pair of Juniper Networks top-of-rack (TOR) switches, for example, QFX5100 switches, through a Layer 2 link aggregation group (LAG) interface. Because of the LAG interface, only one copy of BUM traffic is forwarded from a BMS to the core router. To prevent a Layer 2 loop and flooding of BUM traffic back to the same Ethernet segment to which the multihomed BMS is attached, the BUM traffic applies the multihomed active-active split-horizon filtering rule. To fully support the EVPN multihomed active-active mode feature, the Juniper Networks TOR switch also advertises the EVPN aliasing route to other EVPN PE devices.

The lack of MPLS label support for EVPN over IP with VXLAN data plane encapsulation causes impacts to the following EVPN multihomed active-active mode functions:

NOTE: EVPN over VxLAN does not support active-standby mode of operation. You can only configure the active-active mode by using all-active option for the ESI interface configuration. Single-homing Ethernet segments with EVPN using VxLAN encapsulation using single-active is not supported.

Local Bias and Split-Horizon Filtering Rule

Because of the missing MPLS label, the split-horizon filtering rule for the multihomed Ethernet segment is modified and based on the IP address of the EVPN PE device instead of the MPLS Ethernet segment label. For traffic coming from the access interface, any traffic forwarding to the multihomed Ethernet segment is based on the local bias for EVPN with VXLAN data plane encapsulation. Each EVPN PE device tracks the IP address of its peer multihomed EVPN PE device for which it shares the same Ethernet segment. This tracking provides the source VXLAN virtual tunnel endpoint (VTEP) IP address [outer Session Initiation Protocol (SIP)] for each VXLAN packet received from the other EVPN PE device. The split-horizon filtering rule is enforced on both ingress and egress PE devices for multi-destination traffic:

- Ingress PE—Responsible for forwarding multi-destination packets coming from any of its directly-attached access interfaces to its remaining associated multihomed Ethernet segments, regardless of the subject ingress PE device's designated forwarder (DF) election status.

- Egress PE—Allows no forwarding of any multi-destination packets to the same multihomed Ethernet segment to which an egress PE shares with its ingress PE device, regardless of the subject egress PE device's DF election status.

Aliasing

When you connect a BMS to a pair of Juniper Networks TOR switches through a LAG interface bundle, only one of the switches can learn the local MAC address. To support the load balancing of known unicast traffic coming from the VXLAN to the BMS between the switches, the EVPN PE device on the switch must advertise EVPN per EVPN instance autodiscovery route. This advertisement signals to the remote EVPN PE devices that the MAC learned from the multihomed Ethernet segment from its peer multihomed PE device is also reachable. For VXLAN encapsulation, there is no change to the EVPN procedure when providing the EVPN aliasing function.

Next-Hop Forwarding

The Layer 2 address learning daemon (l2ald) creates the VXLAN encapsulation composite next-hop at ingress, and the VXLAN de-encapsulation composite next-hop at the egress. The VXLAN encapsulation composite next-hop forwards Layer 2 unicast traffic to the remote PE device with the VXLAN encapsulation. If multiple paths are available to reach a remote MAC (as with the multihomed EVPN active-active case), the routing protocol daemon (rpd) informs the l2ald of all the remote VTEP IP addresses associated with the remote MAC. The l2ald is responsible for building an equal-cost multipath (ECMP) next hop for the remote MAC. The VXLAN de-encapsulation composite next-hop de-encapsulates the VXLAN tunnel header at the egress and then forwards the traffic to the BMS.

For known unicast traffic:

- At ingress, the rpd does not need to add a label route in the mpls.0 table.
- At egress, the rpd does not need to add a label route to point to the table next-hop in the mpls.0 table.

Control Plane MAC Learning Method

A unique characteristic of EVPN is that MAC address learning between PE devices occurs in the control plane. The local PE router detects a new MAC address from a CE device and then, using MP-BGP, advertises the address to all the remote PE devices. This method differs from existing Layer 2 VPN solutions such as VPLS, which learn by flooding unknown unicast in the data plane. This control plane MAC learning method is a crucial component of the features and benefits that EVPN provides. Because MAC learning is handled in the control plane, EVPN has the flexibility to support different data plane encapsulation technologies between PE devices. This flexibility is important because not every backbone network may be running MPLS, especially in enterprise networks.

For control plane remote MAC learning, because the l2ald creates the VXLAN encapsulation composite next-hop and VXLAN de-encapsulation composite next-hop, the rpd no longer creates an indirect next-hop used in the MAC forwarding table. The rpd relies on the existing mechanism to inform the l2ald of the remote MAC addresses learned from the control plane. Instead of informing the l2ald about the VLAN ID and the indirect next-hop index used by the remote MAC in the MAC FIB table, the rpd informs the l2ald about the remote VTEP IP address and VXLAN network identifier. The control plane remote MAC learned route points to VXLAN encapsulation composite next-hop, or an ECMP next-hop created by the l2ald in the MAC forwarding table.

For multihomed EVPN active-active, a pair of remote VTEP IP addresses are associated with the remote MAC address. The remote VTEP IP addresses are obtained from the MAC route received either from the remote PE device, or from the aliasing route from the remote PE device. When a remote PE device withdraws a MAC route or the aliasing route for the Ethernet segment from where the MAC learned, the rpd alerts the l2ald about the change to the pair of remote VTEP IP addresses accordingly. As a result, the l2ald updates the uni-list next-hop built for this MAC. When both remote MAC routes and its associated aliasing route are withdrawn or become unresolved, the rpd informs the l2ald about the deletion of this remote MAC and the l2ald withdraws this MAC from the MAC forwarding table.

Contrail vRouters and the L3-VRF Table

The Contrail virtualization software creates virtual networks (VNs) associated with routes in a Layer 3 virtual routing and forwarding (L3-VRF) table.

The following are the associated routes:

Subnet Routes for an IRB Interface

For each pair of MAC-(virtual routing and forwarding) VRF and L3-VRF tables created for a virtual network on an MX Series router, a corresponding IRB interface is associated with the MAC-VRF and L3-VRF table pair. The L3-VRF table on the MX Series router has the subnet route from the IRB interface associated with its local virtual networks, as well as, all subnet routes of the IRB interfaces associated with other virtual networks within a data center provided by the Junos OS routing leaking feature. The MX Series routers advertise these subnet routes through MP-BGP to the Contrail control nodes. As a result, the L3-VRF table in the Contrail vRouter contains the same set of subnet routes for the IRB interfaces in its IP FIB, and the subnet routes point their next hops to the MX Series routers.

Virtual Machine Host Routes

The Contrail vRouters support proxy ARP and advertise the IP address with the EVPN MAC route for its virtual machine (VM). For both Contrail vRouters and MX Series routers, the L3-VRF table for a virtual network contains all of the VM host routes for those VMs that reside in the same virtual network, as well as, routes in all other virtual networks. intra- virtual network and inter- virtual network traffic between the VMs is forwarded directly at Layer 3 between the Contrail vRouters.

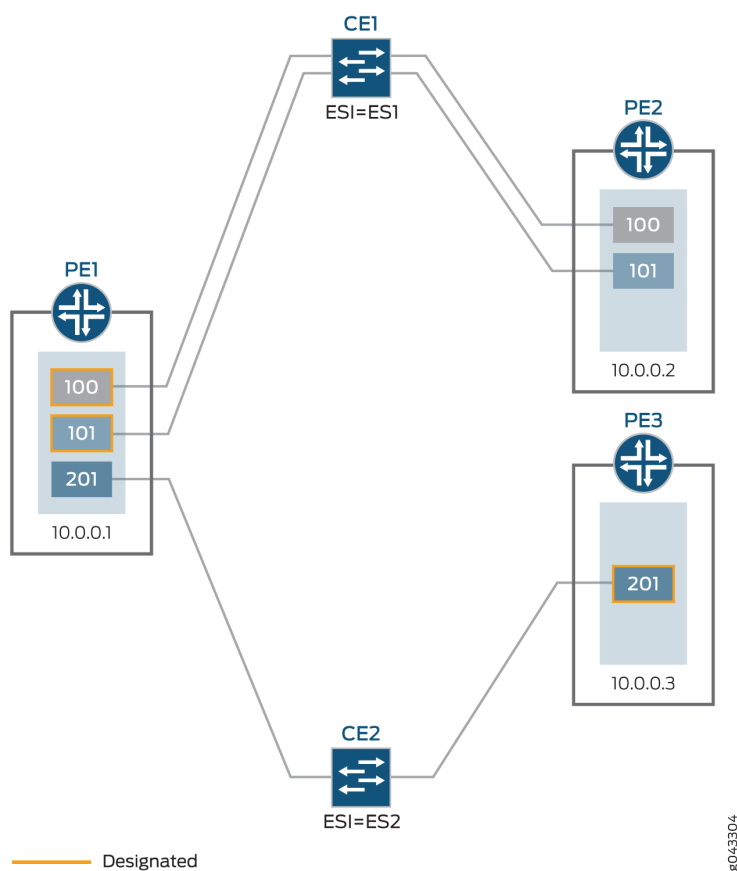
Bare-Metal Server Host Routes

A Juniper Networks TOR switch, for example, a QFX5100 switch, does not advertise the IP address with the EVPN MAC route for the bare-metal server (BMS) to which it is attached. As a result of the EVPN MAC route advertisement from the switch, no BMS host route is installed in the L3-VRF table on Contrail vRouters and MX Series routers. However, ARP routes for the BMSs are installed in the kernel on the MX Series router if the MX Series router receives ARP responses from the BMSs.

Designated Forwarder Election

To provide better load balance and more flexible topology, the election of the designated forwarder is determined by selecting the minimum VLAN ID or VXLAN network ID for each Ethernet segment instead of selecting based on the EVPN instance. [Figure 29 on page 413](#) shows a sample designated forwarder election topology.

Figure 29: Designated Forwarder Election Topology



CE device (CE1) has an Ethernet segment identifier value configured equal to ES1 and is connected to both PE1 and PE2 devices, with configured VLANs 100 and 101. Router PE1 is elected as the designated forwarder for the two VLANs.

CE device (CE2) has an Ethernet segment identifier value configured equal to ES2 and is connected to both PE1 and PE3 devices, with configured VLAN 201. Router PE3 is elected as the designated forwarder for this VLAN.

The Ethernet tag ID can be either of the following:

- VLAN ID (for EVPN-MPLS)
- VXLAN network ID (for EVPN-VXLAN)

RELATED DOCUMENTATION

[Understanding EVPN with VXLAN Data Plane Encapsulation | 398](#)

[EVPN Over VXLAN Encapsulation Configuration Overview for QFX Series and EX4600 Switches | 430](#)

[Using a Default Layer 3 Gateway to Route Traffic in an EVPN-VXLAN Overlay Network | 508](#)

Understanding VXLANs

IN THIS SECTION

- [VXLAN Benefits | 415](#)
- [How Does VXLAN Work? | 416](#)
- [VXLAN Implementation Methods | 417](#)
- [Using QFX5100, QFX5110, QFX5120, QFX5200, QFX5210, EX4300-48MP, and EX4600 Switches with VXLANs | 417](#)
- [Changing the UDP Port on QFX5100, QFX5110, QFX5200, QFX5210, and EX4600 Switches | 418](#)
- [Controlling Transit Multicast Traffic on QFX5100, QFX5110, QFX5200, QFX5210, and EX4600 Switches | 419](#)
- [Using an MX Series Router, EX9200 Switch, or QFX10000 Switch as a VTEP | 420](#)
- [Manual VXLANs Require PIM | 420](#)
- [Load Balancing VXLAN Traffic | 421](#)

- [VLAN IDs for VXLANs | 421](#)
- [Enabling QFX5120 Switches to Tunnel Traffic on Core-Facing Layer 3 Tagged and IRB Interfaces | 421](#)
- [Using ping and traceroute with a VXLAN | 422](#)
- [Supported VXLAN Standards | 422](#)

Virtual Extensible LAN protocol (VXLAN) technology allows networks to support more VLANs. According to the IEEE 802.1Q standard, traditional VLAN identifiers are 12 bits long—this naming limits networks to 4094 VLANs. The VXLAN protocol overcomes this limitation by using a longer logical network identifier that allows more VLANs and, therefore, more logical network isolation for large networks such as clouds that typically include many virtual machines.

VXLAN Benefits

VXLAN technology allows you to segment your networks (as VLANs do), but it provides benefits that VLANs cannot. Here are the most important benefits of using VXLANs:

- You can theoretically create as many as 16 million VXLANs in an administrative domain (as opposed to 4094 VLANs on a Juniper Networks device).
 - MX Series routers and EX9200 switches support as many as 32,000 VXLANs, 32,000 multicast groups, and 8000 virtual tunnel endpoints (VTEPs). This means that VXLANs based on MX Series routers provide network segmentation at the scale required by cloud builders to support very large numbers of tenants.
 - QFX10000 Series switches support 4000 VXLANs and 2000 remote VTEPs.
 - QFX5100, QFX5110, QFX5200, QFX5210, and EX4600 switches support 4000 VXLANs, 4000 multicast groups, and 2000 remote VTEPs.
 - EX4300-48MP switches support 4000 VXLANs.
- You can enable migration of virtual machines between servers that exist in separate Layer 2 domains by tunneling the traffic over Layer 3 networks. This functionality allows you to dynamically allocate resources within or between data centers without being constrained by Layer 2 boundaries or being forced to create large or geographically stretched Layer 2 domains.

Using VXLANs to create smaller Layer 2 domains that are connected over a Layer 3 network means that you do not need to use Spanning Tree Protocol (STP) to converge the topology but can use more robust routing protocols in the Layer 3 network instead. In the absence of STP, none of your links are blocked, which means you can get full value from all the ports that you purchase. Using routing protocols to connect your Layer 2 domains also allows you to load-balance the traffic to ensure that you get the best

use of your available bandwidth. Given the amount of east-west traffic that often flows within or between data centers, maximizing your network performance for that traffic is very important.

The video *Why Use an Overlay Network in a Data Center?* presents a brief overview of the advantages of using VXLANs.



Video: [Why Use an Overlay Network in a Data Center?](#)

How Does VXLAN Work?

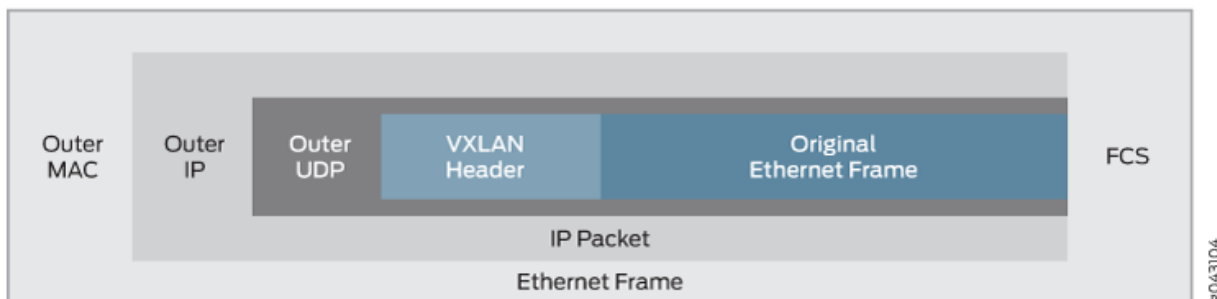
VXLAN is often described as an overlay technology because it allows you to stretch Layer 2 connections over an intervening Layer 3 network by encapsulating (tunneling) Ethernet frames in a VXLAN packet that includes IP addresses. Devices that support VXLANs are called *virtual tunnel endpoints (VTEPs)*—they can be end hosts or network switches or routers. VTEPs encapsulate VXLAN traffic and de-encapsulate that traffic when it leaves the VXLAN tunnel. To encapsulate an Ethernet frame, VTEPs add a number of fields, including the following fields:

- Outer media access control (MAC) destination address (MAC address of the tunnel endpoint VTEP)
- Outer MAC source address (MAC address of the tunnel source VTEP)
- Outer IP destination address (IP address of the tunnel endpoint VTEP)
- Outer IP source address (IP address of the tunnel source VTEP)
- Outer UDP header
- A VXLAN header that includes a 24-bit field—called the *VXLAN network identifier (VNI)*—that is used to uniquely identify the VXLAN. The VNI is similar to a VLAN ID, but having 24 bits allows you to create many more VXLANs than VLANs.

NOTE: Because VXLAN adds 50 to 54 bytes of additional header information to the original Ethernet frame, you might want to increase the MTU of the underlying network. In this case, configure the MTU of the physical interfaces that participate in the VXLAN network, not the MTU of the logical VTEP source interface, which is ignored.

Figure 30 on page 417 shows the VXLAN packet format.

Figure 30: VXLAN Packet Format



VXLAN Implementation Methods

Junos OS supports implementing VXLANs in the following environments:

- **Manual VXLAN**—In this environment, a Juniper Networks device acts as a transit device for downstream devices acting as VTEPs, or a gateway that provides connectivity for downstream servers that host virtual machines (VMs), which communicate over a Layer 3 network. In this environment, software-defined networking (SDN) controllers are not deployed.

NOTE: QFX10000 switches do not support manual VXLANs.

- **OVSDB-VXLAN**—In this environment, SDN controllers use the Open vSwitch Database (OVSDB) management protocol to provide a means through which controllers (such as a VMware NSX or Juniper Networks Contrail controller) and Juniper Networks devices that support OVSDB can communicate.
- **EVPN-VXLAN**—In this environment, Ethernet VPN (EVPN) is a control plane technology that enables hosts (physical servers and VMs) to be placed anywhere in a network and remain connected to the same logical Layer 2 overlay network, and VXLAN creates the data plane for the Layer 2 overlay network.

Using QFX5100, QFX5110, QFX5120, QFX5200, QFX5210, EX4300-48MP, and EX4600 Switches with VXLANs

You can configure the switches to perform all of the following roles:

- (All switches except EX4300-48MP) In an environment without an SDN controller, act as a transit Layer 3 switch for downstream hosts acting as VTEPs. In this configuration, you do not need to configure any VXLAN functionality on the switch. You do need to configure IGMP and PIM so that the switch can form the multicast trees for the VXLAN multicast groups. (See "[Manual VXLANs Require PIM](#)" on page 420 for more information.)
- (All switches except EX4300-48MP) In an environment with or without an SDN controller, act as a Layer 2 gateway between virtualized and nonvirtualized networks in the same data center or between data centers. For example, you can use the switch to connect a network that uses VXLANs to one that uses VLANs.
- (EX4300-48MP switches) Act as a Layer 2 gateway between virtualized and nonvirtualized networks in a campus network. For example, you can use the switch to connect a network that uses VXLANs to one that uses VLANs.
- (All switches except EX4300-48MP) Act as a Layer 2 gateway between virtualized networks in the same or different data centers and allow virtual machines to move (VMotion) between those networks and data centers. For example, if you want to allow VMotion between devices in two different networks, you can create the same VLAN in both networks and put both devices on that VLAN. The switches connected to these devices, acting as VTEPs, can map that VLAN to the same VXLAN, and the VXLAN traffic can then be routed between the two networks.
- (QFX5110 and QFX5120 switches with EVPN-VXLAN) Act as a Layer 3 gateway to route traffic between different VXLANs in the same data center.
- (QFX5110 and QFX5120 switches with EVPN-VXLAN) Act as a Layer 3 gateway to route traffic between different VXLANs in different data centers over a WAN or the Internet using standard routing protocols or virtual private LAN service (VPLS) tunnels.

NOTE: If you want a QFX5110 or QFX5120 switch to be a Layer 3 VXLAN gateway in an EVPN-VXLAN environment, you must configure integrated routing and bridging (IRB) interfaces to connect the VXLANs, just as you do if you want to route traffic between VLANs.

Because the additional headers add 50 to 54 bytes, you might need to increase the MTU on a VTEP to accommodate larger packets. For example, if the switch is using the default MTU value of 1514 bytes and you want to forward 1500-byte packets over the VXLAN, you need to increase the MTU to allow for the increased packet size caused by the additional headers.

Changing the UDP Port on QFX5100, QFX5110, QFX5200, QFX5210, and EX4600 Switches

Starting with Junos OS Release 14.1X53-D25 on QFX5100 switches, Junos OS Release 15.1X53-D210 on QFX5110 and QFX5200 switches, Junos OS Release 18.1R1 on QFX5210 switches, and Junos OS

Release 18.2R1 on EX4600 switches, you can configure the UDP port used as the destination port for VXLAN traffic. To configure the VXLAN destination port to be something other than the default UDP port of 4789, enter the following statement:

```
set protocols l2-learning destination-udp-port port-number
```

The port you configure will be used for all VXLANs configured on the switch.

NOTE: If you make this change on one switch in a VXLAN, you must make the same change on all the devices that terminate the VXLANs configured on your switch. If you do not do so, traffic will be disrupted for all the VXLANs configured on your switch. When you change the UDP port, the previously learned remote VTEPs and remote MACs are lost and VXLAN traffic is disrupted until the switch relearns the remote VTEPs and remote MACs.

Controlling Transit Multicast Traffic on QFX5100, QFX5110, QFX5200, QFX5210, and EX4600 Switches

When the switch acting as a VTEP receives a broadcast, unknown unicast, or multicast packet, it performs the following actions on the packet:

1. It de-encapsulates the packet and delivers it to the locally attached hosts.
2. It then adds the VXLAN encapsulation again and sends the packet to the other VTEPs in the VXLAN.

These actions are performed by the loopback interface used as the VXLAN tunnel address and can, therefore, negatively impact the bandwidth available to the VTEP. Starting with Junos OS Release 14.1X53-D30 for QFX5100 switches, Junos OS Release 15.1X53-D210 for QFX5110 and QFX5200 switches, Junos OS Release 18.1R1 for QFX5210 switches, and Junos OS Release 18.2R1 for EX4600 switches, if you know that there are no multicast receivers attached to other VTEPs in the VXLAN that want traffic for a specific multicast group, you can reduce the processing load on the loopback interface by entering the following statement:

```
set protocols l2-learning disable-vxlan-multicast-transit vxlan-multicast-group multicast-group
```

In this case, no traffic will be forwarded for the specified group but all other multicast traffic will be forwarded. If you do not want to forward any multicast traffic to other VTEPs in the VXLAN, enter the following statement:

```
set protocols l2-learning disable-vxlan-multicast-transit vxlan-multicast-group all
```

Using an MX Series Router, EX9200 Switch, or QFX10000 Switch as a VTEP

You can configure an MX Series router, EX9200 switch, or QFX10000 switch to act as a VTEP and perform all of the following roles:

- Act as a Layer 2 gateway between virtualized and nonvirtualized networks in the same data center or between data centers. For example, you can use an MX Series router to connect a network that uses VXLANs to one that uses VLANs.
- Act as a Layer 2 gateway between virtualized networks in the same or different data centers and allow virtual machines to move (VMotion) between those networks and data centers.
- Act as a Layer 3 gateway to route traffic between different VXLANs in the same data center.
- Act as a Layer 3 gateway to route traffic between different VXLANs in different data centers over a WAN or the Internet using standard routing protocols or virtual private LAN service (VPLS) tunnels.

NOTE: If you want one of the devices described in this section to be a VXLAN Layer 3 gateway, you must configure integrated routing and bridging (IRB) interfaces to connect the VXLANs, just as you do if you want to route traffic between VLANs.

Manual VXLANs Require PIM

In an environment with a controller (such as a VMware NSX or Juniper Networks Contrail controller), you can provision VXLANs on a Juniper Networks device. A controller also provides a control plane that VTEPs use to advertise their reachability and learn about the reachability of other VTEPs. You can also manually create VXLANs on Juniper Networks devices instead of using a controller. If you use this approach, you must also configure Protocol Independent Multicast (PIM) on the VTEPs so that they can create VXLAN tunnels between themselves.

You must also configure each VTEP in a given VXLAN to be a member of the same multicast group. (If possible, you should assign a different multicast group address to each VXLAN, although this is not required. Multiple VXLANs can share the same multicast group.) The VTEPs can then forward ARP requests they receive from their connected hosts to the multicast group. The other VTEPs in the group de-encapsulate the VXLAN information, and (assuming they are members of the same VXLAN) they forward the ARP request to their connected hosts. When the target host receives the ARP request, it responds with its MAC address, and its VTEP forwards this ARP reply back to the source VTEP. Through this process, the VTEPs learn the IP addresses of the other VTEPs in the VXLAN and the MAC addresses of the hosts connected to the other VTEPs.

The multicast groups and trees are also used to forward broadcast, unknown unicast, and multicast (BUM) traffic between VTEPs. This prevents BUM traffic from being unnecessarily flooded outside the VXLAN.

NOTE: Multicast traffic that is forwarded through a VXLAN tunnel is sent only to the remote VTEPs in the VXLAN. That is, the encapsulating VTEP does not copy and send copies of the packets according to the multicast tree—it only forwards the received multicast packets to the remote VTEPs. The remote VTEPs de-encapsulate the encapsulated multicast packets and forward them to the appropriate Layer 2 interfaces. Junos OS Release 18.1R1 for QFX5210 switches

Load Balancing VXLAN Traffic

The Layer 3 routes that form VXLAN tunnels use per-packet load balancing by default, which means that load balancing is implemented if there are ECMP paths to the remote VTEP. This is different from normal routing behavior in which per-packet load balancing is not used by default. (Normal routing uses per-prefix load balancing by default.)

The source port field in the UDP header is used to enable ECMP load balancing of the VXLAN traffic in the Layer 3 network. This field is set to a hash of the inner packet fields, which results in a variable that ECMP can use to distinguish between tunnels (flows).

None of the other fields that flow-based ECMP normally uses are suitable for use with VXLANs. All tunnels between the same two VTEPs have the same outer source and destination IP addresses, and the UDP destination port is set to port 4789 by definition. Therefore, none of these fields provide a sufficient way for ECMP to differentiate flows.

VLAN IDs for VXLANs

When configuring a VLAN ID for a VXLAN on any Juniper Networks device that supports VXLANs except QFX10000 switches, we strongly recommend using a VLAN ID of 3 or higher. If you use a VLAN ID of 1 or 2, replicated broadcast, multicast, and unknown unicast (BUM) packets for these VXLANs might be untagged, which in turn might result in the packets being dropped by a device that receives the packets.

Enabling QFX5120 Switches to Tunnel Traffic on Core-Facing Layer 3 Tagged and IRB Interfaces

NOTE: This section applies only to QFX5120 switches running Junos OS Releases 18.4R1, 18.4R2, 18.4R2-S1 through 18.4R2-S3, 19.1R1, 19.1R2, 19.2Rx, and 19.3Rx.

When a QFX5120 switch attempts to tunnel traffic on core-facing Layer 3 tagged interfaces or IRB interfaces, the switch drops the packets. To avoid this issue, you can configure a simple two-term filter-based firewall on the Layer 3 tagged or IRB interface.

NOTE: QFX5120 switches support a maximum of 256 two-term filter-based firewalls.

For example:

```
set interfaces et-0/0/3 unit 0 family inet filter input vxlan100
set firewall family inet filter vxlan100 term 1 from destination-address 192.168.0.1/24 then
accept
set firewall family inet filter vxlan100 term 2 then routing-instance route1
```

Term 1 matches and accepts traffic that is destined for the QFX5210 switch, which is identified by the source VTEP IP address (192.168.0.1/24) assigned to the switch's loopback interface. For term 1, note that when specifying an action, you can alternatively count traffic instead of accepting it.

Term 2 matches and forwards all other data traffic to a routing instance (route 1), which is configured interface et-0/0/3.

In this example, note that interface et-0/0/3 is referenced by routing instance route1. As a result, you must include the `set firewall family inet filter vxlan100 term 2 then routing-instance route1` command. Without this command, the firewall filter will not work properly.

Using ping and traceroute with a VXLAN

On QFX5100 and QFX5110 switches, you can use the `ping` and `traceroute` commands to troubleshoot traffic flow through a VXLAN tunnel by including the `overlay` parameter and various options. You use these options to force the `ping` or `traceroute` packets to follow the same path as data packets through the VXLAN tunnel. In other words, you make the underlay packets (`ping` and `traceroute`) take the same route as the overlay packets (data traffic). See *ping overlay* and *traceroute overlay* for more information.

Supported VXLAN Standards

RFCs and Internet drafts that define standards for VXLAN:

- RFC 7348, *Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks*
- Internet draft draft-ietf-nvo3-vxlan-gpe, *Generic Protocol Extension for VXLAN*

Release History Table

Release	Description
14.1X53-D30	Starting with Junos OS Release 14.1X53-D30 for QFX5100 switches, Junos OS Release 15.1X53-D210 for QFX5110 and QFX5200 switches, Junos OS Release 18.1R1 for QFX5210 switches, and Junos OS Release 18.2R1 for EX4600 switches, if you know that there are no multicast receivers attached to other VTEPs in the VXLAN that want traffic for a specific multicast group, you can reduce the processing load on the loopback interface
14.1X53-D25	Starting with Junos OS Release 14.1X53-D25 on QFX5100 switches, Junos OS Release 15.1X53-D210 on QFX5110 and QFX5200 switches, Junos OS Release 18.1R1 on QFX5210 switches, and Junos OS Release 18.2R1 on EX4600 switches, you can configure the UDP port used as the destination port for VXLAN traffic.

RELATED DOCUMENTATION

Examples: Manually Configuring VXLANs on QFX Series and EX4600 Switches

[Understanding EVPN with VXLAN Data Plane Encapsulation | 398](#)

OVSDb Support on Juniper Networks Devices

[mtu](#)

VXLAN Constraints on QFX Series and EX Series Switches

IN THIS SECTION

- [VXLAN Constraints on QFX5xxx, EX4300-48MP, EX4400, and EX4600 Switches | 424](#)
- [VXLAN Constraints on QFX10000 Switches | 428](#)

When configuring Virtual Extensible LANs (VXLANs) on QFX Series and EX Series switches, be aware of the constraints described in the following sections. In these sections, “Layer 3 side” refers to a network-facing interface that performs VXLAN encapsulation and de-encapsulation, and “Layer 2 side” refers to a server-facing interface that is a member of a VLAN that is mapped to a VXLAN.

VXLAN Constraints on QFX5xxx, EX4300-48MP, EX4400, and EX4600 Switches

- VXLAN support on a Virtual Chassis or Virtual Chassis Fabric (VCF) has the following constraints and recommendations:
 - We support EVPN-VXLAN on an EX4300-48MP Virtual Chassis only in campus networks.
 - Standalone EX4400 switches and EX4400 Virtual Chassis support EVPN-VXLAN. For multihoming use cases, hosts can be multihomed to standalone EX4400 switches, but we don't support multihoming a host with ESI-LAG interfaces to EX4400 Virtual Chassis.
 - In data center networks, we support EVPN-VXLAN only on a Virtual Chassis or VCF consisting of all QFX5100 switches, and not on any other mixed or non-mixed Virtual Chassis or VCF. However, we do not recommend using EVPN-VXLAN on a QFX5100 Virtual Chassis or VCF because the feature support is at parity only with support on standalone QFX5100 switches running Junos OS Release 14.1X53-D40.
 - When a QFX5100 Virtual Chassis learns a MAC address on a VXLAN interface, the MAC table entry can possibly take up to 10 to 15 minutes to age out (two to three times the 5-minute default aging interval). This happens when the Virtual Chassis learns a MAC address from an incoming packet on one Virtual Chassis member switch, and then must forward that packet over the Virtual Chassis port (VCP) links to another member switch in the Virtual Chassis on the way to its destination. The Virtual Chassis marks the MAC address as seen again at the second member switch, so the MAC address might not age out for one or two additional aging intervals beyond the first one. MAC learning can't be disabled on VXLAN interfaces only at the second Virtual Chassis member switch, so you can't avoid the extra delay in this case.
- (QFX5120 switches only) Traffic that is tunneled via a core-facing Layer 3 tagged interface or IRB interface is dropped. To avoid this limitation, you can configure flexible VLAN tagging. For more information, see ["Understanding VXLANs" on page 414](#).
- (QFX5110 and QFX5120) If you configure an interface in the enterprise style with flexible Ethernet services encapsulation, the device drops transit Layer 2 VXLAN-encapsulated packets on that interface. To work around this issue, configure the interface in the service provider style instead of using the enterprise style. For more information on enterprise style and service provider style configurations, see . For an overview of configuring flexible Ethernet services in an EVPN-VXLAN fabric, see ["Understanding Flexible Ethernet Services Support With EVPN-VXLAN" on page 465](#).
- (QFX5110 and QFX5120 switches) In EVPN-VXLAN fabrics, we don't support enterprise style, service provider style, and native VLAN configurations on the same physical interface if the native VLAN is the same as one of the VLANs in the service provider style configuration. The native VLAN can be one of the VLANs in the enterprise style configuration. For more information on enterprise style and service provider style configurations, see .

- (QFX5100, QFX5110, QFX5200, and QFX5210 switches) We support VXLAN configuration in the default-switch routing instance and in MAC VRF routing instances ([instance-type mac-vrf](#)).

(EX4300-48MP and EX4600 switches) We support VXLAN configuration only in the default-switch routing instance.
- (QFX5100, QFX5200, QFX5210, EX4300-48MP, and EX4600 switches) Routing traffic between different VXLANs is not supported.
- (QFX5100, QFX5110, QFX5120, EX4600, and EX4650 switches) These switches support only one VTEP next hop on VXLAN underlay IRB interfaces. If you don't want to use multiple egress ports but need more than one VTEP next hop, as a workaround you can do one of the following:
 - Place a router between the switch and the remote VTEPs so only one next hop is between them.
 - Use physical Layer 3 interfaces instead of IRB interfaces for remote VTEP reachability.
- (QFX5110 switches) By default, routing traffic between a VXLAN and a Layer 3 logical interface—for example, an interface configured with the `set interfaces interface-name unit logical-unit-number family inet address ip-address/prefix-length` command—is disabled. If this routing functionality is required in your EVPN-VXLAN network, you can perform some additional configuration to make it work. For more information, see ["Understanding How to Configure VXLANs and Layer 3 Logical Interfaces to Interoperate"](#) on page 453.
- Integrated routing and bridging (IRB) interfaces used in EVPN-VXLAN overlay networks do not support the IS-IS routing protocol.
- (QFX5100, QFX5110, QFX5120, QFX5200, QFX5210, EX4300-48MP, and EX4600 switches) A physical interface cannot be a member of a VLAN and a VXLAN. That is, an interface that performs VXLAN encapsulation and de-encapsulation cannot also be a member of a VLAN. For example, if a VLAN that is mapped to a VXLAN is a member of trunk port xe-0/0/0, any other VLAN that is a member of xe-0/0/0 must also be assigned to a VXLAN.

NOTE: Starting in Junos OS Releases 18.1R3 and 18.4R1 for QFX5100, QFX5110, QFX5200, and QFX5210 switches and Junos OS Release 19.4R1 for QFX5120 and EX4650 switches, this limitation no longer applies because you can configure flexible Ethernet services on the physical interface. For more information, see ["Understanding Flexible Ethernet Services Support With EVPN-VXLAN"](#) on page 465.

- Multichassis link aggregation groups (MC-LAGs) are not supported with VXLAN.

NOTE: In an EVPN-VXLAN environment, EVPN multihoming active-active mode is used instead of MC-LAG for redundant connectivity between hosts and leaf devices.

- IP fragmentation and defragmentation are not supported on the Layer 3 side.
- The following features are not supported on the Layer 2 side:
 - (QFX5100, QFX5200, QFX5210, EX4300-48MP, and EX4600 switches) IGMP snooping with EVPN-VXLAN.
 - Redundant trunk groups (RTGs).
 - The ability to shut down a Layer 2 interface or temporarily disable the interface when a storm control level is exceeded is not supported.
 - STP (any variant).
- Access port security features such as the following are not supported with VXLAN:
 - DHCP snooping.
 - Dynamic ARP inspection.
 - MAC limiting and MAC move limiting.

NOTE: Exceptions:

MAC limiting is supported on OVSDB-managed interfaces in an OVSDB-VXLAN environment with Contrail controllers. For more information, see *Features Supported on OVSDB-Managed Interfaces*.

- Ingress node replication is not supported in the following cases:
 - When PIM is used for the control plane (manual VXLAN).
 - When an SDN controller is used for the control plane (OVSDB-VXLAN).

Ingress node replication is supported with EVPN-VXLAN.

- PIM-BIDIR and PIM-SSM are not supported with VXLANs.
- If you configure a port-mirroring instance to mirror traffic exiting from an interface that performs VXLAN encapsulation, the source and destination MAC addresses of the mirrored packets are invalid. The original VXLAN traffic is not affected.

- (QFX5110 switches only) VLAN firewall filters are not supported on IRB interfaces on which EVPN-VXLAN is enabled.
- (QFX5100 and QFX5110 switches) Firewall filters and policers are not supported on transit traffic on which EVPN-VXLAN is enabled. They are supported only in the ingress direction on CE-facing interfaces.
- (QFX5100 and QFX5110 switches) For IRB interfaces in an EVPN-VXLAN one-layer IP fabric, firewall filtering and policing is supported only at the ingress point of non-encapsulated frames routed through the IRB interface.
- (EX4300-48MP switches only) The following styles of interface configuration are not supported:
 - Service provider style, where a physical interface is divided into multiple logical interfaces, each of which is dedicated to a particular customer VLAN. The `extended-vlan-bridge` encapsulation type is configured on the physical interface.
 - Flexible Ethernet services, which is an encapsulation type that enables a physical interface to support both service provider and enterprise styles of interface configuration.

For more information about these styles of interface configuration, see [Flexible Ethernet Services Encapsulation](#).

- (QFX5100 switches only) Using the `no-arp-suppression` configuration statement, you can disable the suppression of ARP requests on one or more specified VLANs. However, starting in Junos OS Release 18.4R3, you must disable this feature on all VLANs. To do so, you can use one of these configuration options:
 - Use a batch command to disable the feature on all VLANs—`set groups group-name vlans * no-arp-suppression`. With this command, we use the asterisk (*) as a wildcard that specifies all VLANs.
 - Use a command to disable the feature on each individual VLAN—`set vlans vlan-name no-arp-suppression`.
- QFX5120-48Y, QFX5120-32C, and QFX5200 switches support hierarchical equal-cost multipath (ECMP), which allows these switches to perform a two-level route resolution. However, all other QFX5xxx switches do not support hierarchical ECMP. As a result, when an EVPN Type-5 packet is encapsulated with a VXLAN header, then de-encapsulated by a QFX5xxx switch that does not support hierarchical ECMP, the switch is unable to resolve the two-levels of routes that were in the inner packet. The switch then drops the packet.
- (QFX5100, QFX5110, QFX5120, QFX5200, and QFX5210 switches) When you configure the IRB interfaces on a device that is concurrently supporting configured VLANs on both a centrally-routed bridging overlay and an edge-routed bridging overlay, the IRB MAC address and virtual gateway MAC address on the IRB interface for the centrally-routed bridging overlay must be different from

the IRB MAC address and virtual gateway MAC address on the IRB interface for the edge-routed bridging overlay.

- (QFX5110 and QFX5120 switches only) In an EVPN-VXLAN overlay, we do not support running a routing protocol on an IRB interface that is in a default routing instance (default.inet.0). Instead, we recommend including the IRB interface in a routing instance of type vrf.
- The following constraints apply to both VXLANs and VLANs.
 - (QFX5xxx switches only) When configuring route leaking between virtual routing and forwarding (VRF) instances, you must specify each prefix with a subnet mask that is equal to or longer than /16 for IPv4 prefixes or equal to or longer than /64 for IPv6 prefixes. If a subnet mask that you specify does not meet these parameters, the specified routes will not be leaked.
 - (QFX5120 switches only) By default, QFX5120 switches allocate 5 MB of a shared buffer to absorb bursts of multicast traffic. If a burst of multicast traffic exceeds 5 MB, the switch will drop the packets that the switch receives after the buffer space is exceeded. To prevent the switch from dropping multicast packets when this situation occurs, you can do one of the following:
 - Use the [shared-buffer](#) configuration statement in the [edit class-of-service] hierarchy level to reallocate a higher percentage of the shared buffer for multicast traffic. For more information about fine-tuning a shared buffer, see [Understanding CoS Buffer Configuration](#).
 - On the Juniper Networks switch from which the bursts of multicast traffic are coming, apply a shaper on the egress link.
 - (QFX5xxx switches only) If you have enabled storm control on an interface, you might notice a significant difference between the configured and actual traffic rates for the interface. This difference is the result of an internal storm control meter that quantifies the actual traffic rate for the interface in increments of 64 kbps, for example, 64 kbps, 128 kbps, 192 kbps, and so on.

VXLAN Constraints on QFX10000 Switches

- MC-LAGs are not supported with VXLAN.

NOTE: In an EVPN-VXLAN environment, EVPN multihoming active-active mode is used instead of MC-LAG for redundant connectivity between hosts and leaf devices.

- IP fragmentation is not supported on the Layer 3 side.
- The following features are not supported on the Layer 2 side:
 - IGMP snooping with EVPN-VXLAN in Junos OS Releases before Junos OS Release 17.2R1.
 - STP (any variant).

- Access port security features are not supported with VXLAN. For example, the following features are not supported:
 - DHCP snooping.
 - Dynamic ARP inspection.
 - MAC limiting and MAC move limiting.
- Ingress node replication is not supported when an SDN controller is used for the control plane (OVSDB-VXLAN). Ingress node replication is supported for EVPN-VXLAN.
- QFX10000 switches that are deployed in an EVPN-VXLAN environment do not support an IPv6 physical underlay network.
- When the next-hop database on a QFX10000 switch includes next hops for both the underlay network and the EVPN-VXLAN overlay network, the next hop to a VXLAN peer cannot be an Ethernet segment identifier (ESI) or a virtual tunnel endpoint (VTEP) interface.
- IRB interfaces used in EVPN-VXLAN overlay networks do not support the IS-IS routing protocol.
- VLAN firewall filters applied to IRB interfaces on which EVPN-VXLAN is enabled.
- Filter-based forwarding (FBF) is not supported on IRB interfaces used in an EVPN-VXLAN environment.
- QFX10002, QFX10008, and QFX10016 switches do not support port mirroring and analyzing when EVPN-VXLAN is also configured.
- When you configure the IRB interfaces on a device that is concurrently supporting configured VLANs on both a centrally-routed bridging overlay and an edge-routed bridging overlay, the IRB MAC address and virtual gateway MAC address on the IRB interface for the centrally-routed bridging overlay must be different from the IRB MAC address and virtual gateway MAC address on the IRB interface for the edge-routed bridging overlay.
- In an EVPN-VXLAN overlay, we do not support running a routing protocol on an IRB interface that is in a default routing instance (default.inet.0). Instead, we recommend including the IRB interface in a routing instance of type vrf.

RELATED DOCUMENTATION

[Understanding VXLANs | 414](#)

Examples: Manually Configuring VXLANs on QFX Series and EX4600 Switches

Manually Configuring VXLANs on QFX Series and EX4600 Switches

EVPN Over VXLAN Encapsulation Configuration Overview for QFX Series and EX4600 Switches

This topic provides information about configuring Ethernet VPN (EVPN) with Virtual Extensible Local Area Networks (VXLAN) data plane encapsulation on QFX5100, QFX5110, QFX5200, QFX5210, and EX4600 switches.

The configuration statements described in this topic are in the `switch-options` and `protocols evpn` hierarchy levels.

CLI Statement	Description
<code>route-distinguisher</code>	Specifies an identifier attached to a route—this enables you to distinguish to which VPN or VPLS the route belongs. Each routing instance must have a unique route distinguisher associated with it. The route distinguisher is used to place bounds around a VPN so that the same IP address prefixes can be used in different VPNs without having them overlap.
<code>vrf-target</code>	Specifies a VRF target community. In effect, this statement configures a single policy for import and a single policy for export to replace the per-VRF policies for every community. <code>import</code> , <code>export</code> , and <code>auto</code> options are available.
<code>vrf-import</code>	Specifies how routes are imported into the VRF table of the local provider edge (PE) router or switch from the remote PE.
<code>vrf-export</code>	Specifies how routes are exported from the local PE router's VRF table to the remote PE router.

(Continued)

CLI Statement	Description
<code>designated-forwarder-election-hold-time</code>	Establishes when a designated forwarder is required for customer edge (CE) devices that are multihomed to more than one provider edge (PE) device. Without a designated forwarder, multihomed hosts receive duplicate packets. Designated forwarders are chosen for an Ethernet segment identifier (ESI) based on type-4 route advertisements.
<code>encapsulation vxlan</code>	Configures a VXLAN encapsulation type.
<code>extended-vni-list</code> and <code>extended-vni-all</code>	Establishes which VXLAN virtual network identifiers are designated as part of the virtual switch instance.
<code>multicast-mode</code>	Configures the multicast server mode for delivering traffic and packets for EVPN.
<code>vni-options</code>	Configures different route targets for each VXLAN virtual network identifier instance under <code>vni-options</code> .
<code>show route table</code>	Displays both imported EVPN routes, and export/import EVPN routes for the default-switch routing instance.

(Continued)

CLI Statement	Description
<code>show configuration protocols evpn</code>	Displays results of the <code>extended-vni-list</code> and <code>vni-options</code> statements.

RELATED DOCUMENTATION

EVPN Multihoming Overview 96
Understanding VXLANs 414
EVPN-over-VXLAN Supported Functionality 408
Example: Configure an EVPN-VXLAN Centrally-Routed Bridging (CRB) Using MX Routers as Spines 580

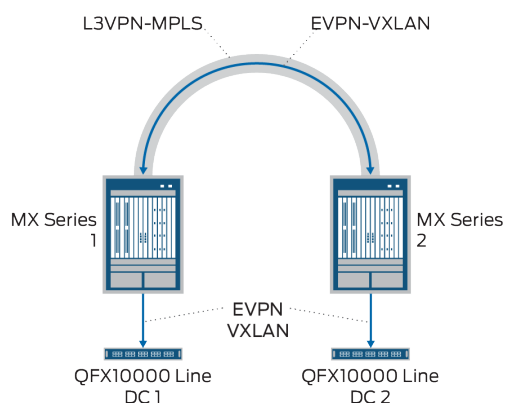
Implementing EVPN-VXLAN for Data Centers

Although there are various Data center interconnect (DCI) technologies available, EVPN has an added advantage over other MPLS technologies because of its unique features, such as active/active redundancy, aliasing, and mass MAC withdrawal. To provide a DCI solution, VXLAN is integrated with EVPN.

There are different options for using EVPN-VXLAN with DCI:

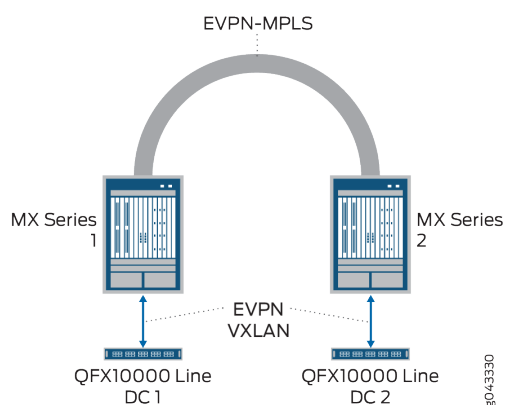
- DCI can connect multiple data centers in your WAN using MX Series edge routers with a Layer 3 VPN MPLS network between them. QFX10000 switches start and stop the VXLAN tunnel. This option requires no changes to your WAN.

Figure 31: DCI Option: Layer 3 VPN-MPLS



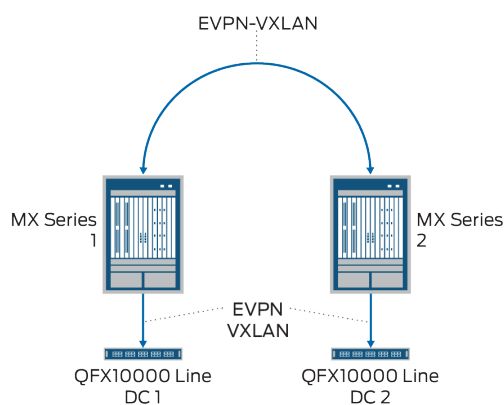
- A second option connects multiple data centers in your WAN using either MX Series edge routers or supported QFX Series switches with an EVPN-MPLS network between them. This option uses an EVPN control plane and an MPLS data plane and requires changes to your WAN. You must change your LAN architecture to natively support EVPN, and you must implement EVPN stitching between each MX router/QFX Series switch and the corresponding QFX10000 switch. For details about releases where QFX Series switches are supported, see <https://pathfinder.juniper.net/feature-explorer> and then search on EVPN.

Figure 32: DCI Option: EVPN-MPLS



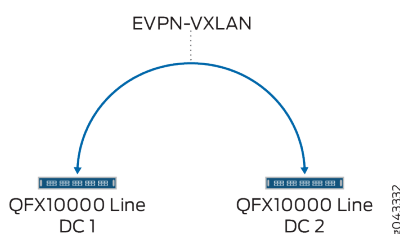
- You can also tunnel two branch locations across the Internet. In this case, implementation requires neither a traditional WAN nor MPLS. This method can use the Internet or an IP tunnel, where VXLAN rides on top of IP and EVPN is used throughout.

Figure 33: DCI Option: EVPN-VXLAN over the Internet



- If you do not have a branch router or a peering router, you can simply connect the data centers directly and EVPN is again used natively throughout. This implementation requires neither a traditional WAN nor MPLS, but you typically need a dark fiber connection.

Figure 34: DCI Option: Layer 3 VPN-MPLS Direct Connection



You can alternately create an EVPN-VXLAN fabric internally in the data center using bare-metal servers and/or virtual servers and using OpenClo for management. Here you also use VXLAN L2 gateways and L3 gateways on switches such as a QFX10000 switch. The underlying fabric is built on BGP.

EVPN-VXLAN uses both routers and switches—the configurations are the same for both devices but they are located in different areas of the Junos OS CLI. MX Series routers are configured under a routing instance with the instance type `virtual switch`. QFX Series switches are configured under global `switching-options` and global protocol `evpn`. See [Table 13 on page 435](#) for a list of CLI commands used by EVPN-VXLAN.

Table 13: CLI Commands for EVPN-VXLAN

Function	CLI Command
Specifies an identifier attached to a route. This enables you to distinguish to which VPN or VPLS the route belongs. Each routing instance must have a unique route distinguisher (RD) associated with it. The RD is used to place bounds around a VPN so that the same IP address prefixes can be used in different VPNs without having them overlap.	"route-distinguisher" on page 1651
Specifies a VRF target community. In effect, this statement configures a single policy for import and a single policy for export to replace the per-VRF policies for every community. The options <code>import</code> and <code>export</code> apply to both routers and QFX Series switches. The option <code>auto</code> applies to QFX Series switches only.	"vrf-target" on page 1670
Specifies how routes are imported into the VRF table of the local PE router or switch from the remote PE router.	"vrf-import" on page 1668
Specifies how routes are exported from the local PE router's VRF table to the remote PE router.	"vrf-export" on page 1667
A designated forwarder (DF) is required when CEs are multihomed to more than one PE. Without a designated forwarder, multihomed hosts would receive duplicate packets. Designated forwarders are chosen for an Ethernet segment identifier (ESI) based on type-4 route advertisements.	"designated-forwarder-election-hold-time" on page 1552
Configures a logical link-layer encapsulation type.	<i>encapsulation</i>
Establishes which VXLAN virtual network identifiers (VNIs) will be part of the EVPN-VXLAN MP-BGP domain. There are different BUM replication options available in EVPN—using <code>extended-vni-list</code> forgoes a multicast underlay in favor of EVPN-VXLAN ingress replication.	"extended-vni-list" on page 1680
You configure different route targets (RTs) for each VNI instance under <code>"vni-options"</code> on page 1706.	"vni-options" on page 1706 (QFX Series switches only)
Displays both imported EVPN routes and export/import EVPN routes for the default switch routing instances.	"show route table" on page 2003

Table 13: CLI Commands for EVPN-VXLAN (Continued)

Function	CLI Command
Displays results of the configuration commands "extended-vni-list" on page 1680 and "vni-options" on page 1706 .	show configuration protocols evpn

RELATED DOCUMENTATION

[Understanding EVPN with VXLAN Data Plane Encapsulation](#) | 398

PIM NSR and Unified ISSU Support for VXLAN Overview

Starting in Junos OS Release 16.2R1, Protocol Independent Multicast (PIM) nonstop active routing (NSR) support for Virtual Extensible LAN (VXLAN) is supported on MX Series routers.

The Layer 2 address learning daemon (l2ald) passes VXLAN parameters (VXLAN multicast group addresses and the source interface for a VXLAN tunnel [vtep-source-interface]) to the routing protocol process on the primary Routing Engine. The routing protocol process forms PIM joins with the multicast routes through the pseudo-VXLAN interface based on these configuration details.

Because the l2ald daemon does not run on the backup Routing Engine, the configured parameters are not available to the routing protocol process in the backup Routing Engine when NSR is enabled. The PIM NSR mirroring mechanism provides the VXLAN configuration details to the backup Routing Engine, which enables creation of the required states. The routing protocol process matches the multicast routes on the backup Routing Engine with PIM states, which maintains the multicast routes in the Forwarding state.

In response to Routing Engine switchover, the multicast routes remain in the Forwarding state on the new primary Routing Engine. This prevents traffic loss during Routing Engine switchover. When the l2ald process becomes active, it refreshes VXLAN configuration parameters to PIM.

NOTE: For this feature, NSR support is available for VXLAN in PIM sparse mode.

This feature does not introduce any new CLI commands. You can issue the following `show` commands on the backup Routing Engine to monitor the PIM joins and multicast routes on the backup Routing Engine:

- `show pim join extensive`

- `show multicast route extensive`

Unified ISSU Support

Starting in Junos OS Release 17.2 R1, unified in-service software upgrade is supported for VXLAN using PIM on MX Series routers. ISSU enables you to upgrade your Junos OS software on your MX Series router with no disruption on the control plane and with minimal disruption of traffic. Unified ISSU is supported only on dual Routing Engine platforms. The graceful Routing Engine switchover (GRES) and nonstop active routing (NSR) features must both be enabled. Unified ISSU allows you to eliminate network downtime, reduce operating costs, and deliver higher levels of services. See [Getting Started with Unified In-Service Software Upgrade](#).

NOTE: Unified ISSU is not supported on the QFX series switches.

To enable GRES, include the `graceful-switchover` statement at the `[edit chassis redundancy]` hierarchy level.

To enable NSR, include the `nonstop-routing` statement at the `[edit routing-options]` hierarchy level and the `commit synchronize` statement at the `[edit system]` hierarchy level.

Release History Table

Release	Description
17.2R1	Starting in Junos OS Release 17.2 R1, unified in-service software upgrade is supported for VXLAN using PIM on MX Series routers.
16.2R1	Starting in Junos OS Release 16.2R1, Protocol Independent Multicast (PIM) nonstop active routing (NSR) support for Virtual Extensible LAN (VXLAN) is supported on MX Series routers.

RELATED DOCUMENTATION

<code>show pim join</code>
<code>show multicast route</code>
Understanding Graceful Routing Engine Switchover

Routing IPv6 Data Traffic through an EVPN-VXLAN Network with an IPv4 Underlay

IN THIS SECTION

- [Benefits of Routing IPv6 Data Traffic through an EVPN-VXLAN Network With an IPv4 Underlay | 439](#)
- [Centrally-Routed Bridging Overlay—How to Set Up the Routing of IPv6 Data Traffic | 439](#)
- [Edge-Routed Bridging Overlay—How to Set Up the Routing of IPv6 Data Traffic | 449](#)

QFX10000 and QFX5110 switches can function as Layer 3 VXLAN gateways in an EVPN-VXLAN overlay network. Starting with Junos OS Release 15.1X53-D30 for QFX10000 switches and Junos OS Release 18.4R1 for QFX5110 switches, the IRB interfaces on these Layer 3 VXLAN gateways can route Layer 2 or 3 data packets from one IPv6 host to another IPv6 host through an EVPN-VXLAN network with an IPv4 underlay.

Upon receipt of a Layer 2 or 3 data packet from an IPv6 host, a Layer 3 VXLAN gateway encapsulates the packet with an IPv4 outer header, thereby tunneling the packet through the IPv4 underlay network. The Layer 2 or 3 VXLAN gateway at the other end of the tunnel de-encapsulates the packet and forwards the packet towards the other IPv6 host.

The Layer 3 VXLAN gateways in the EVPN-VXLAN overlay network learn the IPv6 routes through the exchange of EVPN Type-2 and Type-5 routes.

This feature is supported in EVPN-VXLAN overlay networks with the following architectures, which are commonly deployed within a data center:

- **EVPN-VXLAN centrally-routed bridging overlay (EVPN-VXLAN topology with a two-layer IPv4 fabric)**—A two-layer IPv4 fabric in which QFX10000 or QFX5110 switches function as Layer 3 VXLAN gateways, and QFX5100 or QFX5200 switches function as Layer 2 VXLAN gateways. In this architecture, the IRB interfaces configured on the Layer 3 VXLAN gateways function in a central location in the fabric.
- **EVPN-VXLAN edge-routed bridging overlay (EVPN-VXLAN topology with a collapsed IPv4 fabric)**—A one-layer IPv4 fabric in which QFX10000 or QFX5110 switches function as both Layer 2 and Layer 3 VXLAN gateways. In this architecture, the IRB interfaces configured on the Layer 3 VXLAN gateways function at the edge of the fabric.

This topic includes the following sections:

Benefits of Routing IPv6 Data Traffic through an EVPN-VXLAN Network With an IPv4 Underlay

Routing IPv6 data traffic through an EVPN-VXLAN network with an IPv4 underlay provides the following benefits:

- Eliminates the need to deploy an IPv6 underlay network. The supported IPv4 fabric architectures are agile enough to support both IPv4 and IPv6 data traffic needs.
- Leverages the EVPN control plane's inherent support for exchanging IPv4 and IPv6 routes.
- Does not introduce any new configuration. To set up this feature, you configure IRB interfaces with IPv6 addresses.

Centrally-Routed Bridging Overlay—How to Set Up the Routing of IPv6 Data Traffic

NOTE: The focus of this section is the configuration of IRB interfaces on Layer 3 VXLAN gateways. For a more comprehensive example of configuring EVPN-VXLAN, see ["Example: Configure an EVPN-VXLAN Centrally-Routed Bridging \(CRB\) Overlay" on page 530](#).

The key to setting up this feature is the configuration of IRB interfaces on Layer 3 VXLAN gateways. In general, you configure the IRB interfaces as you would with only IPv4 hosts in the topology. However, in addition to specifying IPv4 addresses for the IRB interface and default Layer 3 gateway (virtual gateway), you also specify IPv6 addresses.

[Table 14 on page 439](#) shows how to configure the addresses for IRB interfaces irb.100 and irb.200 on the three Layer 3 VXLAN gateways. [Table 15 on page 446](#) outlines some additional required global configuration and configuration for irb.100 and irb.200.

Table 14: Centrally-Routed Bridging Overlay—Sample IRB Interface Addresses

Addresses	Layer 3 VXLAN Gateway 1	Layer 3 VXLAN Gateway 2	Layer 3 VXLAN Gateway 3	Description
IRB interface irb.100				
IRB IPv4 address	192.168.100.1/24	192.168.100.2/24	192.168.100.3/24	Specify a different IPv4 address for irb.100 on each device.

Table 14: Centrally-Routed Bridging Overlay—Sample IRB Interface Addresses (Continued)

Addresses	Layer 3 VXLAN Gateway 1	Layer 3 VXLAN Gateway 2	Layer 3 VXLAN Gateway 3	Description
IRB global IPv6 address	2001:db8::192.168. 100.1/96	2001:db8::192.168. 100.2/96	2001:db8::192.168. 100.3/96	Specify a different IPv6 address for irb.100 on each device.
IRB link-local IPv6 address	fe80::100:00:01/96	fe80::100:00:01/96	fe80::100:00:01/96	Specify the same link-local IPv6 address for irb.100 on each device. Any packet destined to this IPv6 address is intercepted for Network Discovery Protocol (NDP) processing.
IRB link-local IPv6 address (DHCPv6 or SLAAC)	fe80::100:00:11/96	fe80::100:00:12/96	fe80::100:00:13/96	Specify a different link local address on each device when using DHCPv6 or SLAAC in the EVPN Fabric. You must also configure the router advertisement for this IRB.
Virtual gateway IPv4 address	192.168.100.254/2 4	192.168.100.254/2 4	192.168.100.254/2 4	Specify the same IPv4 anycast address for the virtual gateway on each device.

Table 14: Centrally-Routed Bridging Overlay—Sample IRB Interface Addresses *(Continued)*

Addresses	Layer 3 VXLAN Gateway 1	Layer 3 VXLAN Gateway 2	Layer 3 VXLAN Gateway 3	Description
Virtual gateway IPv6 address	2001:db8::192.168. 100.254/96	2001:db8::192.168. 100.254/96	2001:db8::192.168. 100.254/96	Specify the same IPv6 anycast address for the virtual gateway on each device.
Virtual gateway IPv6 address (DHCPv6 or SLAAC)	fe80::100:00:100	fe80::100:00:100	fe80::100:00:100	Specify the same virtual-gateway- address for the link local on all the devices when using DHCPV6 or SLAAC in the EVPN Fabric. You must also configure the router advertisement for this IRB.

Table 14: Centrally-Routed Bridging Overlay—Sample IRB Interface Addresses (*Continued*)

Addresses	Layer 3 VXLAN Gateway 1	Layer 3 VXLAN Gateway 2	Layer 3 VXLAN Gateway 3	Description
Virtual gateway IPv4 and IPv6 MAC addresses	Method 1	Method 1	Method 1	<p>For the default Layer 3 gateway, the IPv4 and IPv6 MAC addresses can be the same or different, as long as they are consistent across the three devices. Here are the supported options:</p> <ul style="list-style-type: none"> Method 1—you explicitly configure the same IPv4 and IPv6 MAC address on each device. Method 2—you explicitly configure different IPv4 and IPv6 MAC addresses on each device. Method 3—you do not explicitly configure IPv4 and IPv6 MAC addresses, and the system automatically generates 00:00:5e:00:01:01 as the IPv4 MAC address
	<ul style="list-style-type: none"> IPv4 MAC address: 10:00:00:00:00:fe IPv6 MAC address: 10:00:00:00:00:fe 	<ul style="list-style-type: none"> IPv4 MAC address: 10:00:00:00:00:fe IPv6 MAC address: 10:00:00:00:00:fe 	<ul style="list-style-type: none"> IPv4 MAC address: 10:00:00:00:00:fe IPv6 MAC address: 10:00:00:00:00:fe 	
	Method 2	Method 2	Method 2	
	<ul style="list-style-type: none"> IPv4 MAC address: 10:00:00:00:00:fe IPv6 MAC address: 10:00:00:00:00:f 	<ul style="list-style-type: none"> IPv4 MAC address: 10:00:00:00:00:fe IPv6 MAC address: 10:00:00:00:00:f 	<ul style="list-style-type: none"> IPv4 MAC address: 10:00:00:00:00:fe IPv6 MAC address: 10:00:00:00:00:f 	
	Method 3	Method 3	Method 3	
	<ul style="list-style-type: none"> IPv4 MAC address: 00:00:5e:00:01:01 IPv6 MAC address: 00:00:5e:00:02:01 	<ul style="list-style-type: none"> IPv4 MAC address: 00:00:5e:00:01:01 IPv6 MAC address: 00:00:5e:00:02:01 	<ul style="list-style-type: none"> IPv4 MAC address: 00:00:5e:00:01:01 IPv6 MAC address: 00:00:5e:00:02:01 	

Table 14: Centrally-Routed Bridging Overlay—Sample IRB Interface Addresses (Continued)

Addresses	Layer 3 VXLAN Gateway 1	Layer 3 VXLAN Gateway 2	Layer 3 VXLAN Gateway 3	Description
				and 00:00:5e:00:02: 01 as the IPv6 MAC address.

IRB interface irb.200

IRB IPv4 address	192.168.200.1/24	192.168.200.2/24	192.168.200.3/24	Specify a different IPv4 address for irb.200 on each device.
IRB global IPv6 address	2001:db8::192.168.200.1/96	2001:db8::192.168.200.2/96	2001:db8::192.168.200.3/96	Specify a different IPv6 address for irb.200 on each device.
IRB link-local IPv6 address	fe80::200:00:01/96	fe80::200:00:01/96	fe80::200:00:01/96	Specify the same link-local IPv6 address for irb.200 on each device. Any packet destined to this IPv6 address is intercepted for NDP processing.
IRB link-local IPv6 address (DHCPv6 or SLAAC)	fe80::200:00:11/96	fe80::200:00:12/96	fe80::200:00:13/96	Specify a different link local address on each device when using DHCPV6 or SLAAC in the EVPN Fabric. You must also configure the router advertisement for this IRB.

Table 14: Centrally-Routed Bridging Overlay—Sample IRB Interface Addresses *(Continued)*

Addresses	Layer 3 VXLAN Gateway 1	Layer 3 VXLAN Gateway 2	Layer 3 VXLAN Gateway 3	Description
Virtual gateway IPv4 address	192.168.200.254/24	192.168.200.254/24	192.168.200.254/24	Specify the same IPv4 anycast address for the virtual gateway on each device.
Virtual gateway IPv6 address	2001:db8::192.168.200.254/96	2001:db8::192.168.200.254/96	2001:db8::192.168.200.254/96	Specify the same IPv6 anycast address for the virtual gateway on each device.
Virtual gateway IPv6 address (DHCPv6 or SLAAC)	fe80::100:00:200	fe80::100:00:200	fe80::100:00:200	Specify the same virtual-gateway-address for the link local on all the devices when using DHCPV6 or SLAAC in the EVPN Fabric. You must also configure the router advertisement for this IRB.

Table 14: Centrally-Routed Bridging Overlay—Sample IRB Interface Addresses (*Continued*)

Addresses	Layer 3 VXLAN Gateway 1	Layer 3 VXLAN Gateway 2	Layer 3 VXLAN Gateway 3	Description
Virtual gateway IPv4 and IPv6 MAC addresses	Method 1	Method 1	Method 1	<p>For the default Layer 3 gateway, the IPv4 and IPv6 MAC addresses can be the same or different, as long as they are consistent across the three devices. Here are the supported options:</p> <ul style="list-style-type: none"> Method 1—you explicitly configure the same IPv4 and IPv6 MAC address on each device. Method 2—you explicitly configure different IPv4 and IPv6 MAC addresses on each device. Method 3—you do not explicitly configure IPv4 and IPv6 MAC addresses, and the system automatically generates 00:00:5e:00:01:01 as the IPv4 MAC address
	<ul style="list-style-type: none"> IPv4 MAC address: 10:00:00:00:00:fe IPv6 MAC address: 10:00:00:00:00:fe 	<ul style="list-style-type: none"> IPv4 MAC address: 10:00:00:00:00:fe IPv6 MAC address: 10:00:00:00:00:fe 	<ul style="list-style-type: none"> IPv4 MAC address: 10:00:00:00:00:fe IPv6 MAC address: 10:00:00:00:00:fe 	
	Method 2	Method 2	Method 2	
	<ul style="list-style-type: none"> IPv4 MAC address: 10:00:00:00:00:fe IPv6 MAC address: 10:00:00:00:00:ff 	<ul style="list-style-type: none"> IPv4 MAC address: 10:00:00:00:00:fe IPv6 MAC address: 10:00:00:00:00:ff 	<ul style="list-style-type: none"> IPv4 MAC address: 10:00:00:00:00:fe IPv6 MAC address: 10:00:00:00:00:ff 	
	Method 3	Method 3	Method 3	
	<ul style="list-style-type: none"> IPv4 MAC address: 00:00:5e:00:01:01 IPv6 MAC address: 00:00:5e:00:02:01 	<ul style="list-style-type: none"> IPv4 MAC address: 00:00:5e:00:01:01 IPv6 MAC address: 00:00:5e:00:02:01 	<ul style="list-style-type: none"> IPv4 MAC address: 00:00:5e:00:01:01 IPv6 MAC address: 00:00:5e:00:02:01 	

Table 14: Centrally-Routed Bridging Overlay—Sample IRB Interface Addresses *(Continued)*

Addresses	Layer 3 VXLAN Gateway 1	Layer 3 VXLAN Gateway 2	Layer 3 VXLAN Gateway 3	Description
				and 00:00:5e:00:02: 01 as the IPv6 MAC address.

Table 15: Centrally-Routed Bridging Overlay—Required IRB Interface Configuration

Description	Layer 3 VXLAN Gateway 1	Layer 3 VXLAN Gateway 2	Layer 3 VXLAN Gateway 3
-------------	-------------------------	-------------------------	-------------------------

Global IRB interface configuration

Specify that the IPv4 and IPv6 MAC addresses of the default Layer 3 gateway are advertised to the Layer 2 VXLAN gateways without the extended community option.	set protocols evpn default-gateway no-gateway-community	set protocols evpn default-gateway no-gateway-community	set protocols evpn default-gateway no-gateway-community
---	---	---	---

IRB interface irb.100 configuration

Table 15: Centrally-Routed Bridging Overlay—Required IRB Interface Configuration (Continued)

Description	Layer 3 VXLAN Gateway 1	Layer 3 VXLAN Gateway 2	Layer 3 VXLAN Gateway 3
Configure the Layer 3 VXLAN gateway to advertise the MAC and IP routes (MAC+IP type 2 routes) on behalf of the Layer 2 VXLAN gateways.	set interfaces irb unit 100 proxy-macip-advertisement	set interfaces irb unit 100 proxy-macip-advertisement	set interfaces irb unit 100 proxy-macip-advertisement
Enable the default Layer 3 gateway to be pinged by either IPv4 or IPv6 addresses.	set interfaces irb unit 100 virtual-gateway-accept-data set interfaces irb unit 100 family inet address 192.168.100.1/24 preferred set interfaces irb unit 100 family inet6 address 2001:db8::192.168.100.1/96 preferred	set interfaces irb unit 100 virtual-gateway-accept-data set interfaces irb unit 100 family inet address 192.168.100.2/24 preferred set interfaces irb unit 100 family inet6 address 2001:db8::192.168.100.2/96 preferred	set interfaces irb unit 100 virtual-gateway-accept-data set interfaces irb unit 100 family inet address 192.168.100.3/24 preferred set interfaces irb unit 100 family inet6 address 2001:db8::192.168.100.3/96 preferred

IRB interface irb.200 configuration

Configure the Layer 3 VXLAN gateway to advertise the MAC and IP routes (MAC+IP type 2 routes) on behalf of the Layer 2 VXLAN gateways.	set interfaces irb unit 200 proxy-macip-advertisement	set interfaces irb unit 200 proxy-macip-advertisement	set interfaces irb unit 200 proxy-macip-advertisement
--	--	--	--

Table 15: Centrally-Routed Bridging Overlay—Required IRB Interface Configuration (Continued)

Description	Layer 3 VXLAN Gateway 1	Layer 3 VXLAN Gateway 2	Layer 3 VXLAN Gateway 3
Enable the default Layer 3 gateway to be pinged by either IPv4 or IPv6 addresses.	<pre>set interfaces irb unit 200 virtual-gateway-accept-data set interfaces irb unit 200 family inet address 192.168.200.1/24 preferred set interfaces irb unit 200 family inet6 address 2001:db8::192.168.200.1/96 preferred</pre>	<pre>set interfaces irb unit 200 virtual-gateway-accept-data set interfaces irb unit 200 family inet address 192.168.200.2/24 preferred set interfaces irb unit 200 family inet6 address 2001:db8::192.168.200.2/96 preferred</pre>	<pre>set interfaces irb unit 200 virtual-gateway-accept-data set interfaces irb unit 200 family inet address 192.168.200.3/24 preferred set interfaces irb unit 200 family inet6 address 2001:db8::192.168.200.3/96 preferred</pre>
DHCPv6 or SLAAC configuration			
Configure the linked local address.	<pre>set interfaces irb.100 family inet6 address fe80::100:00:11/96 virtual- gateway-address fe80::100:00:100 set interfaces irb.200 family inet6 address fe80::200:00:11/96 fe80::200:00:100</pre>	<pre>set interfaces irb.100 family inet6 address fe80::100:00:12/96 virtual- gateway-address fe80::100:00:100 set interfaces irb.200 family inet6 address fe80::200:00:12/96 fe80::200:00:100</pre>	<pre>set interfaces irb.100 family inet6 address fe80::100:00:13/96 virtual- gateway-address fe80::100:00:100 set interfaces irb.200 family inet6 address fe80::200:00:13/96 fe80::200:00:100</pre>
Configure the global address.	<pre>set interfaces irb.100 family inet6 address 2001:db8::192.168.100.1/96 virtual-gateway-address 2001:db8::192.168.100.254 set interfaces irb.200 family inet6 address 2001:db8::192.168.200.1/96 virtual-gateway-address 2001:db8::192.168.200.254</pre>	<pre>set interfaces irb.100 family inet6 address 2001:db8::192.168.100.2/96 virtual-gateway-address 2001:db8::192.168.100.254 set interfaces irb.200 family inet6 address 2001:db8::192.168.200.2/96 virtual-gateway-address 2001:db8::192.168.200.254</pre>	<pre>set interfaces irb.100 family inet6 address 2001:db8::192.168.100.3/96 virtual-gateway-address 2001:db8::192.168.100.254 set interfaces irb.200 family inet6 address 2001:db8::192.168.200.3/96 virtual-gateway-address 2001:db8::192.168.200.254</pre>

Table 15: Centrally-Routed Bridging Overlay—Required IRB Interface Configuration (Continued)

Description	Layer 3 VXLAN Gateway 1	Layer 3 VXLAN Gateway 2	Layer 3 VXLAN Gateway 3
Configure the gateway to send router advertisement packets only for the link local virtual-gateway-address.	set protocols router-advertisement interface irb.100 virtual-router-only set protocols router-advertisement interface irb.200 virtual-router-only	set protocols router-advertisement interface irb.100 virtual-router-only set protocols router-advertisement interface irb.200 virtual-router-only	set protocols router-advertisement interface irb.100 virtual-router-only set protocols router-advertisement interface irb.200 virtual-router-only

Edge-Routed Bridging Overlay—How to Set Up the Routing of IPv6 Data Traffic

NOTE: The focus of this section is the configuration of IRB interfaces on Layer 3 VXLAN gateways. For a more comprehensive example of configuring EVPN-VXLAN, see ["Example: Configuring an EVPN-VXLAN Edge-Routed Bridging Overlay Within a Data Center" on page 617](#).

The key to setting up this feature is the configuration of IRB interfaces on Layer 3 VXLAN gateways. In general, you configure the IRB interfaces as you would with only IPv4 hosts in the topology. However, in addition to specifying IPv4 addresses for the IRB interface, you also specify IPv6 addresses. You must also configure a MAC address for the IRB interface.

[Table 16 on page 449](#) shows how to configure the addresses for IRB interfaces irb.100 and irb.200 on the three Layer 3 VXLAN gateways. [Table 17 on page 452](#) outlines some additional required global IRB interface configuration.

Table 16: Edge-Routed Bridging Overlay—Sample IRB Interface Addresses

Addresses	Layer 2 and 3 VXLAN Gateway 1	Layer 2 and 3 VXLAN Gateway 2	Layer 2 and 3 VXLAN Gateway 3	Description
IRB interface irb.100				
IPv4 address	192.168.100.1/24	192.168.100.1/24	192.168.100.1/24	Specify the same IPv4 address for irb.100 on each device.

Table 16: Edge-Routed Bridging Overlay—Sample IRB Interface Addresses (Continued)

Addresses	Layer 2 and 3 VXLAN Gateway 1	Layer 2 and 3 VXLAN Gateway 2	Layer 2 and 3 VXLAN Gateway 3	Description
Global IPv6 address	2001:db8::192.168.100.1/96	2001:db8::192.168.100.1/96	2001:db8::192.168.100.1/96	Specify the same IPv6 address for irb.100 on each device.
Link-local IPv6 address	fe80::100:00:01/96	fe80::100:00:01/96	fe80::100:00:01/96	Specify the same link-local IPv6 address for irb.100 on each device. Any packet destined to this IPv6 address is intercepted for NDP processing.
IRB link-local IPv6 address (DHCPv6 or SLAAC)	fe80::100:00:11/96	fe80::100:00:12/96	fe80::100:00:13/96	Specify a different link local address on each device when using DHCPV6 or SLAAC in the EVPN Fabric. You must also configure the router advertisement for this IRB.
IRB MAC address	10:00:00:00:00:fe	10:00:00:00:00:fe	10:00:00:00:00:fe	Specify the same MAC address for irb.100 on each device.

IRB interface irb.200

IPv4 address	192.168.200.1/24	192.168.200.1/24	192.168.200.1/24	Specify the same IPv4 address for irb.200 on each device.
--------------	------------------	------------------	------------------	---

Table 16: Edge-Routed Bridging Overlay—Sample IRB Interface Addresses (Continued)

Addresses	Layer 2 and 3 VXLAN Gateway 1	Layer 2 and 3 VXLAN Gateway 2	Layer 2 and 3 VXLAN Gateway 3	Description
Global IPv6 address	2001:db8::192.168. 200.1/96	2001:db8::192.168. 200.1/96	2001:db8::192.168. 200.1/96	Specify the same IPv6 address for irb.200 on each device.
Link-local IPv6 address	fe80::200:00:01/96	fe80::200:00:01/96	fe80::200:00:01/96	Specify the same link-local IPv6 address for irb.200 on each device. Any packet destined to this IPv6 address is intercepted for NDP processing.
IRB link-local IPv6 address (DHCPv6 or SLAAC)	fe80::200:00:11/96	fe80::200:00:12/96	fe80::200:00:13/96	Specify a different link local address on each device when using DHCPV6 or SLAAC in the EVPN Fabric. You must also configure the router advertisement for this IRB..
IRB MAC address	10:00:00:00:00:ff	10:00:00:00:00:ff	10:00:00:00:00:ff	Specify the same MAC address for irb.200 on each device.

Table 17: Edge-Routed Bridging Overlay—Required IRB Interface Configuration

Description	Layer 2 and 3 VXLAN Gateway 1	Layer 2 and 3 VXLAN Gateway 2	Layer 2 and 3 VXLAN Gateway 3
Global IRB interface configuration			
For IPv4 fabric 2, a default Layer 3 gateway is not configured. Therefore, you must disable the advertisement of a default Layer 3 gateway.	set protocols evpn default-gateway do-not-advertise	set protocols evpn default-gateway do-not-advertise	set protocols evpn default-gateway do-not-advertise
DHCPv6 or SLAAC configuration			
Configure the linked local address.	set interfaces irb.100 family inet6 address fe80::100:00:11/96 virtual-gateway-address fe80::100:00:254 set interfaces irb.200 family inet6 address fe80::200:00:11/96 virtual-gateway-address fe80::200:00:254	set interfaces irb.100 family inet6 address fe80::100:00:12/96 virtual-gateway-address fe80::100:00:254 set interfaces irb.200 family inet6 address fe80::200:00:12/96 virtual-gateway-address fe80::200:00:254	set interfaces irb.100 family inet6 address fe80::100:00:13/96 virtual-gateway-address fe80::100:00:254 set interfaces irb.200 family inet6 address fe80::200:00:13/96 virtual-gateway-address fe80::200:00:254
Configure the global IPv6 address.	set interfaces irb.100 family inet6 address 2001:db8::192.168.100.1/96 set interfaces irb.200 family inet6 address 2001:db8::192.168.200.1/9	set interfaces irb.100 family inet6 address 2001:db8::192.168.100.1/96 set interfaces irb.200 family inet6 address 2001:db8::192.168.200.1/9	set interfaces irb.100 family inet6 address 2001:db8::192.168.100.1/96 set interfaces irb.200 family inet6 address 2001:db8::192.168.200.1/9

Table 17: Edge-Routed Bridging Overlay—Required IRB Interface Configuration *(Continued)*

Description	Layer 2 and 3 VXLAN Gateway 1	Layer 2 and 3 VXLAN Gateway 2	Layer 2 and 3 VXLAN Gateway 3
Configure the gateway to send router advertisement packets only for the link local virtual-gateway-address.	<pre>set protocols router- advertisement interface irb.100 virtual-router-only</pre> <pre>set protocols router- advertisement interface irb.200 virtual-router-only</pre>	<pre>set protocols router- advertisement interface irb.100 virtual-router-only</pre> <pre>set protocols router- advertisement interface irb.200 virtual-router-only</pre>	<pre>set protocols router- advertisement interface irb.100 virtual-router-only</pre> <pre>set protocols router- advertisement interface irb.200 virtual-router-only</pre>

Release History Table

Release	Description
15.1X53-D30	Starting with Junos OS Release 15.1X53-D30 for QFX10000 switches and Junos OS Release 18.4R1 for QFX5110 switches, the IRB interfaces on these Layer 3 VXLAN gateways can route Layer 2 or 3 data packets from one IPv6 host to another IPv6 host through an EVPN-VXLAN network with an IPv4 underlay.

RELATED DOCUMENTATION

| [Using a Default Layer 3 Gateway to Route Traffic in an EVPN-VXLAN Overlay Network](#) | 508

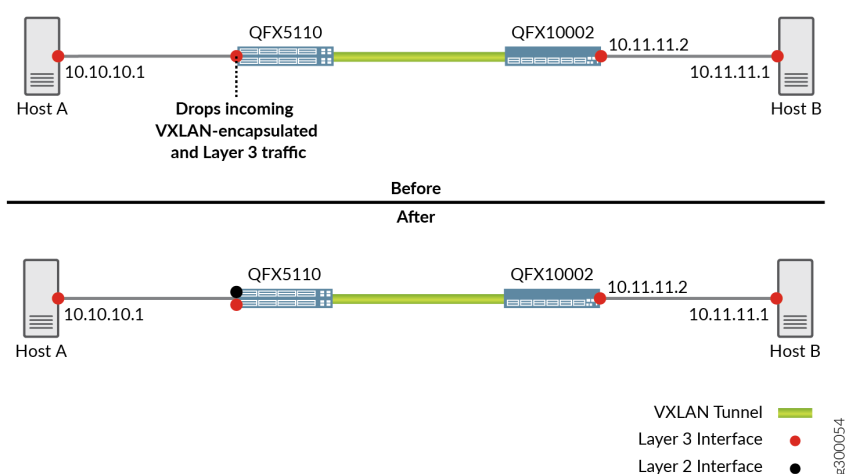
Understanding How to Configure VXLANs and Layer 3 Logical Interfaces to Interoperate

NOTE: The use case and additional configuration described in this topic apply only to QFX5110 switches.

In the sample EVPN-VXLAN network segments shown in [Figure 35 on page 454](#) (Before), hosts A and B need to exchange traffic. When host A sends a packet to host B or vice versa, the packet must traverse the following networking entities:

- On QFX5110 switch: a pure Layer 3 logical interface configured using the `set interfaces interface-name unit logical-unit-number family inet address ip-address/prefix-length` or the `set interfaces interface-name unit logical-unit-number family inet6 address ipv6-address/prefix-length` command.
- A VXLAN tunnel between the QFX5110 switch and the QFX10002 switch.
- On QFX10002 switch: a pure Layer 3 logical interface configured as described in the first bullet.

Figure 35: Results When Routing Traffic Between a VXLAN and a Layer 3 Logical Interface Is Disabled (Before) and Enabled (After)



By default, routing traffic between a VXLAN and a Layer 3 logical interface is disabled. When this functionality is disabled, the pure Layer 3 logical interface on the QFX5110 switch drops Layer 3 traffic from host A and VXLAN-encapsulated traffic from the QFX10002 switch. To prevent the pure Layer 3 logical interface on the QFX5110 switch from dropping this traffic, you can perform some additional configuration on the switch.

The additional configuration on the QFX5110 switch entails the following on a physical interface ([Figure 35 on page 454](#) (After)):

- Reconfiguring the pure Layer 3 logical interface as a Layer 2 logical interface and associating this interface with a dummy VLAN and a dummy VXLAN network identifier (VNI).
- Creating an IRB interface, which provides Layer 3 functionality within the dummy VLAN.

For example:

```
set interfaces irb unit 3500 family inet address 10.10.10.2/30
set interfaces xe-0/0/13 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/13 unit 0 family ethernet-switching vlan members VLAN-3500
```

```
set vlans VLAN-3500 vlan-id 3500
set vlans VLAN-3500 l3-interface irb.3500
set vlans VLAN-3500 vxlan vni 16770000
```

In lieu of the original pure Layer 3 logical interface, the newly created Layer 2-Layer 3 logical interfaces can now handle Layer 3 traffic from host A and VXLAN-encapsulated traffic from the QFX10002 switch.

Dynamic Load Balancing in an EVPN-VXLAN Network

IN THIS SECTION

- [Benefits of Dynamic Load Balancing in an EVPN-VXLAN Network | 455](#)
- [How Dynamic Load Balancing Works | 456](#)
- [How Traffic Is Balanced | 457](#)
- [How to Verify That Dynamic Load Balancing Is Enabled | 458](#)

When your EVPN-VXLAN network includes a multihomed device that can be reached through multiple VTEPs that share a common Ethernet segment identifier (ESI), dynamic load balancing works as follows:

- The EVPN control plane (overlay) identifies the common ESI as the next hop for a destination device with a particular MAC address.
- Based on the parameters in a packet, the forwarding plane in a Juniper Networks switch (hardware) dynamically chooses one of the VTEPs associated with the ESI. The VTEP then forwards the packet along the selected underlay path to the destination device.

By default, Juniper Networks switches have dynamic load balancing enabled. So, you don't need to configure the feature to get it up and running in an EVPN-VXLAN network.

Instead of statically assigning one virtual tunnel endpoint (VTEP) to forward traffic to a destination device in an EVPN-VXLAN network, Juniper Networks switches now support dynamic load balancing.

Benefits of Dynamic Load Balancing in an EVPN-VXLAN Network

- More efficient use of aggregated Ethernet links that share a common ESI.
- Better bandwidth utilization throughout an EVPN-VXLAN network.

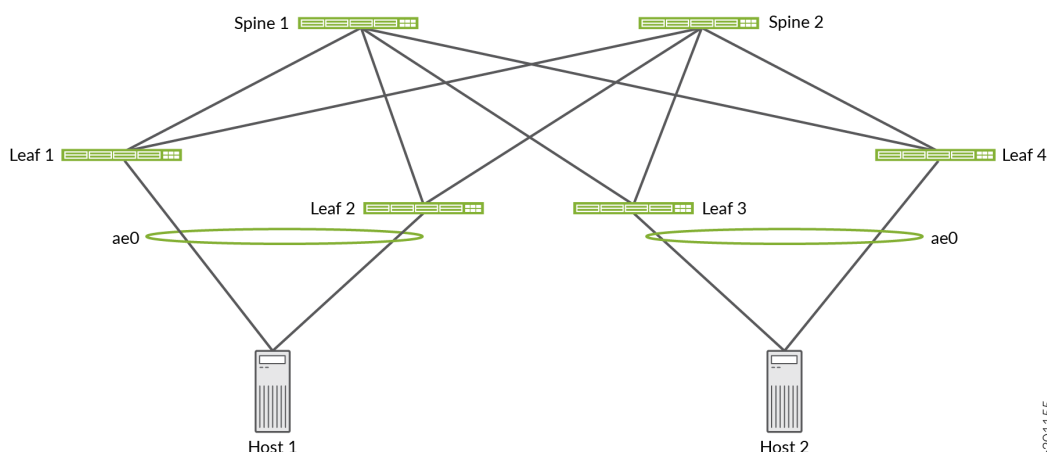
How Dynamic Load Balancing Works

Figure 36 on page 456 shows a sample EVPN-VXLAN network in which we support dynamic load balancing. This network includes the following elements:

- Multihomed Hosts 1 and 2. Each of the hosts is connected to two leaf devices through an aggregated Ethernet LAG that is assigned a common ESI.
- Multihomed Leafs 1 through 4. Each of the leaf devices is connected to Spines 1 and 2.

NOTE: To keep things simple, the sample EVPN-VXLAN network in Figure 36 on page 456 shows that the leaf devices are multihomed to two spine devices. However, we support dynamic load balancing among more than two spine devices.

Figure 36: EVPN-VXLAN Network with Dynamic Load Balancing

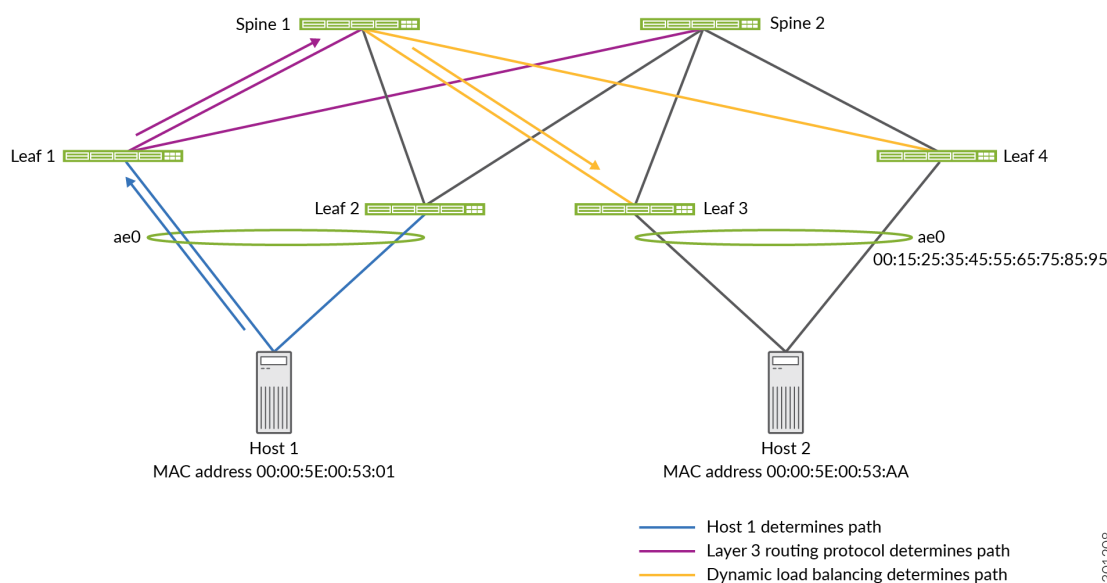


In this EVPN-VXLAN network, the leaf devices perform dynamic load balancing. To understand how dynamic load balancing works, here's what happens when Host 1 sends a packet to Host 2. In addition to the following dynamic load balancing explanation, Figure 37 on page 457 provides a graphical summary of the path options and the choices made.

- Host 1 must choose one of the aggregated Ethernet interfaces through which to forward the packet. In this case, Host 1 chooses the interface to Leaf 1.
- Upon receipt of the packet, Leaf 1 identifies Host 2's destination MAC address 00:00:5E:00:53:AA as a member of remote ESI 00:15:25:35:45:55:65:75:85:95. This ESI is assigned to aggregated Ethernet interface ae0, to which Leafs 3 and 4 are connected.

- Leaf 1 can choose either Leaf 3 or 4 as the intermediate Layer 2 EVPN-VXLAN next hop to which to forward the packet. Using packet parameters established by the dynamic load balancing feature, Leaf 1 dynamically chooses Leaf 3.
- Leaf 1 can choose either Spine 1 or 2 as the next hop to reach Leaf 3. Using Layer 3 routing tables and routes programmed into the switch hardware, Leaf 1 chooses Spine 1.

Figure 37: Summary of Path Options and Choices



How Traffic Is Balanced

Juniper Networks switches use a hash of the following packet parameters to dynamically select the next-hop VTEP:

- Packets with an IP header:
 - IP header fields:
 - Source IP address
 - Destination IP address
 - Protocol
 - VLAN ID
 - Layer 4 (TCP and UDP) source and destination ports

- Packets with an MPLS/IP header:
 - Up to three top labels
 - IP header fields:
 - Source IP address
 - Destination IP address
 - Layer 4 (TCP and UDP) source and destination ports
- Packets with a Layer 2 header only:
 - Source MAC address
 - Destination MAC address
 - VLAN ID

The hashing takes place before a packet undergoes VXLAN encapsulation.

To refine the hashing input used by dynamic load balancing, you can include the [enhanced-hash-key hash-parameters](#) `ecmp` configuration statements at the `[edit forwarding-options]` hierarchy level.

How to Verify That Dynamic Load Balancing Is Enabled

You can verify that dynamic load balancing is enabled by entering the following command:

```
user@switch> show ethernet-switching global-information
Global Configuration:

MAC aging interval      : 300
...
LE  VLAN aging time    : 1200
RE state                : Master
VXLAN Overlay load bal: Enabled
VXLAN ECMP              : Enabled
```

In the output that appears, check the `VXLAN Overlay load bal` field to make sure that it is set to `Enabled`.

Release History Table

Release	Description
20.3R1	Starting with Junos OS Release 20.3R1, QFX5100, QFX5110, QFX5120, QFX5200, QFX5210, and QFX5220 switches support dynamic load balancing in an EVPN-VXLAN network.

RELATED DOCUMENTATION

- [show ethernet-switching table | 1773](#)
- [show ethernet-switching vxlan-tunnel-end-point svlnh | 1811](#)

MAC-VRF Routing Instance Type Overview

IN THIS SECTION

- [Benefits of a MAC-VRF Routing Instance Type | 459](#)
- [MAC-VRF Enables Customer Specific VRF Tables | 460](#)
- [MAC-VRF Enables Common EVPN-VXLAN Configuration across Platforms | 460](#)
- [MAC-VRF Conforms to RFC 7432 | 463](#)
- [Usage and Behavior Notes | 463](#)

Use a MAC VRF routing instance type to configure multiple customer-specific EVPN instances (EVIs), each of which can support a different EVPN service type. With this configuration, you create customer-specific virtual routing and forwarding (VRF) tables. These tables have MAC addresses on each Juniper Networks device that serves as a virtual tunnel endpoint (VTEP) in an EVPN-VXLAN network. You use MAC-VRF routing instances for EVPN unicast routes only.

Benefits of a MAC-VRF Routing Instance Type

- Customer-specific VRF tables
- Consistent configuration across supported router and switch platforms within the EVPN-VXLAN network

- Configuration alignment with [RFC 7432](#)

MAC-VRF Enables Customer Specific VRF Tables

When you configure a MAC-VRF routing instance, you can isolate routing and forwarding traffic by customer. In fact, you can isolate the MAC-VRF instances around multiple schemes, including department, division, geographic location, etc. The isolation capabilities allow you to design and to implement traffic isolation in any way you want. The traffic within any one MAC-VRF instance cannot interact with traffic from any other MAC-VRF instances.

MAC-VRF Enables Common EVPN-VXLAN Configuration across Platforms

MX Series routers, QFX Series switches, and the EX9200 line of switches continue to support EVPN-VXLAN configurations. However, the configuration methods vary from platform to platform:

On MX Series routers, you must configure:

- `bridge-domains`
- `routing-instances`

On QFX Series switches, you must configure:

- `ethernet-switching`
- `routing-instances`

This disparity can lead to confusion and error when configuring EVPN-VXLAN across multiple platforms.

We introduced the `mac-vrf` routing instance type in Junos OS Release 20.4R1. You can use `mac-vrf` to create a common EVPN-VXLAN configuration on all supported platforms. This common configuration hierarchy also adheres to [RFC 7432](#).

NOTE: The existence of `mac-vrf` routing instances on any supported platform does not deprecate the previous methods of configuration. In fact, both methods of configurations can coexist in an active configuration.

Some command options within the `mac-vrf` hierarchy apply to only one specific platform. For example, `show mac-vrf mac-table age` (and the corresponding `show bridge mac-table age` command) applies only to MX Series routers. If you issue the `show mac-vrf mac-table age` command on a QFX Series switch, the output is blank and doesn't show an error.

See [Table 18 on page 461](#) and [Table 19 on page 462](#) for references to the existing commands.

Table 18: List of MAC-VRF Forwarding Commands by Platform

MAC-VRF Command	MX Series Routers and the EX9200 Line of Switches	QFX Series Switches
<code>show mac-vrf forwarding flood</code>	<code>show bridge flood</code>	<code>show ethernet-switching flood</code>
<code>show mac-vrf forwarding flood-group</code>	<code>show l2-learning flood-group</code>	<code>show ethernet-switching flood-group</code>
<code>show mac-vrf forwarding global-information</code>	<code>show l2-learning global-information</code>	<code>show ethernet-switching global-information</code>
<code>show mac-vrf forwarding global-mac-count</code>	<code>show l2-learning global-mac-count</code>	<code>show ethernet-switching global-mac-count</code>
<code>show mac-vrf forwarding global-mac-ip-count</code>	<code>show l2-learning global-mac-ip-count</code>	<code>show ethernet-switching global-mac-ip-count</code>
<code>show mac-vrf forwarding instance</code>	<code>show l2-learning instance</code>	<code>show ethernet-switching instance</code>
<code>show mac-vrf forwarding instance-mapping</code>	–	–
<code>show mac-vrf forwarding interface</code>	<code>show l2-learning interface</code>	<code>show ethernet-switching interface</code>
<code>show mac-vrf forwarding mac-ip-table</code>	<code>show bridge mac-ip-table</code>	<code>show ethernet-switching mac-ip-table</code>
<code>show mac-vrf forwarding mac-table</code>	<code>show bridge mac-table</code>	<code>show ethernet-switching table</code>
<code>show mac-vrf forwarding mgrp-policy</code>	<code>show l2-learning mgrp-policy</code>	<code>show ethernet-switching mgrp-policy</code>
<code>show mac-vrf forwarding statistics</code>	<code>show bridge statistics</code> <code>show evpn statistics</code>	<code>show ethernet-switching</code>
<code>show mac-vrf forwarding vlans</code>	<code>show bridge domains</code>	<code>show ethernet-switching vlans</code>

Table 18: List of MAC-VRF Forwarding Commands by Platform *(Continued)*

MAC-VRF Command	MX Series Routers and the EX9200 Line of Switches	QFX Series Switches
<code>show mac-vrf forwarding vxlan-tunnel-endpoint esi</code>	<code>show l2-learning vxlan-tunnel-end-point esi</code>	<code>show ethernet-switching vxlan-tunnel-end-point esi</code>
<code>show mac-vrf forwarding vxlan-tunnel-endpoint remote</code>	<code>show l2-learning vxlan-tunnel-end-point remote</code>	<code>show ethernet-switching vxlan-tunnel-end-point remote</code>
<code>show mac-vrf forwarding vxlan-tunnel-endpoint svlnh</code>	<code>show l2-learning vxlan-tunnel-end-point svlnh</code>	<code>show ethernet-switching vxlan-tunnel-end-point svlnh</code>

Table 19: List of MAC-VRF Routing Commands by Platform

MAC-VRF Command	MX Series Routers and the EX9200 Line of Switches	QFX Series Switches
<code>show mac-vrf routing evpn database</code>	<code>show evpn database</code>	<code>show evpn database</code>
<code>show mac-vrf routing igmp-snooping database</code>	<code>show evpn igmp-snooping database</code>	<code>show evpn igmp-snooping</code>
<code>show mac-vrf routing instance</code>	<code>show evpn instance</code>	<code>show evpn instance</code>
<code>show mac-vrf routing mld-snooping database</code>	<code>show evpn mld-snooping database</code>	<code>show evpn mld-snooping</code>
<code>show mac-vrf routing multicast-snooping status</code>	<code>show evpn multicast-snooping status</code>	<code>show evpn multicast-snooping</code>
<code>show mac-vrf routing p2mp</code>	<code>show evpn p2mp</code>	<code>show evpn p2mp</code>

NOTE: We have integrated the syntax of the commands from the `show mac-vrf routing` hierarchy into the existing `show evpn` command documentation. The links in [Table 19 on page 462](#) lead to the existing `show evpn` commands.

MAC-VRF Conforms to RFC 7432

The common configuration hierarchy across routing and switching platforms brings our implementation of MAC-VRF into compliance with <https://datatracker.ietf.org/doc/html/rfc7432>. RFC compliance allows our MAC-VRF implementation to work well in differentiated services code point (DSCP) and public cloud environments.

Usage and Behavior Notes

Read the following notes to know more about using MAC-VRF routing instances and the observed behaviors of those MAC-VRF routing instances:

- VLANs

Each supported platform has its own support framework for VLANs. Thus, the platforms vary in the number of supported VLANs and how the VLANs can overlap.

- QFX5000 Line of Switches

The QFX5000 line of switches supports only one forwarding instance. Therefore, you cannot configure overlapping VLANs within a single MAC-VRF routing instance or across multiple MAC-VRF routing instances.

- QFX10000 Line of Switches

The QFX10000 line of switches supports multiple forwarding instances. Therefore, you can configure overlapping VLANs across multiple MAC-VRF routing instances if you've mapped each routing instance to a unique forwarding instance. You cannot configure overlapping VLANs within a single MAC-VRF routing instance or across routing instances each of which maps to the same forwarding instance.

- MX Series Routers and the EX9200 Line of Switches

You can configure overlapping VLANs across multiple MAC-VRF routing instances. You cannot configure overlapping VLANs within a single MAC-VRF routing instance.

- Extended VNI List Behavior

You can configure extended virtual network identifier (VNI) lists within any MAC-VRF routing instance at the edit routing-instances *mac-vrf routing instance name* protocols evpn hierarchy level. The *extended-vni-list* keyword is an optional configuration element. By default, the device extends all VNI (whether in a VNI list or not) within the MAC-VRF routing instance. If you configure a specific VNI list, then you can extend only those VNIs that are in the list.

- The forwarding-instance Keyword

On the QFX10000 line of switches, we've included the *forwarding-instance* keyword at the edit routing-instances *mac-vrf routing-instance name* hierarchy level. Use this configuration to specify which one of the 100 available forwarding instances is used within the specified MAC-VRF routing instance. You can configure multiple MAC-VRF routing instances to use one forwarding instance. If you do not

configure the forwarding instance, the MAC-VRF routing instances use the default forwarding instance (default-switch).

On MX Series routers and the EX9200 line of switches, each MAC-VRF routing instance that you configure automatically maps to its own forwarding instance. No forwarding-instance keyword exists on these platforms.

RELATED DOCUMENTATION

[show mac-vrf forwarding mac-table | 1951](#)

[show mac-vrf forwarding vxlan-tunnel-end-point remote | 1973](#)

[show evpn mac-ip-table | 1866](#)

Configuring EVPN-VXLAN Interfaces

IN THIS CHAPTER

- [Understanding Flexible Ethernet Services Support With EVPN-VXLAN | 465](#)
- [MAC Filtering, Storm Control, and Port Mirroring Support in an EVPN-VXLAN Environment | 468](#)
- [Example: Micro and Macro Segmentation using Group Based Policy in a VXLAN | 486](#)
- [DHCP Smart Relay in EVPN-VXLAN | 495](#)

Understanding Flexible Ethernet Services Support With EVPN-VXLAN

IN THIS SECTION

- [Sample Configuration 1—Layer 2 VXLANs and Layer 3 IPv4 Logical Interfaces on Same Physical Interface | 466](#)
- [Sample Configuration 2—Layer 2 VLAN and VXLAN Logical Interfaces on Same Physical Interface | 467](#)

Flexible Ethernet services are an encapsulation type that enables a physical interface to support different types of Ethernet encapsulations at the logical interface level. Juniper Networks supports flexible Ethernet services with EVPN VXLAN.

The benefits of flexible Ethernet services support with EVPN-VXLAN include the following:

- A physical interface now supports `family ethernet-switching` and Virtual Extensible LANs (VXLANs) configured on one or more logical interfaces and `family inet` configured on other logical interfaces.
- A physical interface now supports Layer 2 VLANs configured on one or more logical interfaces and VXLANs configured on other logical interfaces.
- Instead of configuring the following Layer 2 features only on logical interface unit number 0, you can now configure them on any logical interface unit number (unit 0 and any non-zero unit number):

- Layer 2 bridging (family ethernet-switching)
- Layer 2 bridging (encapsulation vlan-bridge)
- VXLANs
- On a physical interface, you can now configure Layer 2 bridging with family ethernet-switching on one or more logical interfaces and Layer 2 bridging with encapsulation vlan-bridge on one or more logical interfaces.
- On a physical interface, you can now configure one or more logical interfaces with encapsulation extended-vlan-bridge and one or more logical interfaces with interface-mode trunk.

This topic provides the following information about configuring this feature:

Sample Configuration 1—Layer 2 VXLANs and Layer 3 IPv4 Logical Interfaces on Same Physical Interface

This sample configuration shows how to configure a logical interface for Layer 2 VXLAN forwarding and a logical interface for Layer 3 IPv4 routing on the same physical interface.

When configuring the logical interfaces, keep the following in mind:

- For this configuration to be successfully committed and work properly, you must specify the encapsulation flexible-ethernet-services configuration statements at the physical interface level—for example, set interfaces xe-0/0/5 encapsulation flexible-ethernet-services.
- You must configure Layer 2 VXLAN and Layer 3 routing on separate logical interfaces.

The following sample configuration shows physical interface et-0/0/16, on which the flexible Ethernet services encapsulation type is enabled. This physical interface is divided into logical interface 100, which is a Layer 2 (family ethernet-switching) interface that is associated with VXLANs v100 and v500, and logical interface 600, which is a Layer 3 (family inet) interface. Note that specifying the flexible Ethernet services encapsulation type on physical interface et-0/0/16 also enables you to configure the Layer 2 logical interface on a non-zero unit number, in this case, 100.

```
set interfaces et-0/0/16 flexible-vlan-tagging
set interfaces et-0/0/16 encapsulation flexible-ethernet-services
set interfaces et-0/0/16 unit 100 family ethernet-switching
interface-mode trunk
set interfaces et-0/0/16 unit 100 family ethernet-switching
vlan members v100
set interfaces et-0/0/16 unit 100 family ethernet-switching
vlan members v500
set interfaces et-0/0/16 unit 600 family inet
```

```

address 10.1.1.2/24
set vlans v100 vlan-id 100
set vlans v100 vxlan vni 100
set vlans v500 vlan-id 500
set vlans v500 vxlan vni 500

```

NOTE: This example focuses on the configuration of the physical and logical interfaces and the VXLANs with which logical interface 100 is associated. This sample configuration does not include a comprehensive configuration of EVPN and VXLAN. For a more comprehensive EVPN-VXLAN configuration, see ["Example: Configure an EVPN-VXLAN Centrally-Routed Bridging \(CRB\) Overlay" on page 530](#).

Sample Configuration 2—Layer 2 VLAN and VXLAN Logical Interfaces on Same Physical Interface

This sample configuration shows how to configure a logical interface for Layer 2 VXLAN forwarding and logical interfaces for Layer 2 VLAN forwarding on the same physical interface.

When configuring the logical interfaces for Layer 2 VLAN and Layer 2 VXLAN forwarding, you must specify the encapsulation flexible-ethernet-services configuration statements at the physical interface level—for example, set interfaces xe-0/0/5 encapsulation flexible-ethernet-services.

The following sample configuration shows physical interface xe-0/0/4, on which the flexible Ethernet services encapsulation type is enabled. This physical interface is divided into the following logical interfaces:

- Logical interface 100, which is a Layer 2 (family ethernet-switching) interface that is associated with VXLAN v100.
- Logical interface 200, which is a Layer 2 (encapsulation vlan-bridge) interface that is associated with VLAN v200
- Logical interface 300, which is a Layer 2 (encapsulation vlan-bridge) interface that is associated with VLAN v300

```

set interfaces xe-0/0/4 flexible-vlan-tagging
set interfaces xe-0/0/4 encapsulation flexible-ethernet-services
set interfaces xe-0/0/4 unit 100 family ethernet-switching
interface-mode trunk
set interfaces xe-0/0/4 unit 100 family ethernet-switching
vlan members v100

```

```

set interfaces xe-0/0/4 unit 200 encapsulation
vlan-bridge vlan members v200
set interfaces xe-0/0/4 unit 300 encapsulation
vlan-bridge vlan members v300
set vlans v100 vlan-id 100
set vlans v100 vxlan vni 100
set vlans v200 vlan-id 200
set vlans v300 vlan-id 300

```

Note that specifying the flexible Ethernet services encapsulation type on physical interface et-0/0/4 enables you to configure all Layer 2 logical interfaces on non-zero unit numbers, in this case, 100, 200, and 300. Also, you can now configure Layer 2 bridging with family ethernet-switching on one or more logical interfaces—in this case, on logical interface 100—and Layer 2 bridging with encapsulation vlan-bridge on one or more logical interfaces—in this case, logical interfaces 200 and 300.

MAC Filtering, Storm Control, and Port Mirroring Support in an EVPN-VXLAN Environment

IN THIS SECTION

- [Benefits of MAC Filtering, Storm Control, and Port Mirroring Support in an EVPN-VXLAN Environment | 469](#)
- [MAC Filtering | 469](#)
- [Storm Control | 473](#)
- [Port Mirroring and Analyzers | 474](#)
- [Configuring Remote Mirroring with GRE Encapsulation | 480](#)
- [Configuring Remote Mirroring with VXLAN Encapsulation | 482](#)
- [Remote Port Mirroring Using VNI Match Conditions | 484](#)

We support MAC filtering, storm control, and port mirroring and analyzing in an Ethernet VPN-Virtual Extensible LAN (EVPN-VXLAN) overlay network.

We support the configuration of each of these features using the Enterprise style of configuration.

We support these features only in an EVPN-VXLAN edge-routed bridging overlay (EVPN-VXLAN topology with a collapsed IP fabric). This overlay network includes the following components:

- A single layer of Juniper Networks switches—for example, QFX10002 or QFX5110 switches—each of which functions as both a Layer 3 spine device and a Layer 2 leaf device.
- Customer edge (CE) devices that are single-homed or multihomed in active/active mode to the spine-leaf devices.

This topic includes the following information:

Benefits of MAC Filtering, Storm Control, and Port Mirroring Support in an EVPN-VXLAN Environment

- MAC filtering enables you to filter and accept packets from ingress CE-facing interfaces, thereby reducing the volume of associated MAC addresses in the Ethernet switching table and traffic in a VXLAN.
- Storm control allows you to monitor traffic levels on EVPN-VXLAN interfaces, and if a specified traffic level is exceeded, drop broadcast, unknown unicast, and multicast (BUM) packets and on some Juniper Networks switches, disable the interface for a specified amount of time. This feature can prevent excessive traffic from degrading the network.
- With port mirroring and analyzers, you can analyze traffic down to the packet level in an EVPN-VXLAN environment. You can use this feature to enforce policies related to network usage and file sharing and to identify problem sources by locating abnormal or heavy bandwidth usage by particular stations or applications.

MAC Filtering

MAC filtering enables you to filter MAC addresses and accept traffic. We support this feature only on ingress CE-facing interfaces, which are interfaces on which VXLAN encapsulation is typically not enabled. To use this feature, you must do the following:

- Create a firewall filter in which you specify one or more of the supported match conditions in [Table 20 on page 469](#) and [Table 21 on page 470](#).
- Apply the firewall filter to a Layer 2 interface configured in the `[edit interfaces interface-name unit logical-unit-number family ethernet-switching filter]` hierarchy.

Table 20: Match Conditions Supported on QFX5100 and QFX5110 Switches

Match Conditions	Interface Input Filter Support	Interface Output Filter Support
Source MAC address	X	X

Table 20: Match Conditions Supported on QFX5100 and QFX5110 Switches (Continued)

Match Conditions	Interface Input Filter Support	Interface Output Filter Support
Destination MAC address	X	X
User VLAN ID	X	X
Source port	X	
Destination port	X	
Ether-type	X	
IP protocol	X	
IP precedence	X	
ICMP codes	X	
TCP flags	X	
IP address	X	

NOTE: In Junos OS Release 18.4R1, QFX5100 and QFX5110 switches support MAC filtering only on an interface. And, starting in Junos OS Release 18.4R2 and in later releases, QFX5100, QFX5110, QFX5120-48Y, and EX4650-48Y switches also support MAC filtering on a VXLAN-mapped VLAN.

Table 21: Match Conditions Supported on QFX10000 Switches

Match Conditions	Interface Input Filter Support	Interface Output Filter Support
Source MAC address	X	

Table 21: Match Conditions Supported on QFX10000 Switches (Continued)

Match Conditions	Interface Input Filter Support	Interface Output Filter Support
Destination MAC address	X	
User VLAN ID		
Source port	X	X
Destination port	X	X
Ether-type	X	X
IP protocol	X	
IP precedence	X	X
ICMP codes	X	X
TCP flags	X	X
IP address	X	X

NOTE: When configuring MAC filters on QFX10000 switches, keep the following in mind:

- You can apply a filter to an interface only. You cannot apply a filter to a VXLAN-mapped VLAN.
- We do not support a mix of Layer 2 match conditions and Layer 3/Layer 4 match conditions in the same firewall filter. For example, if you include source MAC address and source port match conditions in the same firewall filter on a QFX10002 switch, the firewall filter will not work.

- We do not support the user VLAN ID match condition. Therefore, if you need to filter logical interfaces, each of which is mapped to a particular VLAN, you must use the service provider style of configuration when configuring the physical interface and associated logical interfaces. After creating a firewall filter, you must then apply the filter to each logical interface to achieve the effect of the user VLAN ID match condition.

Through the firewall filter, you specify MAC addresses associated with a VXLAN that are allowed on a particular interface.

NOTE: After you apply the firewall filter to a Layer 2 interface, the interface resides under the default-switch instance.

The following sample configuration on a QFX5110 switch creates a firewall filter named DHCP-Discover-In that accepts and counts incoming traffic that meets multiple match conditions (source MAC address, destination MAC address, destination ports, and VLAN ID) on Layer 2 logical interface xe-0/0/6.0:

```
set firewall family ethernet-switching filter
DHCP-Discover-In term 1 from source-mac-address 00:00:5E:00:53:ab/48
set firewall family ethernet-switching filter
DHCP-Discover-In term 1 from destination-mac-address ff:ff:ff:ff:ff:ff/48
set firewall family ethernet-switching filter
DHCP-Discover-In term 1 from destination-port dhcp
set firewall family ethernet-switching filter
DHCP-Discover-In term 1 from destination-port bootps
set firewall family ethernet-switching filter
DHCP-Discover-In term 1 from destination-port bootpc
set firewall family ethernet-switching filter
DHCP-Discover-In term 1 from user-vlan-id 803
set firewall family ethernet-switching filter
DHCP-Discover-In term 1 then accept
set firewall family ethernet-switching filter
DHCP-Discover-In term 1 then count DHCP-Discover-In
set firewall family ethernet-switching filter
DHCP-Discover-In term 2 then accept
set interfaces xe-0/0/6 unit 0 family ethernet-switching
filter input DHCP-Discover-In
```

Storm Control

By default, storm control is enabled on Layer 2 interfaces that are associated with VXLANs. The storm control level is set to 80 percent of the combined BUM traffic streams.

In an EVPN-VXLAN environment, storm control is implemented and configured on Layer 2 interfaces that are associated with VXLANs the same as in a non-EVPN-VXLAN environment except for the following differences:

- In an EVPN-VXLAN environment, the traffic types that storm control monitors are as follows:
 - Layer 2 BUM traffic that originates in a VXLAN and is forwarded to interfaces within the same VXLAN.
 - Layer 3 multicast traffic that is received by an integrated routing and bridging (IRB) interface in a VXLAN and is forwarded to interfaces in another VXLAN.
- After creating a storm control profile, you must bind it to an ingress Layer 2 interface at the [edit interfaces *interface-name* unit *logical-unit-number* family ethernet-switching] hierarchy.

NOTE: After you bind the profile to a Layer 2 interface, the interface resides within the default-switch instance.

- If the traffic streams on an interface exceed the specified storm control level, the Juniper Networks switch drops the excess packets, which is known as *rate limiting*. In addition, QFX10000 switches in an EVPN-VXLAN environment support the disabling of the interface for a specified amount of time using the action-shutdown configuration statement at the [edit forwarding-options storm-control-profiles] hierarchy level and the recovery-timeout configuration statement at the [edit interfaces *interface-name* unit *logical-unit-number* family ethernet-switching] hierarchy level.

NOTE: QFX5100 and QFX5110 switches in an EVPN-VXLAN environment do not support the disabling of the interface for a specified amount of time.

NOTE: On QFX5110 switches, if you configure enhanced storm control and a native analyzer on an interface, and the native analyzer has the VxLAN VLAN as input, the shutdown action will not work for the VLAN on that interface. Rate limiting will work as expected.

The following configuration creates a profile named `scp`, which specifies that if the bandwidth used by the combined BUM traffic streams exceeds 5 percent on Layer 2 logical interface `et-0/0/23.0`, the interface drops the excess BUM traffic.

```
set forwarding-options storm-control-profiles
scp all bandwidth-percentage 5
set interfaces et-0/0/23 unit 0 family ethernet-switching
storm-control scp
```

The following configuration creates a profile named `scp`, which specifies that if the bandwidth used by the multicast traffic stream (broadcast and unknown unicast traffic streams are excluded) exceeds 5 percent on Layer 2 logical interface `et-0/0/23.0`, the interface drops the excess multicast traffic.

```
set forwarding-options storm-control-profiles
scp all bandwidth-percentage 5 no-broadcast no-unknown-unicast
set interfaces et-0/0/23 unit 0 family ethernet-switching
storm-control scp
```

The following configuration on a QFX10000 switch creates the same profile as in the previous configuration. However, instead of implicitly dropping multicast traffic if the traffic stream exceeds 5 percent, the following configuration explicitly disables the interface for 120 seconds and then brings the interface back up.

```
set forwarding-options storm-control-profiles
scp all bandwidth-percentage 5 no-broadcast no-unknown-unicast
set forwarding-options storm-control-profiles
scp all action-shutdown
set interfaces ge-0/0/0 unit 0 family ethernet-switching
storm-control scp recovery-timeout 120
```

Port Mirroring and Analyzers

To analyze traffic in an EVPN-VXLAN environment, we support the following port mirroring and analyzer functionality:

- Local mirroring
 - On an interface
 - On a VXLAN

- Remote mirroring
 - On an interface
 - On a VXLAN

The following sections provide more information about the supported functionality and include sample configurations.

Local Mirroring

NOTE: Local mirroring is also known as Switched Port Analyzer (SPAN).

Table 22: Local Mirroring Support

Entity to Which Local Mirroring Is Applied	Traffic Direction	Filter-based Support	Analyzer-based Support
CE-facing interface	Ingress	Supported. See Use Case 1: Sample Configuration.	Supported. See Use Case 2: Sample Configuration.
CE-facing interface	Egress	Not supported.	Supported; however, egress mirrored traffic might carry incorrect VLAN tags that differ from the tags in the original traffic. See Use Case 3: Sample Configuration.
IP fabric-facing interface	Ingress	Supported.	Supported. See Use Case 4: Sample Configuration.

Table 22: Local Mirroring Support (Continued)

Entity to Which Local Mirroring Is Applied	Traffic Direction	Filter-based Support	Analyzer-based Support
IP fabric-facing interface	Egress	Not supported.	Supported. However, the decision to mirror happens at ingress so the layer 2 header will not be the same as a switched or routed packet. Mirrored VXLAN-encapsulated packets will not include a VXLAN header. See Use Case 5: Sample Configuration.
VXLAN-mapped VLAN	Ingress	Supported.	Supported only for traffic entering through a CE-facing interface. See Use Case 6: Sample Configuration.

Configuring Local Mirroring

Use Case 1: Firewall filter-based

Through the use of a port mirroring instance named pm1 and a firewall filter, this configuration specifies that Layer 2 traffic that enters VXLAN100 through logical interface xe-0/0/8.0 is mirrored to an analyzer on logical interface xe-0/0/6.0 and then to port mirroring instance pm1.

```

set interfaces xe-0/0/8 unit 0 family ethernet-switching
interface-mode access
set interfaces xe-0/0/8 unit 0 family ethernet-switching
vlan members VXLAN100
set interfaces xe-0/0/8 unit 0 family ethernet-switching
filter input IPACL
set interfaces xe-0/0/6 unit 0 family ethernet-switching
set forwarding-options port-mirroring instance
pm1 family ethernet-switching output interface xe-0/0/6

```

```
set firewall family ethernet-switching filter
IPACL term to-analyzer then port-mirror-instance pm1
```

Use Case 2: Analyzer-based

Through the use of the analyzer configuration statement at the [set forwarding-options] hierarchy level, this configuration specifies that Layer 2 traffic that enters logical interface xe-0/0/8.0 is mirrored to an analyzer on logical interface xe-0/0/6.0.

```
set interfaces xe-0/0/8 unit 0 family ethernet-switching
interface-mode access
set interfaces xe-0/0/8 unit 0 family ethernet-switching
vlan members VXLAN100
set interfaces xe-0/0/6 unit 0 family ethernet-switching
set forwarding-options analyzer ANA1 input ingress
interface xe-0/0/8.0
set forwarding-options analyzer ANA1 output
interface xe-0/0/6.0
```

Use Case 3: Analyzer-based

Through the use of the analyzer configuration statement at the [set forwarding-options] hierarchy level, this configuration specifies that Layer 2 traffic that exits logical interface xe-0/0/8.0 is mirrored to an analyzer on logical interface xe-0/0/6.0.

```
set vlans VXLAN100 vlan-id 100
set interfaces xe-0/0/8 unit 0 family ethernet-switching
interface-mode trunk
set interfaces xe-0/0/8 unit 0 family ethernet-switching
vlan members VXLAN100
set interfaces xe-0/0/6 unit 0 family ethernet-switching
set forwarding-options analyzer test input egress
interface xe-0/0/8
set forwarding-options analyzer test output
interface xe-0/0/6
```

Use Case 4: Analyzer-based

Through the use of the analyzer configuration statement at the [set forwarding-options] hierarchy level, this configuration specifies that Layer 2 traffic that enters logical interface xe-0/0/29.0 is mirrored to an analyzer on logical interface xe-0/0/6.0.

```
set vlans VXLAN100 vlan-id 100
set interfaces xe-0/0/29 unit 0 family ethernet-switching
interface-mode trunk
set interfaces xe-0/0/29 unit 0 family ethernet-switching
vlan members VXLAN100
set interfaces xe-0/0/6 unit 0 family ethernet-switching
set forwarding-options analyzer test input ingress
interface xe-0/0/29
set forwarding-options analyzer test output
interface xe-0/0/6
```

Use Case 5: Analyzer-based

Through the use of the analyzer configuration statement at the [set forwarding-options] hierarchy level, this configuration specifies that Layer 2 traffic that exits logical interface xe-0/0/29.0 is mirrored to an analyzer on logical interface xe-0/0/6.0.

```
set vlans VXLAN100 vlan-id 100
set interfaces xe-0/0/29 unit 0 family ethernet-switching
interface-mode trunk
set interfaces xe-0/0/29 unit 0 family ethernet-switching
vlan members VXLAN100
set interfaces xe-0/0/6 unit 0 family ethernet-switching
set forwarding-options analyzer test input egress
interface xe-0/0/29
set forwarding-options analyzer test output
interface xe-0/0/6
```

Use Case 6: Analyzer-based

Through the use of the analyzer configuration statement at the [set forwarding-options] hierarchy level, this configuration specifies that Layer 2 traffic that enters the VLAN named VXLAN100 and is mirrored to an analyzer on logical interface xe-0/0/6.0.

```
set vlans VXLAN100 vlan-id 100
set interfaces xe-0/0/8 unit 0 family ethernet-switching
interface-mode trunk
```

```

set interfaces xe-0/0/8 unit 0 family ethernet-switching
vlan members VXLAN100
set interfaces xe-0/0/6 unit 0 family ethernet-switching
set forwarding-options analyzer test input ingress
vlan VXLAN100
set forwarding-options analyzer test output
interface xe-0/0/6

```

Remote Mirroring

Remote port mirroring is used when the output destination is not on the same switch as the source. Remote mirroring delivers the mirrored traffic to one or more remote destination hosts. It is often used in the data center environment for troubleshooting or monitoring.

In an EVPN-VXLAN environment, the mirrored traffic flow at the source switch is encapsulated and tunneled through the underlay IP fabric to the destination host IP address. We support the following types of encapsulation:

- Generic routing encapsulation (GRE) is used with remote mirroring to encapsulate the traffic between switches separated by a routing domain. In an EVPN-VXLAN overlay, GRE encapsulation enables mirroring between leaf devices over the IP fabric. Use remote mirroring with GRE when the mirroring destination hosts are connected to a switch that is part of the same fabric as the source switch. Remote mirroring with GRE encapsulation is comparable to encapsulated remote SPAN (ERSPAN).
- VXLAN encapsulation supports remote mirroring for EVPN-VXLAN when the source and the output destinations are in separate VNI domains. You must configure a specific VXLAN for mirroring traffic for the output destination interfaces and mapped to a VNI. Remote mirroring with VXLAN encapsulation is comparable to remote SPAN (RSPAN).

Table 23: Remote Mirroring Support

Entity to Which Remote Mirroring Is Applied	Traffic Direction	Filter-based Support	Analyzer-based Support
CE-facing interface	Ingress	Supported.	Supported. See Use Case 1: Sample Configuration.

Table 23: Remote Mirroring Support (Continued)

Entity to Which Remote Mirroring Is Applied	Traffic Direction	Filter-based Support	Analyzer-based Support
CE-facing interface	Egress	Not supported.	Supported. See Use Case 2: Sample Configuration.
IP fabric-facing interface	Ingress	Supported.	Supported. See Use Case 3: Sample Configuration.
IP fabric-facing interface	Egress	Not supported.	Supported. However, the decision to mirror happens at ingress so the layer 2 header will not be the same as a switched or routed packet. NOTE: Mirrored traffic might include a bogus VLAN ID tag of 4094 on the native MAC frame. See Use Case 4: Sample Configuration.
VXLAN-mapped VLAN	Ingress	Supported.	Supported only for traffic entering a CE-facing interface. See Use Case 5: Sample Configuration.

Configuring Remote Mirroring with GRE Encapsulation

The following sample configurations are for analyzer-based remote mirroring with GRE encapsulation.

Use Case 1

Through the use of the analyzer configuration statement at the [set forwarding-options] hierarchy level, this configuration specifies that Layer 2 traffic that enters logical interface xe-0/0/8.0 is mirrored to a remote logical interface with an IP address of 10.9.9.2.

```
set vlans VXLAN100 vlan-id 100
set interfaces xe-0/0/8 unit 0 family ethernet-switching
interface-mode access
set interfaces xe-0/0/8 unit 0 family ethernet-switching
vlan members VXLAN100
```

```

set forwarding-options analyzer test input ingress
interface xe-0/0/8.0
set forwarding-options analyzer test output
ip-address 10.9.9.2

```

Use Case 2

Through the use of the analyzer configuration statement at the [set forwarding-options] hierarchy level, this configuration specifies that Layer 2 traffic that exits logical interface xe-0/0/8.0 is mirrored to a remote logical interface with an IP address of 10.9.9.2.

```

set vlans VXLAN100 vlan-id 100
set interfaces xe-0/0/8 unit 0 family ethernet-switching
interface-mode trunk
set interfaces xe-0/0/8 unit 0 family ethernet-switching
vlan members VXLAN100
set forwarding-options analyzer test input egress
interface xe-0/0/8
set forwarding-options analyzer test output
ip-address 10.9.9.2

```

Use Case 3

Through the use of the analyzer configuration statement at the [set forwarding-options] hierarchy level, this configuration specifies that Layer 2 traffic that enters logical interface xe-0/0/29.0 is mirrored to a remote logical interface with an IP address of 10.9.9.2.

```

set vlans VXLAN100 vlan-id 100
set interfaces xe-0/0/29 unit 0 family ethernet-switching
interface-mode trunk
set interfaces xe-0/0/29 unit 0 family ethernet-switching
vlan members VXLAN100
set forwarding-options analyzer test input ingress
interface xe-0/0/29
set forwarding-options analyzer test output
ip-address 10.9.9.2

```

Use Case 4

Through the use of the analyzer configuration statement at the [set forwarding-options] hierarchy level, this configuration specifies that Layer 2 traffic that exits logical interface xe-0/0/29.0 is mirrored to a remote logical interface with an IP address of 10.9.9.2.

```
set vlans VXLAN100 vlan-id 100
set interfaces xe-0/0/29 unit 0 family ethernet-switching
interface-mode trunk
set interfaces xe-0/0/29 unit 0 family ethernet-switching
vlan members VXLAN100
set forwarding-options analyzer test input egress
interface xe-0/0/29
set forwarding-options analyzer test output
ip-address 10.9.9.2
```

Use Case 5

Through the use of the analyzer configuration statement at the [set forwarding-options] hierarchy level, this configuration specifies that Layer 2 traffic that enters VXLAN100, which is mapped to logical interface xe-0/0/8.0, is mirrored to a remote logical interface with an IP address of 10.9.9.2.

```
set vlans VXLAN100 vlan-id 100
set interfaces xe-0/0/8 unit 0 family ethernet-switching
interface-mode trunk
set interfaces xe-0/0/8 unit 0 family ethernet-switching
vlan members VXLAN100
set forwarding-options analyzer test input ingress
vlan VXLAN100
set forwarding-options analyzer test output
ip-address 10.9.9.2
```

Configuring Remote Mirroring with VXLAN Encapsulation

Analyzer-based configuration

The following sample configuration is for analyzer-based remote mirroring with VXLAN encapsulation. Layer 2 traffic that enters VLAN100 is mirrored to remote output destination VLAN3555, which maps to VNI 1555.

This configuration uses a loopback interface on the destination VLAN to encapsulate the mirrored packets.

- Destination interface xe-0/0/2 is externally connected to loopback interface xe-0/0/3.
- Logical interfaces xe-0/0/2.0 and xe-0/0/3.0 are members of destination VLAN3555.
- Interface xe-0/0/2.0 is configured in enterprise style without any VNI mapping, while interface xe-0/0/3.0 is configured in service provider style with the same vlan ID and with the VNI mapping. This is to prevent flooding or looping between these ports.
- Mac-learning must be disabled on xe-0/0/2.0.

NOTE: The ingress interface must be configured in trunk mode, so the encapsulated packets are tagged. To decapsulate the tagged packets, configure the `set protocols l2-learning decapsulate-accept-inner-vlan` command at the decapsulation node.

```
set vlans VLAN3555 vlan-id 3555
set vlans v3555 vxlan vni 1555
set vlans v3555 interface xe-0/0/3.3555
set interfaces xe-0/0/2 unit 0 family ethernet-switching
interface-mode trunk
set interfaces xe-0/0/2 unit 0 family ethernet-switching vlan members VLAN3555
set interfaces xe-0/0/3 flexible-vlan-tagging
set interfaces xe-0/0/3 encapsulation extended-vlan-bridge
set interfaces xe-0/0/3 unit 3555 vlan-id 3555
set switch-options interface xe-0/0/2 no-mac-learning
set forwarding-options analyzer test input ingress vlan VLAN100
set forwarding-options analyzer test output vlan VLAN3555
```

To configure a logical loopback interface instead of using an external connection, use the following command:

set interfaces *interface-name* ether-options loopback

The sample configuration below uses a logical loopback on interface xe-0/0/2:

```
set vlans VLAN3555 vlan-id 3555
set vlans v3555 vxlan vni 1555
set interfaces xe-0/0/2 unit 0 family ethernet-switching
interface-mode trunk
set interfaces xe-0/0/2 unit 0 family ethernet-switching vlan members VLAN3555
set interfaces xe-0/0/2 ether-options loopback
set switch-options interface xe-0/0/2 no-mac-learning
```

```
set forwarding-options analyzer test input ingress vlan VLAN100
set forwarding-options analyzer test output vlan VLAN3555
```

Firewall filter-based configuration

The following configuration applies a firewall filter, `filter1`, to ingress traffic at interface `xe-0/0/34`. Traffic ingressing on this interface is mirrored to the destination `VLAN3555`. The destination VLAN is defined using a port mirroring instance named `pm1`.

```
set interfaces xe-0/0/34 unit 0 family ethernet-switching filter input filter1
set forwarding-options port-mirroring instance pm1 family ethernet-switching output vlan vlan3555
set firewall family ethernet-switching filter filter1 term to-analyzer then port-mirror-instance pm1
```

Remote Port Mirroring Using VNI Match Conditions

For QFX10002, QFX10008, and QFX10016 Series switches, you can use VXLAN network identifier (VNI) values as a match condition when filtering traffic for remote port mirroring. This ability is often used in network planning and for analytics such as deep packet inspection (DPI).

The remote port mirroring feature acts to create a copy of targeted ingress packets, which are encapsulated in an outer IPv4 GRE header, and then forwarded to designated remote destination. Support for VNI match conditions means you can select packets to be mirrored on the basis of their VNI so that only those flows are directed to the remote mirror port. You can also configure a differentiated service code point (DSCP) value to prioritize the flow, for example, high priority or best-effort delivery. Integrated routing and bridging (IRB) interfaces are not supported as a destination mirror port.

At a high level, the procedure for remote port mirroring based on VNIs is to create a remote port mirroring instance, promote VNI in the firewall filter family to handle VNIs, create the necessary filter rules and actions, and then apply the firewall policy to an ingress interface. GRE tunneling should also be in place on the interface used to send the mirrored packets.

Remote Port Mirroring on the Basis of VNI Use Case: Sample Configuration

The code samples below highlight the key Junos CLI configurations you need to mirror packets according to VNI.

1. Enable remote port mirroring, and also configure the traffic source and the destination to which you want to send the mirrored packets.

```
set forwarding-options port-mirroring remote-port-mirroring
set port-mirroring remote-port-mirroring instance name output ip-source-address IPv4 address
```

```
set port-mirroring remote-port-mirroring instance name output ip-destination-address IPv4 address
```

2. Create a firewall filter (here, named *bf_vni_st*), and promote VNI in this filter to be the packet forwarding module (in other words, this command sets the entire filter to optimize VNI match conditions).

```
set firewall family inet filter fbf_vni_st promote vni
```

3. Create an ingress firewall filter (*bf_vni_st*), that specifies a VNI match condition (6030 in this sample), and an action (*count*, in this sample).

```
set firewall family inet filter fbf_vni_st term t1-vni from protocol udp
set firewall family inet filter fbf_vni_st term t1-vni from vxlan vni 6030
set firewall family inet filter fbf_vni_st term t1-vni then count t1-vni-6030
set firewall family inet filter fbf_vni_st term default-term then count default-cnt
```

4. Apply the filter to ingress traffic at interface you want to mirror.

```
set interfaces interface unit number family inet filter input fbf_vni_st
```

RELATED DOCUMENTATION

[Understanding Storm Control](#)

[Enabling and Disabling Storm Control \(ELS\)](#)

[Understanding Port Mirroring and Analyzers](#)

[Port Mirroring Constraints and Limitations](#)

[VXLAN Constraints on QFX Series and EX Series Switches](#) | 423

Example: Micro and Macro Segmentation using Group Based Policy in a VXLAN

SUMMARY

VXLAN-GBP

IN THIS SECTION

- [Overview | 486](#)
- [Requirements | 491](#)
- [Configuration | 491](#)

Overview

IN THIS SECTION

- [VXLAN-GBP Supported Switches | 487](#)
- [Assigning SGTs with a RADIUS Server | 487](#)
- [Topology | 490](#)

You can achieve micro and macro segmentation, for example to secure data and assets, in a VXLAN architecture using Group Based Policy (GBP). VXLAN-GBP works by leveraging reserved fields in the VXLAN header for use as a Scalable Group Tag (SGT), which you can then use as match conditions in firewall filter rules. Using a SGT is more robust than using port or MAC addresses to achieve similar results. SGTs can be assigned statically (by configuring the switch on a per port or per MAC basis), or they can be configured on the RADIUS server and pushed to the switch via 802.1X when the user is authenticated.

The segmentation enabled by VXLAN-GBP is especially useful in campus VXLAN environments because it gives you a practical way to create network access policies that are independent of the underlying network topology. It simplifies the design and implementation phases of developing network-application and endpoint-device security policies.

To accommodate the use of SGTs on the RADIUS server, we need to leverage vendor specific attribute (VSA), as supported by the AAA service framework (these VSA are carried as part of the standard RADIUS request reply message, and provide a built-in extension to handle implementation-specific information such as our SGTs). The exact syntax on the RADIUS server varies according to whether the authentication scheme is MAC or EAP based. For MAC based clients, the configuration looks like this:

```
001094001199 Cleartext-Password := "001094001199"
Juniper-Switching-Filter = "apply action gbp-tag 100"
```

For EAP based clients, the SGT is pushed from RADIUS server at the time of authentication. The configuration looks like this:

```
PermEmp01 Auth-Type = EAP, Cleartext-Password := "gbp"
Juniper-Switching-Filter = "apply action gbp-tag 100"
```

Starting with Junos Release 21.1R1, EX4400 switches introduce a new match criteria for use with VXLAN-GBP that allows the firewall to recognize the SGT tags that get passed by the RADIUS server and inserted into the VXLAN header.

You can see how this works in the following code samples. GBP firewall policies are framed on the basis of source and destination GBP tags. A source tag is the 16-bit field in the VXLAN header in the incoming packet, while the destination tag is derived at the egress tunnel endpoint, according to the configured tag assignment.

Let's say we have an egress end point with the configuration shown below. Packets from source MAC address 00:01:02:03:04:10:10 are assigned the tag 100, and packets from source MAC address 00:01:02:03:04:20:20 are assigned 200.

```
set firewall family ethernet-switching filter assign_tag term tag100 from source-mac-address
00:01:02:03:04:10:10
set firewall family ethernet-switching filter assign_tag term tag100 then gbp-src-tag 100
set firewall family ethernet-switching filter assign_tag term tag200 from source-mac-address
00:01:02:03:04:20:20
set firewall family ethernet-switching filter assign_tag term tag200 then gbp-src-tag 200
```

For packets with GBP tag 100 and a destination MAC address of 00:01:02:03:04:10:10, the destination group tag (*gbp-dst-tag*) will be 100, and it will match on term *t10-100*. Likewise, for packets with GBP

tag 100 and a destination MAC address of 00:01:02:03:04:20:20, the destination group tag will be 200, and it will match term *t10-200*.

```
set firewall family ethernet-switching filter gbp-policy term t10-100 from gbp-src-tag 100
set firewall family ethernet-switching filter gbp-policy term t10-100 from gbp-dst-tag 100
set firewall family ethernet-switching filter gbp-policy term t10-100 then accept
set firewall family ethernet-switching filter gbp-policy term t10-200 from gbp-src-tag 100
set firewall family ethernet-switching filter gbp-policy term t10-200 from gbp-dst-tag 200
set firewall family ethernet-switching filter gbp-policy term t10-200 then discard
```

The same tag assignment used to map the source MAC address to the source tag is also used to map the destination MAC address to the destination tag. This is true for port based assignments as well.

Let's look at another code sample, this time using a GBP source tag of 300, and with packets ingressing interface *ge-0/0/30.0*. As you can see below, GBP source tag 300 is assigned and in egress direction, and 300 is also GBP destination group tag.

```
set firewall family ethernet-switching filter assign_tag term tag300 from interface ge-0/0/30.0
set firewall family ethernet-switching filter assign_tag term tag300 then gbp-src-tag 300
```

Note that you need to configure the GBP firewall filter on the egress switch, because there is no way for the ingress switch to know what group tags are used at the egress switch. In addition, you must enable VXLAN-GBP globally on the ingress node, so it can perform the look-up on the matches and add SGT in the VXLAN header, and also on the egress node. Do this with the configuration command shown here:

```
set chassis forwarding-options vxlan-gbp-profile
```

Before creating any rules, it can be helpful to organize your scheme by creating a table for all your endpoints (users and devices) and the assigned SGT value. Here, we show one such table, the values of which will later be applied in a matrix, that can be used to further simplify the logic and clarify your rules.

Table 25: Endpoints and Their SGT Values

Endpoint	Assigned SGT value
Permanent Employee (PE)	100
Contractor (CON)	200

Table 25: Endpoints and Their SGT Values (Continued)

Endpoint	Assigned SGT value
Security Staff (SS)	300
Security Cam (CAM)	400
Engineering Server (ES)	500

The relationship between the RADIUS server and SGTs, the EX4400 and VXLAN packet headers, and a central firewall filter to manage the access policy, is such that a matrix becomes a handy way to organize the values. In the following table, we list user roles down the first column and device types across the first row to create an access matrix. Each user role and device type is assigned a SGT and the RADIUS configuration has been updated with the information.

This example uses three types of employees, Permanent Employee (PE), Contractor (CON), and Security Staff (SS). It also uses two types of resources, Eng Server (ES) and security camera (CAM). We use **Y** to indicate access is permitted, and **N** to shown when access is blocked. The table serves as a useful resource when creating the various firewall rules in the policy and makes access mapping simple and clear.

Table 26: Access Matrix

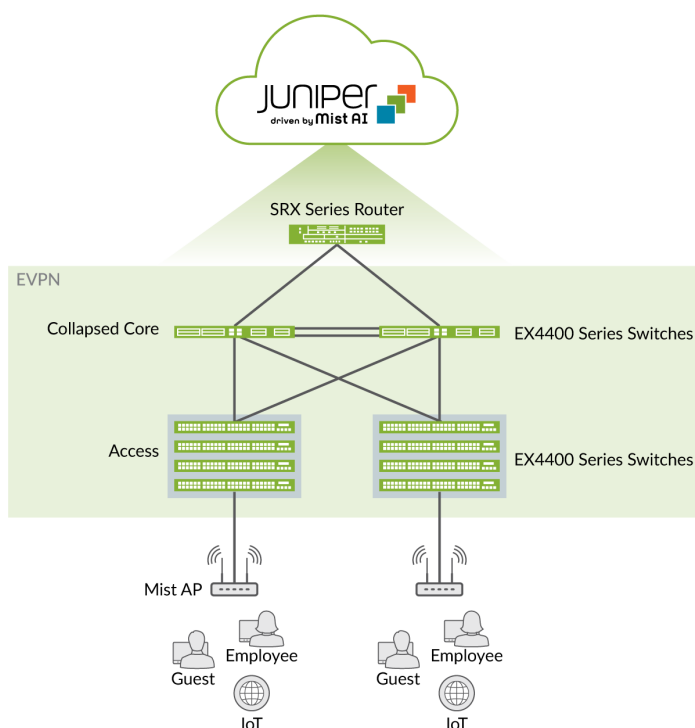
	ES (SGT 500)	CAM (SGT 400)	PE (SGT 100)	CON (SGT 200)	SS (SGT 300)
PE (SGT 100)	Y	N	Y	Y	N
CON (SGT 200)	N	N	Y	N	N
SS (SGT 300)	N	Y	N	N	Y

Topology

For the sake of simplicity, all the configuration in this example is done on a single Juniper EX4400 series switch running Junos OS Release 21.1R1. The switch is connected to a RADIUS server for AAA. This

switch functions as egress in this example. Recall that for SGTs you must define the firewall on the egress switch, whereas you would typically do it on the ingress VXLAN gateway for the access layer.

Figure 39: VXLAN GBP on a EX4400 switch



Requirements

VXLAN-GBP is supported in Junos OS Release 21.1R1 on the following switches: EX4400-24P, EX4400-24T, EX4400-48F, EX4400-48P, and EX4400-48T. Let us consider an EX4400 switch in this example.

Starting with Junos Release 21.4R1, VXLAN-GBP is supported on the following switches as well: QFX5120-32C, QFX5120-48T, QFX5120-48Y, QFX5120-48YM, EX4650, and EX4650-48Y-VC.

Configuration

IN THIS SECTION

- [Configuring a Stand-Alone Juniper EX4400 Switch for VXLAN-GBP | 492](#)

We can summarize the sequence of events underlying VXLAN-GBP based segmentation, laid out in the paragraphs above, as follows:

- Users log on to the network and are authenticated by the RADIUS server (on which SGTs are configured for all the endpoints).
- Using firewall filters, the EX4400 selects traffic on the basis of the 802.1X authentication or MAC address, and then assigns a group tag to matching frames. (for dot1x authenticated clients, the static firewall configuration is not needed). The mechanics of this are performed using firewall policy, as shown here:

```
set firewall family ethernet-switching filter name term name from source-mac-address MAC-Addr
```

and

```
set firewall family ethernet-switching filter name term name then gbp-src-tag PE-GRP
```

- Tagged traffic passing through the EX4400 is evaluated on the basis SGT values, again, using the mechanics of the firewall filter. For this to happen, you first need to enable chassis forwarding-options vxlan-gbp-profile on the switch, then you use the gbp-dst-tag and/or gbp-src-tag match conditions to write your firewall rules, and include them in the routing policy on the egress switch you use for GBP microsegmentation.

Configuring a Stand-Alone Juniper EX4400 Switch for VXLAN-GBP

Use the following commands to configure VXLAN-GBP segmentation in a sandbox environment. Typically, you would create the firewall filter rules on the switch that serves as the (egress) VXLAN gateway for the access layer, but for the sake of simplicity, we're using the same stand-alone EX4400 for both the firewall filter rules and the RADIUS server (EAP, here). The values we use in this example are taken from the previous tables.

The commands below include variables such as profile names and IP addresses, which must be adapted to make sense for your test environment.

1. Configure the radius server:

```
set groups dot1xgbp access radius-server 10.204.96.102 port 1812
set groups dot1xgbp access radius-server 10.204.96.102 secret "secret key"
set groups dot1xgbp access profile radius_profile_dev12 authentication-order radius
set groups dot1xgbp access profile radius_profile_dev12 radius authentication-server
10.204.96.102
```

```
set groups dot1xgbp access profile radius_profile_dev12 radius accounting-server 10.204.96.102
set groups dot1xgbp access profile radius_profile_dev12 accounting order radius
```

2. Configure the physical ports to support RADIUS authentication:

```
set groups dot1xgbp protocols dot1x authenticator authentication-profile-name
radius_profile_dev12
set groups dot1xgbp protocols dot1x authenticator interface xe-0/0/46.0 supplicant multiple
set groups dot1xgbp protocols dot1x authenticator interface xe-0/0/46.0 mac-radius
```

3. Set up the SGT tags on the RADIUS server:

```
Contractor01 Auth-Type = EAP, Cleartext-Password := "gbp"
Juniper-Switching-Filter = "apply action gbp-tag 100"
Contractor01 Auth-Type = EAP, Cleartext-Password := "gbp"
Juniper-Switching-Filter = "apply action gbp-tag 200"
SecurityStaff01 Auth-Type = EAP, Cleartext-Password := "gbp"
Juniper-Switching-Filter = "apply action gbp-tag 300"
SecurityCam01 Auth-Type = EAP, Cleartext-Password := "gbp"
Juniper-Switching-Filter = "apply action gbp-tag 400"
EngServer01 Auth-Type = EAP, Cleartext-Password := "gbp"
Juniper-Switching-Filter = "apply action gbp-tag 500"
```

4. Enable VXLAN-GBP on the switch:

```
set chassis forwarding-options vxlan-gbp-profile
```

5. Create Firewall filter rules that leverage the SGTs (using values organized in the matrix):

```
set groups gbp-policy firewall family ethernet-switching filter gbp-policy term pe-to-pe from
gbp-src-tag 100
set groups gbp-policy firewall family ethernet-switching filter gbp-policy term pe-to-pe from
gbp-dst-tag 100
set groups gbp-policy firewall family ethernet-switching filter gbp-policy term pe-to-pe then
accept
set groups gbp-policy firewall family ethernet-switching filter gbp-policy term pe-to-pe then
count PE-PE

set groups gbp-policy firewall family ethernet-switching filter gbp-policy term pe-to-es from
```



```

gbp-src-tag 100
set groups gbp-policy firewall family ethernet-switching filter gbp-policy term pe-to-es from
gbp-dst-tag 500
set groups gbp-policy firewall family ethernet-switching filter gbp-policy term pe-to-es then
accept
set groups gbp-policy firewall family ethernet-switching filter gbp-policy term pe-to-es then
count PE-ES

set groups gbp-policy firewall family ethernet-switching filter gbp-policy term pe-to-cam
from gbp-src-tag 100
set groups gbp-policy firewall family ethernet-switching filter gbp-policy term pe-to-cam
from gbp-dst-tag 400
set groups gbp-policy firewall family ethernet-switching filter gbp-policy term pe-to-cam
then discard
set groups gbp-policy firewall family ethernet-switching filter gbp-policy term pe-to-cam
then count PE-CAM

set groups gbp-policy firewall family ethernet-switching filter gbp-policy term con-to-cam
from gbp-src-tag 200
set groups gbp-policy firewall family ethernet-switching filter gbp-policy term con-to-cam
from gbp-dst-tag 400
set groups gbp-policy firewall family ethernet-switching filter gbp-policy term con-to-cam
then discard
set groups gbp-policy firewall family ethernet-switching filter gbp-policy term con-to-cam
then count CON-CAM

set groups gbp-policy firewall family ethernet-switching filter gbp-policy term con-to-es
from gbp-src-tag 200
set groups gbp-policy firewall family ethernet-switching filter gbp-policy term con-to-es
from gbp-dst-tag 500
set groups gbp-policy firewall family ethernet-switching filter gbp-policy term con-to-es
then discard
set groups gbp-policy firewall family ethernet-switching filter gbp-policy term con-to-es
then count CON-ES

set groups gbp-policy firewall family ethernet-switching filter gbp-policy term ss-to-cam
from gbp-src-tag 300
set groups gbp-policy firewall family ethernet-switching filter gbp-policy term ss-to-cam
from gbp-dst-tag 400
set groups gbp-policy firewall family ethernet-switching filter gbp-policy term ss-to-cam
then accept
set groups gbp-policy firewall family ethernet-switching filter gbp-policy term ss-to-cam
then count SS-CAM

```

```

set groups gbp-policy firewall family ethernet-switching filter gbp-policy term ss-to-es from
gbp-src-tag 300
set groups gbp-policy firewall family ethernet-switching filter gbp-policy term ss-to-es from
gbp-dst-tag 500
set groups gbp-policy firewall family ethernet-switching filter gbp-policy term ss-to-es then
discard
set groups gbp-policy firewall family ethernet-switching filter gbp-policy term ss-to-es then
count SS-ES

set apply-groups gbp-policy

```

6. Run a commit check in Junos to verify that the commands and the variables you used are valid. When satisfied with your configuration commit the candidate configuration to make it active on the device. These commands are shown below. You can also review your configuration by typing `run show configuration`.

```

commit check
configuration check succeeds
commit
commit complete

```

RELATED DOCUMENTATION

| [vxlan-gbp-profile](#) | 1716

DHCP Smart Relay in EVPN-VXLAN

DHCP smart relay provides redundancy and resiliency to DHCP relay by allowing the relay agent to use multiple IP addresses as the gateway IP address. When forwarding DHCP client requests to a DHCP server, the relay agent initially uses the primary IP address as the gateway IP address. If the DHCP server does not respond with an offer message after three attempts, the relay agent repeats the process using an alternate IP address as the gateway IP address.

To use DHCP smart relay, you must have at least two IP addresses configured on the relay agent interface. If you configure more than two IP addresses on the relay agent interface, smart relay tries each IP address until an offer message is received.

DHCP smart relay is supported for EVPN over VXLAN in centrally routed bridging and edge-routed bridging modes. EVPN is usually configured with DHCP stateless relay to support multihoming, but DHCP smart relay is supported only for stateful relay. To support multihoming in an EVPN over VXLAN deployment with DHCP stateful relay, use the following configuration:

```
set forwarding-options dhcp-relay forward-snooped-clients all-interfaces
set forwarding-options dhcp-relay overrides allow-snooped-clients
set forwarding-options dhcp-relay route-suppression access-internal
```

To enable smart relay on all interfaces that are relay agents, configure the following statement:

```
set forwarding-options dhcp-relay overrides apply-secondary-as-giaddr
```

To enable smart relay on specific interfaces that are relay agents, configure the following statement:

```
set forwarding-options dhcp-relay group group-name interface interface-name overrides apply-
secondary-as-giaddr
```

Configuring VLAN-Aware Bundle Services, VLAN-Based Services, and Virtual Switch Support

IN THIS CHAPTER

- [Understanding VLAN-Aware Bundle and VLAN-Based Service for EVPN | 497](#)
- [Configuring EVPN with VLAN-Based Service | 498](#)
- [Virtual Switch Support for EVPN Overview | 504](#)
- [Configuring EVPN with Support for Virtual Switch | 505](#)

Understanding VLAN-Aware Bundle and VLAN-Based Service for EVPN

IN THIS SECTION

- [VLAN-Aware Bundle Service | 498](#)
- [VLAN-Based Service | 498](#)

A Data Center Service Provider (DCSP) hosts the data center for its multiple customers on a common physical network. To each customer (also called a tenant), the service looks like a full-fledged data center that can expand to 4094 VLANs and all private subnets. For disaster recovery, high availability, and optimization of resource utilization, it is common for the DCSP to span across the data center to more than one site. When deploying the data center services, a DCSP faces the following main challenges:

- Extending Layer 2 domains across more than one data center site. This requires optimal intra-subnet traffic forwarding.
- Supporting optimal inter-subnet traffic forwarding and optimal routing in the event of virtual machine (VM).
- Supporting multiple tenants with independent VLAN and subnet space.

The DCSP might require Ethernet VLAN services to be extended over a WAN with a single EVPN instance (EVI). On the QFX Series, each EVPN and virtual switch routing instance corresponds to an EVI. The QFX Series supports VLAN-aware bundle service and VLAN-based service, which maintain data and control plane separation.

NOTE: If you create VLANs that are not part of a routing instance, they become part of the Default Switch routing instance.

VLAN-Aware Bundle Service

VLAN-aware bundle services supports the mapping of one or more routing instances of type Virtual Switch to many VLAN IDs (VIDs) and multiple bridge tables, with each bridge table corresponding to a different VLAN. To enable VLAN-aware bundle service, configure a Virtual Switch routing instance. For service provider-related applications, where the VLAN ID is local to the Layer 2 logic interface, enable the `flexible-vlan-tagging` statement in your configuration. For enterprise-related applications, where the VLAN ID has global significance, enable the `family ethernet-switching` statement in your configuration. VLAN-aware bundle service supports up to 4000 VLANs per routing instance.

VLAN-Based Service

VLAN-based service supports the mapping of one routing instance of type EVPN to one VLAN. There is only one bridge table that corresponds to the one VLAN. If the VLAN consists of multiple VLAN IDs (VIDs)—for example, there is a different VID per Ethernet segment on a provider edge device—then VLAN translation is required for packets that are destined to the Ethernet segment. To enable VLAN-based service, configure an EVPN routing instance. Up to 100 EVPN routing instances are supported.

Configuring EVPN with VLAN-Based Service

VLAN-based service supports the mapping of one or more routing instances of type EVPN to only one VLAN. There is only one bridge table that corresponds to the one VLAN. If the VLAN consists of multiple VLAN IDs (VIDs)—for example, there is a different VID per Ethernet segment on a provider edge device—then VLAN translation is required for packets that are destined to the Ethernet segment.

To configure VLAN-based service and Layer 3 routing with two EVPN routing instances on a provider edge device.

1. Configure the first routing instance of type `evpn` named `evpn1`.

For example:

```
[edit]
user@switch# set routing-instances evpn1 instance-type
evpn
```

2. Configure the access interface for handling EVPN traffic.

For example:

```
[edit]
user@switch# set routing-instances evpn1 interface
xe-0/0/8.100
```

3. Configure a Layer 3 integrated and routing (IRB) interface for the evpn1 routing instance.

For example:

```
[edit]
user@switch# set routing-instances evpn1 l3-interface
irb.100
```

4. Configure the Virtual Tunnel Endpoint interface for the evpn1 routing instance.

For example:

```
[edit]
user@switch# set routing-instances evpn1 vtep-source-interface
lo0.0
```

5. Configure a VLAN identifier for the evpn1 routing instance.

For example:

```
[edit]
user@switch# set routing-instances evpn1 vlan-id
none
```

6. Configure a route distinguisher for the evpn1 routing instance.

For example:

```
[edit]
user@switch# routing-instances evpn1 route-distinguisher
1.2.3.11:1
```

7. Configure the VPN routing and forwarding (VRF) target community for the evpn1 routing instance.

For example:

```
[edit]
user@switch# set routing-instances evpn1 vrf-target
target:1234:11
```

8. Configure the encapsulation type for the evpn1 routing instance.

For example:

```
[edit]
user@switch# set routing-instances evpn1 protocols
evpn encapsulation vxlan
```

9. Configure the VXLAN Network Identifier (VNI) for the evpn1 routing instance.

For example:

```
[edit]
user@switch# set routing-instances evpn1 vxlan
vni 100
```

10. Configure the second routing instance of type evpn named evpn2:

For example:

```
[edit]
user@switch# set routing-instances evpn2 instance-type
evpn
```

11. Configure the access interface on the provider edge device (PE) for handling EVPN traffic.

For example:

```
[edit]
user@switch# set routing-instances evpn2 interface
xe-0/0/8.200
```

12. Configure a Layer 3 integrated and routing (IRB) interface for the evpn2 routing instance.

For example:

```
[edit]
user@switch# set routing-instances evpn2 l3-interface
irb.200
```

13. Configure the loopback address as the virtual tunnel endpoint source interface for the evpn2 routing instance.

For example:

```
[edit]
user@switch# set routing-instances evpn2 vtep-source-interface
lo0.0
```

14. Configure the VLAN identifier for the evpn2 routing instance to be none.

For example:

```
[edit]
user@switch# set routing-instances evpn2 vlan-id
none
```

15. Configure a route distinguisher for the evpn2 routing instance.

For example:

```
[edit]
user@switch# set routing-instances evpn2 route-distinguisher
1.2.3.11:2
```

16. Configure the VPN routing and forwarding (VRF) target community for the evpn2 routing instance.

For example:

```
[edit]
user@switch# set routing-instances evpn2 vrf-target
target:1234:24
```

17. Configure the encapsulation type for the evpn2 routing instance.

For example:

```
[edit]
user@switch# set routing-instances evpn2 protocols
evpn encapsulation vxlan
```

18. Configure the VXLAN Network Identifier (VNI) for the evpn2 routing instance.

For example:

```
[edit]
user@switch# set routing-instances evpn2 vxlan
vni 200
```

19. Configure a VPN routing and forwarding (VRF) routing instance.

For example:

```
[edit]
user@switch# set routing-instances vrf instance-type
vrf
```

20. Configure the first of two integrated routing and bridging (IRB) interface for the vrf instance.

For example:

```
[edit]
user@switch# set routing-instances vrf interface
irb.100
```

21. Configure the second of two integrated routing and bridging (IRB) interface for the VPN routing and forwarding (VRF) instance.

For example:

```
[edit]
user@switch# set routing-instances vrf interface
  irb.200
```

22. Configure the loopback interface for the vrf instance.

For example:

```
[edit]
user@switch# set routing-instances vrf interface
  lo0.1000
```

23. Configure a unique route distinguisher for the VPN routing and forwarding (VRF) instance to identify from which EVPN the route belongs.

For example:

```
[edit]
user@switch# set routing-instances vrf route-distinguisher
  1.2.3.1:2
```

24. Configure the VPN routing and forwarding (VRF) target community for the VPN routing and forwarding (VRF) routing instance.

For example:

```
[edit]
user@switch# set routing-instances vrf vrf-target
  target:2222:22
```

25. Configure the auto-export option to automatically derive the route target.

For example:

```
[edit]
user@switch# set routing-instances vrf routing-options
  auto-export
```

Virtual Switch Support for EVPN Overview

Starting with Junos OS Release 14.1, VLAN-aware bundle service is introduced on MX Series routers. Starting with Junos OS Release 14.2, VLAN-aware bundle service is introduced on EX9200 switches. Starting with Junos OS Release 17.3, VLAN-aware bundle service is introduced on QFX Series switches. This feature allows Ethernet VLANs over a WAN to share a single EVPN instance while maintaining data-plane separation between the different VLANs.

Junos OS has a highly flexible and scalable virtual switch interface. With a virtual switch, a single router or switch can be divided into multiple logical switches. Layer 2 domains (also called *bridge-domains* or *vlan*s) can be defined independently in each virtual switch. To configure VLAN-aware bundle service, an EVPN must run in a virtual-switch routing instance.

On the EX Series and MX Series, a single EVPN instance can stretch up to 4094 bridge domains or VLANs defined in a virtual switch to remote sites. A virtual switch can have more than 4094 bridge domains or VLANs with a combination of none, single, and dual VLANs. However, because EVPN signaling deals only with single VLAN tags, a maximum of 4094 bridge domains or VLANs can be stretched. The EVPN virtual switch also provides support for trunk and access interfaces.

Starting with Junos OS Release 17.3R1, on the QFX10000 line of switches, you can configure both EVPN and virtual-switch routing instances. The EVPN routing instance supports VLAN-based service. With VLAN-based service, the EVPN instance includes only a single broadcast domain, and there is a one-to-one mapping between a VNI and MAC-VRF. Up to 100 EVPN routing instances are supported. The virtual-switch routing instance supports VLAN-aware service, and up to 10 virtual-switch routing instances with 2000 VLANs are supported.

If you create VLANs that are not part of a routing instance, they become part of the default switch routing instance.

NOTE:

- The `none` VLAN option is supported with bridge domains or VLANs under the virtual switch instance type for EVPNs.
- Access interfaces configured with single or dual VLAN tags are supported in EVPN. By default, only Ethernet frames with single or no VLAN tags are transported across the EVPN core. As a result, dual-tagged Ethernet frames received on the access interfaces must be normalized to Ethernet frames with single or no VLAN tags for proper transmission over the EVPN core.

You can enable transporting of dual-tagged frames across the EVPN core network by including both the `vlan-id none` and `no-normalization` configuration statements together.

There are two types of VLAN-aware bundle service:

- VLAN-aware bundle without translation

The service interface provides bundling of customer VLANs into a single Layer 2 VPN service instance with a guarantee end-to-end customer VLAN transparency. The data-plane separation between the customer VLANs is maintained by creating a dedicated bridge-domain for each VLAN.

- VLAN-aware bundle with translation

The service interface provides bundling of customer VLANs into a single Layer 2 VPN service instance. The data-plane separation between the customer VLANs is maintained by creating a dedicated bridge-domain for each VLAN. The service interface supports customer VLAN translation to handle the scenario where different VLAN Identifiers (VIDs) are used on different interfaces to designate the same customer VLAN.

EVPN with virtual switch provides support for VLAN-aware bundle with translation only.

Release History Table

Release	Description
17.3R1	Starting with Junos OS Release 17.3R1, on the QFX10000 line of switches, you can configure both EVPN and virtual-switch routing instances.
17.3	Starting with Junos OS Release 17.3, VLAN-aware bundle service is introduced on QFX Series switches.
14.2	Starting with Junos OS Release 14.2, VLAN-aware bundle service is introduced on EX9200 switches.
14.1	Starting with Junos OS Release 14.1, VLAN-aware bundle service is introduced on MX Series routers.

RELATED DOCUMENTATION

| [Example: Configuring EVPN with Support for Virtual Switch](#) | 929

Configuring EVPN with Support for Virtual Switch

You can configure an Ethernet VPN (EVPN) with virtual switch support to enable multiple tenants with independent VLAN and subnet space within an EVPN instance. Virtual switch provides the ability to extend Ethernet VLANs over a WAN using a single EVPN instance while maintaining data-plane separation between the various VLANs associated with that instance. A single EVPN instance can stretch up to 4094 bridge domains defined in a virtual switch to remote sites.

When configuring virtual switch for EVPN, be aware of the following considerations:

- Due to default ARP policing, some of the ARP packets not destined for the device can be missed. This can lead to delayed ARP learning and synchronization.
- Clearing ARP for an EVPN can lead to inconsistency between the ARP table and the EVPN ARP table. To avoid this situation, clear both ARP and EVPN ARP tables.
- The `vlan-tag` can be configured for local switching. However, vlan-tagged VLANs should not be extended over the EVPN cloud.

This task explains how to configure one Virtual Switch instance that includes one VLAN.

1. Configure the virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance instance-type virtual-switch
```

2. Configure the interface names for the virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance interface interface-name
```

3. Configure the route distinguisher for the virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance route-distinguisher route-distinguisher-value
```

4. Configure the VPN routing and forwarding (VRF) target community for the virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance vrf-target vrf-target
```

5. List the VLAN identifiers that are to be EVPN extended.

```
[edit routing-instances]
user@PE1# set evpn-instance protocols evpn extended-vni-list [vlan-id-range]
```

6. Configure the VLAN and VLAN ID for the virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance vlans name of VLAN vlan-id VLAN ID number
```

7. Configure VXLAN encapsulation and Virtual Network Identifier for the virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance vxlan vni VNI number
```

8. Configure the virtual tunnel endpoint source interface for the virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance vtep-source-interface interface-name
```

9. Verify and commit the configuration.

```
[edit]
user@PE1# commit
commit complete
```

RELATED DOCUMENTATION

| [Example: Configuring EVPN with Support for Virtual Switch](#) | 929

Setting Up a Layer 3 VXLAN Gateway

IN THIS CHAPTER

- [Using a Default Layer 3 Gateway to Route Traffic in an EVPN-VXLAN Overlay Network | 508](#)
- [Using a RIOT Loopback Port to Route Traffic in an EVPN-VXLAN Network | 514](#)
- [Understanding the MAC Addresses For a Default Virtual Gateway in an EVPN-VXLAN Overlay Network | 524](#)
- [Supported Protocols on an IRB Interface in EVPN-VXLAN | 527](#)
- [Example: Configure an EVPN-VXLAN Centrally-Routed Bridging \(CRB\) Overlay | 530](#)
- [Example: Configuring a QFX5110 Switch as a Layer 3 VXLAN Gateway in an EVPN-VXLAN Centrally-Routed Bridging Overlay | 566](#)

Using a Default Layer 3 Gateway to Route Traffic in an EVPN-VXLAN Overlay Network

IN THIS SECTION

- [Understanding the Default Gateway | 509](#)
- [Understanding the Redundant Default Gateway | 512](#)
- [Understanding Dynamic ARP Processing | 513](#)

Physical (bare-metal) servers in an Ethernet VPN-Virtual Extensible LAN (EVPN-VXLAN) environment rely on a default Layer 3 gateway to route their traffic from one virtual network (VN) to another physical server or a virtual machine (VM) in another VN. You can enable the default gateway functionality on a Juniper Networks device that acts as a Layer 3 VXLAN gateway. On a Layer 3 VXLAN gateway, you can configure an integrated routing and bridging (IRB) interface with a virtual gateway address (VGA), which in turn configures the IRB interface as a default Layer 3 gateway. You can configure an IRB interface

with a VGA when using EVPN-VXLAN within a data center and across the Data Center Interconnect (DCI) solution.

Understanding the Default Gateway

To enable the default gateway function, you configure an IRB interface with a unique IP address and a media access control (MAC) address. In addition, you configure the IRB interface with a VGA, which must be an anycast IP address, and the Layer 3 VXLAN gateway automatically generates a MAC address.

When you specify an IPv4 address for the VGA, the Layer 3 VXLAN gateway automatically generates 00:00:5e:00:01:01 as the MAC address. When you specify an IPv6 address, the Layer 3 VXLAN gateway automatically generates 00:00:5e:00:02:01 as the MAC address.

On Juniper Networks devices that function as Layer 3 VXLAN gateways, you can explicitly configure an IPv4 or IPv6 MAC address for a default gateway by using the `virtual-gateway-v4-mac` or `virtual-gateway-v6-mac` configuration statement at the `[edit interfaces irb unit logical-unit-number]` hierarchy level. With this configuration, the device overrides the automatically generated MAC address with the configured MAC address.

BEST PRACTICE: On QFX5xxx, QFX100xx, and EX4xxx lines of switches that support EVPN-VXLAN, we recommend that you configure the same `virtual-gateway-v4-mac` address for each IRB unit rather than configuring a unique one for each IRB. Similarly, configure the same `virtual-gateway-v6-mac` address for each IPv6 IRB unit rather than a unique one for each.

A VGA and associated MAC address provide the default gateway function in a particular VN. You configure each host (physical server or VM) in the VN to use the VGA.

By using an anycast IP address as the VGA, when a VM is moved from one EVPN provider edge (PE) device to another in the same VN, the VM can use the same default gateway. In other words, you do not need to update the VM with a new default gateway IP address for MAC binding.

Layer 3 VXLAN gateways in an EVPN-VXLAN topology respond to Address Resolution Protocol (ARP) requests for the VGA and forward packets intended for the default gateway MAC address.

BEST PRACTICE: On Juniper Networks devices that function as Layer 3 VXLAN gateways in an EVPN-VXLAN centrally-routed bridging overlay (EVPN-VXLAN topology with a two-layer IP fabric), we recommend that the IRB interface IP address is unique across the different Layer 3 VXLAN gateways for a given virtual network, and you configure a virtual gateway MAC address for the IRB. Following this recommendation avoids an asymmetric data path for the ARP request and response when the IRB interface sends ARP messages intended for an end-destination's

MAC address. If you configure a virtual gateway MAC address for the IRB interface, we recommend you use a unique MAC address across the different Layer 3 VXLAN gateways, and in a given Layer 3 VXLAN gateway, use the same MAC address across different IRB units.

For IRB interfaces configured on QFX10000 switches in an EVPN-VXLAN edge-routed bridging overlay (EVPN-VXLAN topology with a two-layer IP fabric), you can alternatively configure each IRB interface on each Layer 3 VXLAN gateway in a VN with the same MAC address. For more information, see ["Example: Configuring an EVPN-VXLAN Edge-Routed Bridging Overlay Within a Data Center" on page 617](#).

BEST PRACTICE: We recommend that you disable the automatic ESI generation for EVPN networks with edge-routed bridging. To disable automatic ESI generation, include the `no-auto-virtual-gateway-esi` statement at the `[edit interfaces irb unit logical-unit-number]` hierarchy level.

NOTE: To troubleshoot an IRB interface, you can ping the IP address of the interface.

To troubleshoot a default gateway on an MX Series router, you can ping the VGA of the default gateway from a CE device. To support pinging of the VGA, include the `virtual-gateway-accept-data` statement at the `[edit interfaces irb unit]` hierarchy of the preferred virtual gateway.

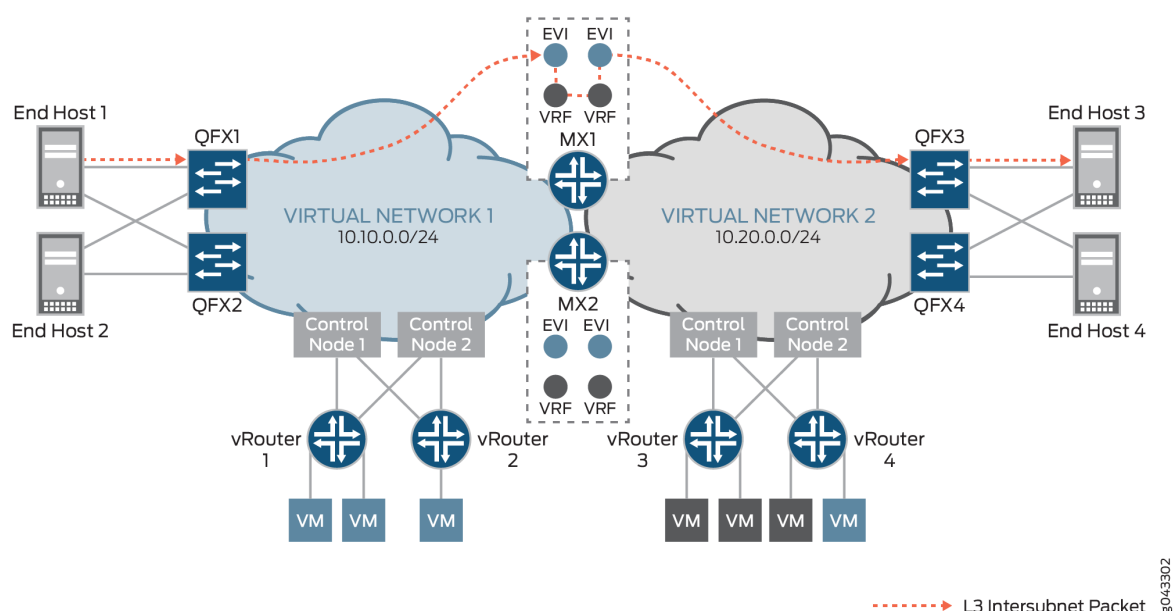
Additionally, you can ping the IP address of the CE device from the PE device (MX Series router). To support pinging of the IP address of the CE device, include the `preferred` statement at `[edit interfaces irb unit logical-unit-number family {inet | inet6} address ip-address]` hierarchy using the unique IRB IP address. Otherwise, you must manually specify the unique IRB IP address as the source IP address when you ping the CE device.

For each IRB interface with a VGA configured, there are two sets of IP and MAC addresses—one set for the IRB interface itself and one set for the default gateway. As a result, MAC routes for both IRB interface and default gateway are advertised. However, no default gateway extended community attribute is associated with the MAC route advertisement for the default gateway because all Layer 3 VXLAN gateways have the same anycast IP address and MAC binding.

Understanding How a Default Gateway Handles Known Unicast Traffic Between Virtual Networks

In the centrally-routed bridging overlay shown in [Figure 40 on page 511](#), MX Series routers function as Layer 3 VXLAN gateways and QFX5200 switches function as Layer 2 VXLAN gateways. End hosts 1 through 4 are physical servers that must communicate with each other.

Figure 40: Handling Known Unicast Traffic Between Virtual Networks



In this topology, end host 1 in VN1 (10.10.0.0/24) and end host 3 in VN2 (10.20.0.0/24) exchange known unicast packets. Before the exchange of packets between the two end hosts, assume that the hosts sent ARP requests to MX1, which is a Layer 3 VXLAN gateway, and that MX1 responded with the MAC address of a default gateway in VN1.

For example, end host 1 originates a packet and sends it to QFX1, which is a Layer 2 VXLAN gateway. QFX1 encapsulates the packet with a VXLAN header and sends it to MX1. For the inner destination MAC, the packet includes the MAC address of a default gateway in VN1. For the inner destination IP, the packet includes the IP address of end host 3. Upon receipt of the packet, MX1 de-encapsulates it, and after detecting the MAC address of the default gateway in the inner destination MAC field, performs a route lookup for end host 3's IP address in the L3-VRF routing table for VN1. After a route is found, the packet is routed to VN2 and based on the ARP route entry, the packet is encapsulated with a VXLAN header and sent to QFX3. QFX3 de-encapsulates the packet, and sends it to end host 3.

NOTE: The traffic flow and handling of known unicast traffic in an edge-routed bridging overlay are essentially the same as described in this section. The only difference is that in the edge-routed bridging overlay, a QFX Series switch that supports Layer 3 VXLAN gateway functionality acts as both Layer 2 and Layer 3 VXLAN gateways.

Understanding How a Default Gateway Handles Unknown Unicast Traffic Between Virtual Networks

NOTE: The information in this section applies to the traffic flow and handling of unknown unicast packets in both centrally-routed and edge-routed bridging overlays.

For unknown unicast traffic between VNs that is initiated by a physical server, an additional ARP request and response process is required at each stage. After the destination MAC addresses for both default gateway and host is resolved, the traffic flows in the same way as described in ["Understanding How a Default Gateway Handles Known Unicast Traffic Between Virtual Networks" on page 511](#).

Understanding the Redundant Default Gateway

The Juniper Networks devices that function as Layer 3 VXLAN gateways can also provide redundant default gateway functionality. A redundant default gateway prevents the loss of communication between physical servers in one VN and physical servers or VMs in another VN.

The redundant default gateway functionality is typically achieved in an EVPN-VXLAN topology where a provider edge (PE) device such as a Layer 2 VXLAN gateway or a Contrail vRouter is multihomed in active-active mode to multiple Layer 3 VXLAN gateways. On the Layer 3 VXLAN gateways, IRB interfaces are configured as default gateways. Note that each default gateway uses the same VGA and MAC address. In addition, the VGAs and MAC addresses are associated with the same Ethernet segment ID (ESI).

The ESI associated with the VGA and MAC address of the default gateway is automatically derived from an autonomous system (AS) and the VXLAN network identifier (VNI) for the VN. As a result, the default gateway MAC routes advertised by each Layer 3 VXLAN gateway for a given VN have the same ESI.

From the perspective of a Layer 2 VXLAN gateway or a Contrail vRouter that is multihomed to the Layer 3 VXLAN gateways, the addresses of each default gateway configured on each Layer 3 VXLAN gateway is the same. As a result, the PE devices build an equal-cost multipath (ECMP) next hop to reach each default gateway. Traffic that originates from a host and is destined for the MAC address of a default gateway is load balanced.

If one of the Layer 3 VXLAN gateways fails, the remote PE devices are notified of the withdrawing or purging of the next hop to the default gateway MAC address. The path to the failed Layer 3 VXLAN gateway is removed from the next-hop database. Despite the removal of the path, the default gateway that is configured on the remaining Layer 3 VXLAN gateway is still reachable, and the ARP entries for the hosts remain unchanged.

Understanding Dynamic ARP Processing

When a physical server needs to determine the MAC address of its default gateway, the physical server initiates an ARP request that includes the VMA of the default gateway. In a centrally-routed bridging overlay, a Layer 2 VXLAN gateway typically receives the ARP request, encapsulates the request in a VXLAN header, and forwards the encapsulated packet to a Layer 3 VXLAN gateway. In an edge-routed bridging overlay, a Layer 2 and 3 VXLAN gateway typically receives the ARP request from the directly connected physical server.

Upon receipt of the ARP request, the Layer 3 VXLAN gateway de-encapsulates the packet if appropriate, learns the IP and MAC binding of the physical server, and creates an ARP entry in its database. The Layer 3 VXLAN gateway then replies with the MAC address of the default gateway.

In a centrally-routed bridging overlay, the ARP response is encapsulated with a VXLAN header and unicast back to the Layer 2 VXLAN gateway. The Layer 2 VXLAN gateway de-encapsulates the ARP response and forward the packet to the physical server.

In an edge-routed bridging overlay, the ARP response is unicast back to the directly connected physical server.

In a situation where a physical server in VN1 originates a packet that is destined for a physical server in VN2, the Layer 3 VXLAN gateway searches its database for an ARP entry for the destination physical server. If a match is not found, the Layer 3 VXLAN gateway initiates an ARP request that includes the IP and MAC addresses of the IRB interface that is mapped to VN2, and sends the request to the destination physical server. The destination physical server learns the IP/MAC binding of the IRB interface, and adds or refreshes the ARP entry in its database accordingly. The physical server then unicasts an ARP response, which includes the MAC address of the IRB interface, back to the Layer 3 VXLAN gateway,

RELATED DOCUMENTATION

[EVPN Multihoming Overview | 96](#)

[Understanding EVPN with VXLAN Data Plane Encapsulation | 398](#)

[EVPN Over VXLAN Encapsulation Configuration Overview for QFX Series and EX4600 Switches | 430](#)

[EVPN-over-VXLAN Supported Functionality | 408](#)

Using a RIOT Loopback Port to Route Traffic in an EVPN-VXLAN Network

SUMMARY

You can configure a RIOT loopback port on a device that doesn't support native VXLAN routing. With this feature, the device can serve as a Layer 3 VXLAN gateway in an EVPN-VXLAN fabric.

IN THIS SECTION

- [Loopback Port Solution for Routing in and out of VXLAN Tunnels \(RIOT\) for Layer 3 VXLAN Gateway Support | 514](#)
- [Configure a RIOT Loopback Port on a Layer 3 VXLAN Gateway Leaf Device | 518](#)

Loopback Port Solution for Routing in and out of VXLAN Tunnels (RIOT) for Layer 3 VXLAN Gateway Support

IN THIS SECTION

- [RIOT Loopback Solution Overview | 514](#)
- [How RIOT Loopback Processing Works | 516](#)
- [Asymmetric EVPN Type 2 Routes with RIOT Loopback Processing | 517](#)
- [EVPN Type 5 Routes with RIOT Loopback Processing | 517](#)
- [IRB Interface Status Dependency on RIOT Loopback Port State | 518](#)

Some Juniper Networks EVPN-VXLAN fabric devices, such as QFX5210 switches, don't have native support for routing in and out of VXLAN tunnels (RIOT). Starting in Junos OS Release 21.3R1, you can configure a loopback port on supporting devices to perform RIOT operations in a two-pass process. With this solution, you can use the device as a Layer 3 VXLAN gateway device in an EVPN-VXLAN edge-routed bridging overlay (ERB) fabric.

RIOT Loopback Solution Overview

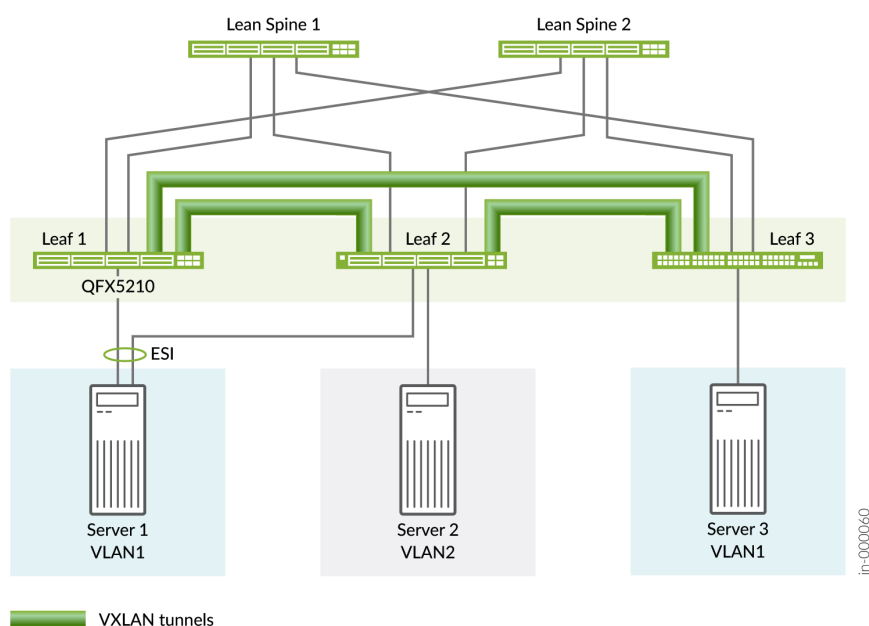
You can configure a Layer 3 VXLAN gateway using a RIOT loopback port in an EVPN-VXLAN ERB fabric with:

- MAC VRF routing instances—either VLAN-based or VLAN-aware bundle service type.

- Enterprise style interface configuration.
- EVPN asymmetric Type 2 and EVPN Type 5 routing.

The following figure shows an EVPN-VXLAN ERB fabric. The leaf devices route traffic through VXLAN tunnels to the other leaf devices in the fabric. Leaf 1 is a QFX5210 switch that doesn't support native VXLAN routing in and out of the VXLAN tunnel.

Figure 41: EVPN-VXLAN ERB Overlay Fabric with a RIOT Loopback Layer 3 Gateway Leaf Device



For Leaf 1 to serve as Layer 3 VXLAN gateway, you need to configure the RIOT loopback port solution on that device. RIOT loopback routing is transparent to the other leaf devices that connect through VXLAN tunnels in the fabric. You can include Layer 3 gateway leaf devices in the same fabric that use RIOT loopback port routing with devices that use native VXLAN routing.

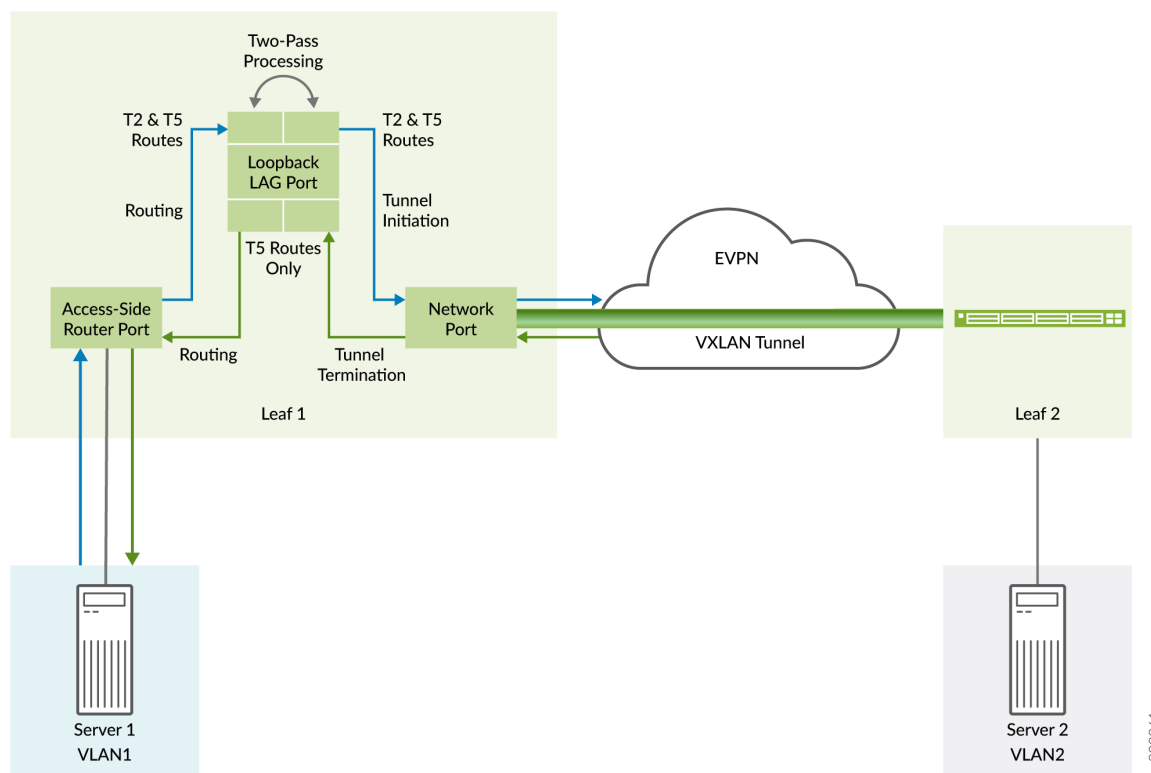
We support the RIOT loopback process for:

- Asymmetric EVPN Type 2 routing
- EVPN Type 5 routing

How RIOT Loopback Processing Works

The following figure shows how the RIOT loopback process routes traffic in or out of a VXLAN tunnel that connects to another leaf device.

Figure 42: RIOT Loopback Two-Pass Processing



You configure the RIOT loopback port as an Ethernet link aggregation group (LAG). Adjust the number of member links depending on the bandwidth of VXLAN traffic that uses the loopback path. You can use any network ports for the LAG that aren't already used for another purpose. The device automatically turns off MAC learning on the RIOT loopback LAG so the port doesn't learn its own MAC address when traffic passes through it.

The traffic flows through the RIOT loopback LAG in the first pass, then loops back into the RIOT loopback LAG for the second pass. What happens during the RIOT loopback process depends on the direction of traffic flow and the type of routes. The device uses the RIOT loopback process for:

- Access port to network port routing and VXLAN tunnel initiation with EVPN asymmetric Type 2 or EVPN Type 5 routing.
- Network port to access port VXLAN tunnel termination and routing with EVPN Type 5 routing only.

The device doesn't need to use the RIOT loopback LAG in the following cases:

- Access port to access port with asymmetric Type 2 routing.

The device routes the traffic locally through the IRB interfaces on the device as usual (no VXLAN bridging needed).

- Network port to access port with asymmetric Type 2 routing.

In this case, the ingress VTEP already routed the traffic exiting the VXLAN tunnel to the destination VLAN. The device uses normal Layer 2 VXLAN traffic processing to bridge the traffic on the destination VLAN.

We describe more about the RIOT loopback process with different EVPN route types next. Also, see ["Configure a RIOT Loopback Port on a Layer 3 VXLAN Gateway Leaf Device" on page 518](#) for details on what you need to configure for each EVPN route type.

Asymmetric EVPN Type 2 Routes with RIOT Loopback Processing

With asymmetric EVPN Type 2 routes, all VLANs extend over the EVPN network on all devices. The integrated bridging and forwarding (IRB) actions on the two VXLAN tunnel endpoints (VTEP) differ as follows:

- The ingress VTEP IRB interface routes the traffic from the source VLAN to the destination VLAN. Then the device bridges the traffic on the destination VLAN across the VXLAN tunnel.

NOTE: With asymmetric routing, you configure all destination VLANs on the ingress VTEP even if the ingress VTEP doesn't host servers on all VLANs.

- The egress VTEP receives the traffic on the destination VLAN, and then forwards the traffic on the destination VLAN. The egress VTEP doesn't need to route the traffic.

This means that devices don't need the RIOT loopback process with asymmetric Type 2 routes on VXLAN tunnel traffic coming *into* the device from the EVPN network.

EVPN Type 5 Routes with RIOT Loopback Processing

For EVPN Type 5 routes to work with the RIOT loopback process, the device uses an extra VLAN for each Type 5 virtual routing and forwarding (VRF) instance. Each extra VLAN maps to the VXLAN network identifier (VNI) for the corresponding Type 5 VRF instance. The extra VLAN enables Type 5 routes in both traffic flow directions (to and from the VXLAN tunnel) as follows:

- Access port to network port traffic:

In the first pass, the RIOT loopback process routes the traffic out of the RIOT loopback port. In the second pass for tunnel initiation, the RIOT loopback process needs the VNI for the corresponding Type 5 VRF instance. The RIOT loopback process uses the extra VLAN's VLAN-to-VNI mapping for that purpose.

- Network port to access port traffic

When terminating the VXLAN tunnel, the device needs a VLAN tag with which to send the traffic out of the RIOT loopback port. The RIOT loopback process adds the extra VLAN ID as the VLAN tag in the first pass. In the second pass, the RIOT loopback process uses the VLAN tag to find the corresponding Type 5 VRF instance to do the route lookup.

IRB Interface Status Dependency on RIOT Loopback Port State

Layer 3 VXLAN gateway devices route traffic between VLANs using IRB interfaces. On devices with RIOT loopback processing, all IRB interfaces you configure for VXLAN routing depend on the RIOT loopback LAG. As a result, the RIOT loopback LAG must be available to process VXLAN traffic before the IRB interfaces are able to route traffic. For this feature to work, the device must consider the RIOT loopback LAG state when determining IRB interface status.

With RIOT loopback configuration, you configure the device to include the state of another local interface when evaluating the status of an IRB interface. In this case, the local interface is the RIOT loopback LAG.

Use the `local-interface name` statement at the `[edit interfaces irb unit unit-number interface-state]` hierarchy. Specify *name* as the logical interface name of the RIOT loopback LAG for the unit.

Also, to configure the delay to ensure that the RIOT loopback LAG is up before the device evaluates the IRB interface as up, you configure the `hold-time up seconds` option at the `[edit interfaces irb unit unit-number interface-state]` hierarchy. This value is the time the device waits after the RIOT loopback interface is up before it includes that state when evaluating the IRB interface status.

The RIOT loopback LAG usually remains up unless you change its configuration. As a result, we recommend to set `hold-time up` to a higher value based on the scale of routes in your network. A higher value helps prevent the IRB logical interfaces from flapping. We recommend that you try approximately 120 seconds for medium-to-large scale deployments.

Configure a RIOT Loopback Port on a Layer 3 VXLAN Gateway Leaf Device

Follow these steps to configure a device to use the RIOT loopback process so it can operate as a Layer 3 VXLAN gateway. Some configuration steps are common for EVPN asymmetrical Type 2 routing and EVPN Type 5 routing, such as:

- Configure the RIOT loopback LAG and the IRB interfaces for VXLAN routing.
- Set the RIOT loopback LAG as the interface that handles RIOT loopback processing on the device.

You perform a few extra steps for EVPN Type 5 routing, including:

- Configure an extra VLAN for each EVPN Type 5 VRF instance. You don't use this VLAN for any other purpose.
- Configure an IRB interface on each extra VLAN.
- Map the extra VLAN to the VXLAN network identifier (VNI) of the corresponding Type 5 VRF instance.

See ["EVPN Type 5 Routes with RIOT Loopback Processing" on page 517](#) for more information on the extra VLAN for Type 5 routing. See ["How RIOT Loopback Processing Works" on page 516](#) for more information on the differences in the RIOT loopback process among the supported route types.

NOTE: Devices that require the RIOT loopback solution to act as a Layer 3 VXLAN gateway include the following statement in the default configuration:

```
set protocols evpn riot-loopback
```

This statement globally enables the RIOT loopback process on the device. You don't need to configure this statement explicitly.

To configure RIOT loopback processing:

1. Define an aggregated Ethernet interface for the RIOT loopback port. Use any network ports on the device in the RIOT loopback LAG that you aren't already using for network traffic.

The sample configuration below first allocates some number of aggregated Ethernet interfaces, and uses an available one (ae0) for the RIOT loopback LAG. For simplicity, this configuration includes one link in the RIOT loopback LAG, which must be up for the interface to be up. You can adjust the number of member links in the RIOT loopback LAG depending on the bandwidth of VXLAN traffic that uses the loopback path.

```
set chassis aggregated-devices ethernet device-count number

set interfaces xe-0/0/10:1 ether-options 802.3ad ae0
set interfaces ae0 aggregated-ether-options loopback
set interfaces ae0 aggregated-ether-options minimum-links 1
```

2. Configure the RIOT loopback LAG in the enterprise style with:

- Flexible VLAN tagging.
- Flexible Ethernet services encapsulation (so the interface can have multiple logical units).

For example:

```
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
```

3. Configure the RIOT loopback LAG interface in enterprise style as a member of all VLANs (units) that have IRB interfaces for which the device does Layer 3 VXLAN gateway routing. Also, configure the interface in trunk mode for each unit.

For example, here the fabric serves three VXLAN VLANs: V100, V110, and V120. Configure the RIOT loopback LAG in each VLAN with `interface-mode trunk`:

```
set interfaces ae0 unit 100 family ethernet-switching interface-mode trunk
set interfaces ae0 unit 100 family ethernet-switching vlan members V100

set interfaces ae0 unit 110 family ethernet-switching interface-mode trunk
set interfaces ae0 unit 110 family ethernet-switching vlan members V110

set interfaces ae0 unit 120 family ethernet-switching interface-mode trunk
set interfaces ae0 unit 120 family ethernet-switching vlan members V120
```

4. Configure an IRB interface for each VLAN (unit) used for VXLAN routing. This step isn't specific to the RIOT loopback process. However, it is a required part of the EVPN-VXLAN fabric setup. You use these IRB interfaces in subsequent steps.

For example, configure IRB interfaces for units 100, 110, and 120:

```
set interfaces irb unit 100 family inet address 10.100.1.254/16
set interfaces irb unit 100 family inet6 address 2001:db8::0192:0100:0001:0254/96
set interfaces irb unit 100 mac 00:01:01:00:00:fe

set interfaces irb unit 110 family inet address 10.110.1.254/16
set interfaces irb unit 110 family inet6 address 2001:db8::0192:0110:0001:0254/96
set interfaces irb unit 110 mac 00:01:01:10:00:fe

set interfaces irb unit 120 family inet address 10.120.1.254/16
set interfaces irb unit 120 family inet6 address 2001:db8::0192:0120:0001:0254/96
set interfaces irb unit 120 mac 00:01:01:20:00:fe
```

5. For each IRB interface, set the RIOT loopback LAG as a local interface whose state the device includes in evaluating the IRB interface state (up or down). Use the `local-interface name` statement at the `[edit interfaces irb unit unit-number interface-state]` hierarchy. Specify the logical interface name of the RIOT loopback LAG for the local interface name. Also set the `hold-time up` option to ensure

the RIOT loopback LAG is up before the device evaluates the IRB interface as up. See ["IRB Interface Status Dependency on RIOT Loopback Port State" on page 518](#) for more information on why we need this step.

For example, for IRB interfaces configured on units 100, 110, and 120, set the local interface to the RIOT loopback LAG logical interface name. Specify a hold time for each IRB—we recommend 120 seconds in this case for a medium-to-large scale deployment:

```
set interfaces irb unit 100 interface-state local-interface ae0.100
set interfaces irb unit 100 interface-state hold-time up 120

set interfaces irb unit 110 interface-state local-interface ae0.110
set interfaces irb unit 110 interface-state hold-time up 120

set interfaces irb unit 120 interface-state local-interface ae0.120
set interfaces irb unit 120 interface-state hold-time up 120
```

6. Set the RIOT loopback LAG as the interface the device uses for the RIOT loopback process for all VXLAN routing. The statement to do this is `loopback-port loopback-port` at the `[edit forwarding options vxlan-routing]` hierarchy level. With this statement, specify the physical interface name of the RIOT loopback port.

For example, in our sample configuration in earlier steps, the RIOT loopback LAG is ae0:

```
set forwarding-options vxlan-routing loopback-port ae0
```

7. Define each VXLAN VLAN. Set the IRB interfaces as Layer 3 IRB interfaces for each VLAN, and map the VLANs to VNI values. This step is required for VXLAN gateway configuration; it isn't specific to RIOT loopback configuration.

For example:

```
set vlans V100 vlan-id 100
set vlans V100 l3-interface irb.100
set vlans V100 vxlan vni 1100

set vlans V110 vlan-id 110
set vlans V110 l3-interface irb.110
set vlans V110 vxlan vni 1110
```

8. (EVPN Type 5 routing use cases only) Configure an additional VLAN for each EVPN Type 5 VRF instance, which the RIOT loopback process uses to support EVPN Type 5 routing. See ["EVPN Type](#)

[5 Routes with RIOT Loopback Processing" on page 517](#) for details. This VLAN is dedicated to this purpose and must be different from any tenant VLANs or VXLAN VLANs the device hosts.

This step combines the earlier steps where you configure the RIOT loopback LAG as part of the VXLAN VLANs. You do the same for this extra VLAN, including:

- Configure the VLAN with an IRB interface.
- Configure the RIOT loopback LAG logical interface for this unit in trunk mode.
- Set the RIOT loopback LAG interface as an IRB-enabled member of this VLAN.
- Set the RIOT loopback LAG as a local interface whose state the device includes in evaluating the IRB interface state (up or down).

For example, define an IRB-enabled VLAN named V-T5-RIOT1 with VLAN ID 999. Include the RIOT loopback LAG as a part of this VLAN. Also set the other parameters listed above that enable the RIOT loopback process:

```
set vlans V-T5-RIOT1 vlan-id 999
set vlans V-T5-RIOT1 l3-interface irb.999

set interfaces ae0 unit 999 family ethernet-switching interface-mode trunk
set interfaces ae0 unit 999 family ethernet-switching vlan members V-T5-RIOT1

set interfaces irb unit 999 family inet
set interfaces irb unit 999 mac 00:01:01:00:00:fe

set interfaces irb unit 999 interface-state local-interface ae0.999
set interfaces irb unit 999 interface-state hold-time up 120
```

NOTE: Repeat this step to create an extra VLAN for each EVPN Type 5 VRF instance.

9. (EVPN Type 5 use cases only) Configure the extra VLAN's IRB logical interface in the VRF instance where you enable Type 5 routing. Map the additional VLAN to a VNI that matches the EVPN encapsulation VNI you configure in the Type 5 VRF instance.

To enable EVPN Type 5 routing in an EVPN-VXLAN fabric, you set up a VRF instance. In that Type 5 VRF instance, you configure the `ip-prefix-routes vni vni-value` statement at the [edit routing-instances *type-5-instance-name* protocols *evpn*] hierarchy level. This *vni-value* is the value you map to the extra VLAN.

We don't include all of the standard EVPN Type 5 VRF instance configuration here. See ["EVPN Type 5 Route with VXLAN encapsulation for EVPN-VXLAN" on page 21](#) for more information on

Type 5 routes. Also see [Configuring EVPN Type 5 for QFX10000 Series Switches: Configuration Example](#) for an example configuration with Type 5 routing between two QFX Series devices in an EVPN network.

For example, if you configure the Type 5 VRF instance T5-VRF with an EVPN-VXLAN encapsulation VNI value of 5000 as follows:

```
set routing-instances T5-VRF protocols evpn ip-prefix-routes vni 5000
```

then map the extra VLAN from step 8 (V-T5-RIOT1 with VLAN ID 999) to VNI 5000:

```
set routing-instances T5-VRF interface irb.999
set vlans V-T5-RIOT1 vxlan vni 5000
```

NOTE: Repeat this step for the extra VLAN for each EVPN Type 5 VRF instance.

10. (EVPN Type 5 use cases only) Finally, configure the [riot-loopback](#) statement at the [edit vlans name vxlan] hierarchy to set the VLAN from step 8 as the extra RIOT loopback VLAN for Type 5 routing. For example:

```
set vlans V-T5-RIOT1 vxlan riot-loopback
```

NOTE: Repeat this step for the extra VLAN for each EVPN Type 5 VRF instance.

SEE ALSO

[Example: Configuring an EVPN-VXLAN Edge-Routed Bridging Overlay Within a Data Center | 617](#)
[Understanding VXLANs | 414](#)

Understanding the MAC Addresses For a Default Virtual Gateway in an EVPN-VXLAN Overlay Network

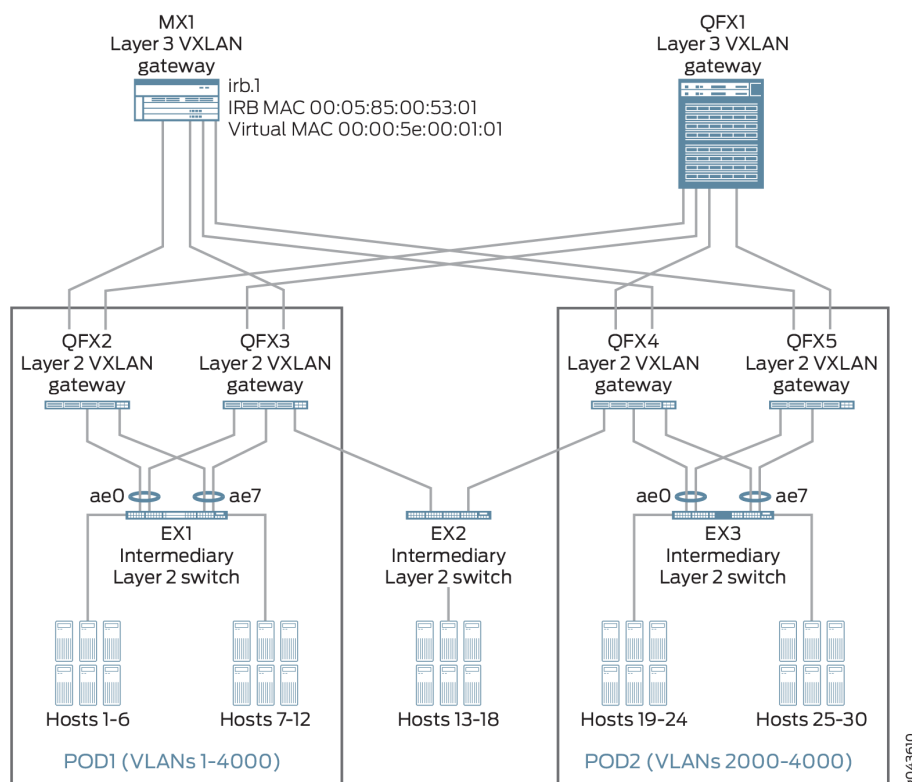
In an Ethernet VPN-Virtual Extensible LAN (EVPN-VXLAN) centrally-routed bridging overlay (EVPN-VXLAN topology with a two-layer IP fabric), an MX Series router or a QFX10000 switch can function as a Layer 3 VXLAN gateway on which you can configure integrated routing and bridging (IRB) interfaces. The configuration of each IRB interface can also include a virtual gateway address (VGA), which creates a default Layer 3 virtual gateway with the specified IP address. Through the IRB interface with which it is configured, the default virtual gateway enables the communication between non-virtualized hosts, virtual machines (VMs), and servers in different VXLANs or IP subnetworks.

When you configure a VGA for an IRB interface, the Layer 3 VXLAN gateway automatically generates IPv4 media access control (MAC) address 00:00:5E:00:01:01 or IPV6 MAC address 00:00:5E:00:02:01 for that particular virtual gateway. (This topic refers to the virtual gateway MAC address as a *virtual MAC*.) The automatically generated virtual MAC is not included as the source MAC address in packets generated by the Layer 3 VXLAN gateway. Instead, data packets and the source MAC address field in the outer Ethernet header of Address Resolution Protocol (ARP) replies and neighbor advertisement packets include the MAC address for the IRB interface. (This topic refers to the MAC address for the IRB interface as the *IRB MAC*.)

When an ARP reply includes the IRB MAC as the source MAC address instead of the virtual MAC, an issue might arise in a centrally-routed bridging overlay. For example, in the overlay network shown in [Figure 43 on page 525](#), an MX Series router and a QFX10000 switch function as Layer 3 VXLAN gateways, and four QFX5100 switches function as Layer 2 VXLAN gateways. Also included in the

overlay network are three intermediary Layer 2 switches, in this case, EX4300 switches, to which hosts are connected.

Figure 43: EVPN-VXLAN Centrally-Routed Bridging Overlay



On the MX Series router, an IRB interface named `irb.1` has a MAC address of `00:05:85:00:53:01` and a VSA of `10.2.1.254`. The MX Series router automatically generates the MAC address `00:00:5e:00:01:01` for the default virtual gateway.

In this overlay network, `irb.1` on the MX Series router receives an ARP request from host 1. In its ARP reply, the MX Series router includes the following:

- Source MAC address in outer Ethernet header: `00:05:85:00:53:01` (IRB MAC) → intermediary Layer 2 switch EX1 learns this MAC address.
- Sender MAC address within ARP reply packet: `00:00:5e:00:01:01` (virtual MAC) → intermediary Layer 2 switch EX1 cannot see this MAC address, and therefore, does not learn it.

When intermediary Layer 2 switch EX1 receives the ARP reply, it learns only the source MAC address (IRB MAC). As a result, if host 1 sends packets that include the virtual MAC in the header, EX1 is unable to find the virtual MAC in its MAC table. Therefore, EX1 floods the domain with unknown-unicast packets.

NOTE: The flooding of unknown-unicast packets is not an issue in EVPN-VXLAN edge-routed bridging overlays (EVPN-VXLAN topologies with a collapsed IP fabric), in which a single layer of QFX10000 switches function as both Layer 3 and Layer 2 VXLAN gateways. In the edge-routed bridging overlay, hosts are directly connected to the Layer 3 and Layer 2 VXLAN gateways. Further, each IRB interface is typically configured with an IP address and a static MAC address. The configuration of each IRB interface on a particular VXLAN gateway is repeated on each gateway in the edge-routed bridging overlay. With the same MAC address configured for each IRB interface on each VXLAN gateway, each host uses the same MAC address when sending inter-VXLAN traffic regardless of where the host is located or which VXLAN gateway receives the traffic. These factors make the configuration of a default virtual gateway unnecessary. For more information about the edge-routed bridging overlay, see ["Example: Configuring an EVPN-VXLAN Edge-Routed Bridging Overlay Within a Data Center" on page 617](#).

Starting with Junos OS Release 14.2R5 for MX Series routers and Junos OS Release 15.1X53-D63 for QFX10000 switches, you can explicitly configure an IPv4 or IPv6 MAC address for a default virtual gateway by using the `virtual-gateway-v4-mac` or `virtual-gateway-v6-mac` configuration statement at the `[edit interfaces name irb unit logical-unit-number]` hierarchy level. After you perform this configuration, the automatically generated virtual MAC is overridden by the configured virtual MAC. That is, when Layer 3 VXLAN gateway MX1 sends data packets, ARP replies, and neighbor advertisement packets, the configured virtual MAC is in the outer Ethernet header of these packets. As a result, intermediary Layer 2 switch EX1 also learns the configured virtual MAC, thereby eliminating the possibility that the switch floods the domain with unknown-unicast packets.

NOTE: The MAC address range 02:00:00:00:00:00:xy is used for internal communication. Do not use addresses in this range for a manual virtual MAC assignment.

Release History Table

Release	Description
14.2R5	Starting with Junos OS Release 14.2R5 for MX Series routers and Junos OS Release 15.1X53-D63 for QFX10000 switches, you can explicitly configure an IPv4 or IPv6 MAC address for a default virtual gateway by using the <code>virtual-gateway-v4-mac</code> or <code>virtual-gateway-v6-mac</code> configuration statement at the <code>[edit interfaces name irb unit logical-unit-number]</code> hierarchy level.

Supported Protocols on an IRB Interface in EVPN-VXLAN

EVPN-VXLAN provides logical layer 2 connectivity over a layer 3 network. It is a logical overlay that is decoupled from the underlay network. The layer 2 subnets in the EVPN-VXLAN network are uniquely identified by the virtual network identifier (VNI) where devices configured with the same VNI are considered to be in the same logical subnet. The devices in the same logical subnet can communicate directly with each other when they are connected to the same virtual gateway. To route traffic when devices with different VNIs are connected to the same gateway, you must configure an IRB interface on the VXLAN gateway to route traffic. This allows protocol adjacencies to be established between the layer 3 gateway IRB interface and customer edge devices in the following scenarios:

- Support networking functions when there are firewalls or load balancers connected between the host and the gateway device.
- Support inter-VNI routing when VNIs for the same tenant extend across data centers.

[Figure 44 on page 528](#) illustrates a simple leaf-spine (centrally-routed bridging overlay) topology in an EVPN-VXLAN network with the VXLAN gateway on spine devices. For host 1 to communicate with hosts in other VNIs, you must configure the IRB interfaces on the VXLAN gateway on the spine devices.

NOTE: When configuring an ESI interface for multihomed devices, Junos OS only supports routing protocols that are configured on an IRB in the all-active multihoming mode in a centrally-routed bridging overlay EVPN-VXLAN network.

Figure 44: Leaf-spine Topology in an EVPN-VXLAN network.

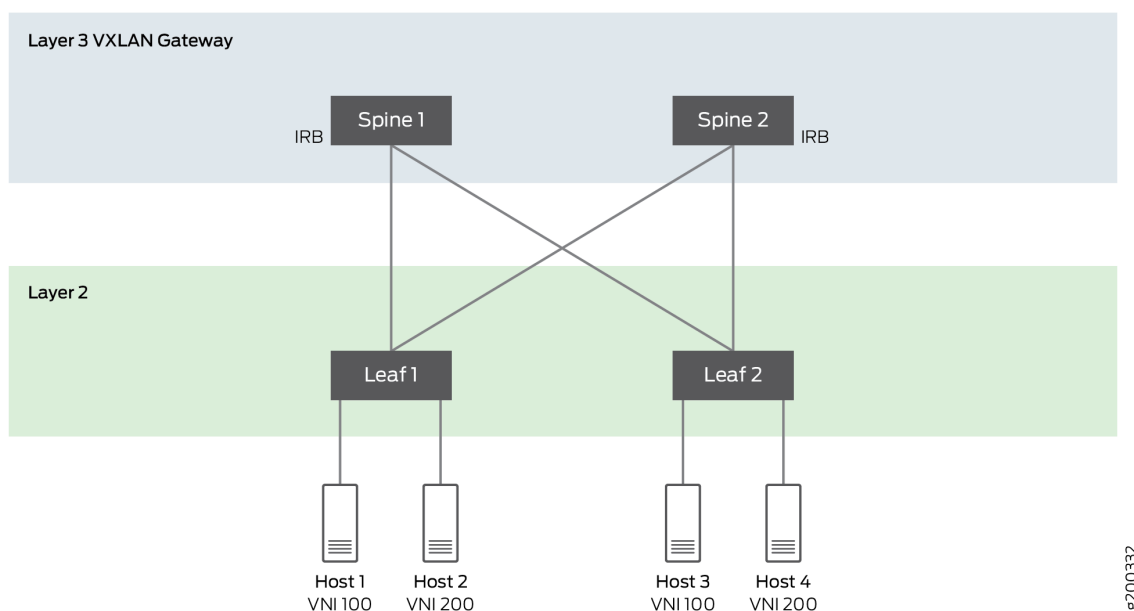
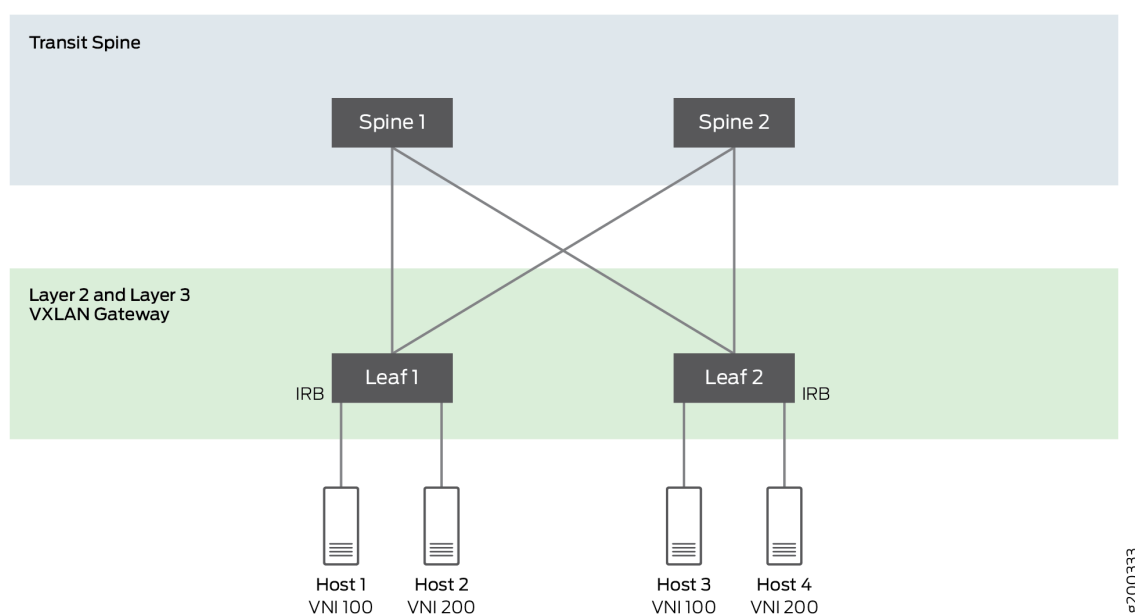


Figure 45 on page 529 illustrates a collapsed leaf-spine (edge-routed bridging) topology with VXLAN gateways on leaf devices. For host 1 to communicate with hosts in other VNIs, you must configure IRB interfaces on the VXLAN gateway on the leaf devices.

Figure 45: Collapsed Leaf-spine Topology in an EVPN-VXLAN network.



Junos OS supports the following protocols on the IRB in the EVPN-VXLAN overlay network:

- OSPF
- IS-IS
- BGP (eBGP and iBGP)
- Bidirectional forwarding detection (BFD) support for ISIS, OSPF, BGP and static routing.

RELATED DOCUMENTATION

[Understanding OSPF Routing Instances](#)

[Understanding IS-IS Configuration](#)

[Understanding External BGP Peering Sessions](#)

[Understanding Internal BGP Peering Sessions](#)

[Understanding BFD for Static Routes for Faster Network Failure Detection](#)

Example: Configure an EVPN-VXLAN Centrally-Routed Bridging (CRB) Overlay

IN THIS SECTION

- Requirements | 531
- Overview | 531
- Spine 1: Underlay Network Configuration | 535
- Spine 1: Overlay Network Configuration | 538
- Spine 1: Access Profile Configuration | 540
- Spine 2: Full Configuration | 543
- Leaf 1: Underlay Network Configuration | 545
- Leaf 1: Overlay Network Configuration | 547
- Leaf 1: Access Profile Configuration | 549
- Leaf 2: Full Configuration | 551
- Leaf 3: Full Configuration | 552
- Leaf 4: Full Configuration | 553
- Verification | 554
- Spine 1 and 2: Route Leaking (Optional) | 561
- Verification with Route Leaking (Optional) | 563

Modern data centers are based on an IP fabric that used BGP based EVPN signaling in the control plane with VXLAN encapsulation in the data plane. This technology provides a standards based, high performance solution for Layer 2 bridging within a VLAN and routing between VLANs.

In most cases there is a one to one relationship between a user VLAN and VXLAN network identifier (VNI). As a result the terms VLAN and VXLAN are often used interchangeably. By default VXLAN encapsulation strips any ingress VLAN tag when received from an access port. The rest of the Ethernet frame is encapsulated in VXLAN for transport across the fabric. At the egress, the VXLAN encapsulation is stripped and the VLAN tag (if any) is reinserted before the frame is sent to the attached device.

For background information on EVPN-VXLAN technology, see [EVPN Primer](#).

Starting with Junos OS Release 15.1X53-D30, you can use integrated routing and bridging (IRB) interfaces to route between VLANs using Virtual Extensible LAN (VXLAN) encapsulation.

In this example, the IRB interfaces provide Layer 3 connectivity to physical servers in the same data center.

NOTE: We updated and revalidated this example using Junos OS Release 20.4R1.

Requirements

The original example used the following hardware and software components:

- Two QFX10002 switches (Spine 1 and Spine 2) running Junos OS Release 15.1X53-D30 software.
- Four QFX5100 switches (Leaf 1 through Leaf 4) running Junos OS Release 14.1X53-D30 software.
 - Updated and revalidated using Junos OS Release 20.4R1.
 - See the [hardware summary](#) for a list of supported platforms.

Overview

IN THIS SECTION

- [Topology](#) | 532

In this example, physical servers that support three groups of users, i.e., three VLANs, require the following connectivity:

1. Servers A and C should be able to communicate at Layer 2. The servers must share a subnet.
2. Servers B and D must be on separate VLANs to isolate broadcast. These servers must be able to communicate at Layer 3.
3. Servers A and C should not be able to communicate with Servers B and D.

To meet these connectivity requirements the below protocols and technologies are used:

- EVPN is used to establish a Layer 2 virtual bridge to connect the servers A and C, and to place servers B and D into their respective VLANs.
- Within the EVPN topology, BGP is used to exchange route information.
- VXLAN is used to tunnel the Layer 2 frames through the underlying Layer 3 fabric. The use of VXLAN encapsulation preserves the Layer 3 fabric for use by routing protocols.

- IRB interfaces are used to route IP packets between the VLANs.

The IRB interfaces are configured on the spine devices only. Hence the phrase centrally-routed with edge bridging (CRB). With this configuration, the spine devices function as Layer 3 gateways for the physical servers that are connected through their respective leaf devices. When the physical servers exchange data, the spine devices route the IP packets to their respective destinations.

Topology

The simple IP Clos topology shown in [Figure 46 on page 533](#) includes two spine switches, four leaf switches, and four servers. Each leaf switch has a connection to each of the spine switches for redundancy.

The server networks are segmented into 3 VLANs, each of which is mapped to a VXLAN Virtual Network Identifier (VNI). VLAN v101 supports servers A and C, and VLANs v102 and v103 support servers B and D, respectively. See [Table 27 on page 534](#) for configuration parameters.

Figure 46: EVPN-VXLAN Topology

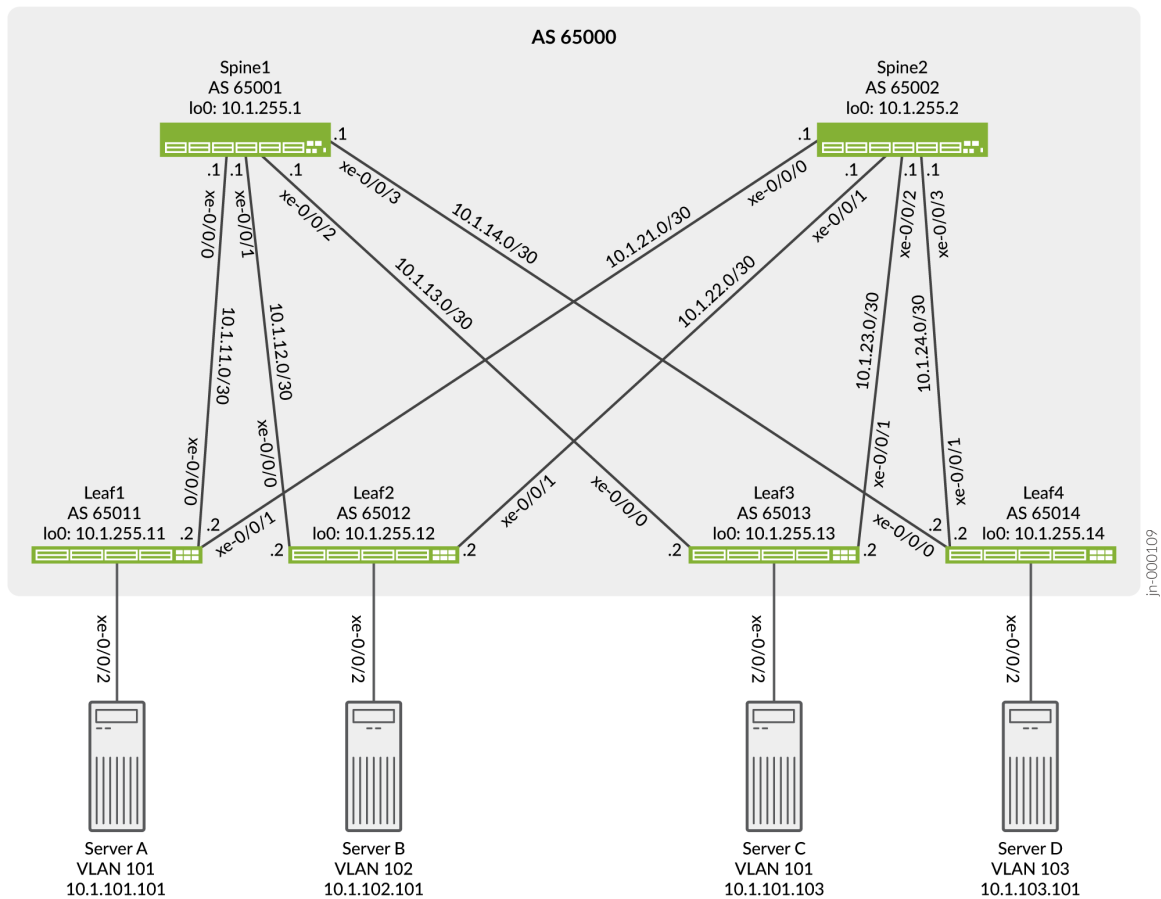


Figure 47: EVPN-VXLAN Logical Topology

The logical topology shows the expected connectivity. In this example one routing instance is used to connect servers A and C using VLAN 101, and one routing instance is used to connect servers B and D using VLANs 102 and 103. The servers are only able to communicate with other servers that are in the same routing instance by default.

Because servers A and C share the same VLAN, the servers communicate at Layer 2. Therefore, the IRB interface is not needed for servers A and C to communicate. We define an IRB interface in the routing

instance as a best practice to enable future Layer 3 connectivity. In contrast, Servers B and D require Layer 3 connectivity through their respective IRBs to communicate as the servers are in different VLANs/IP subnets.

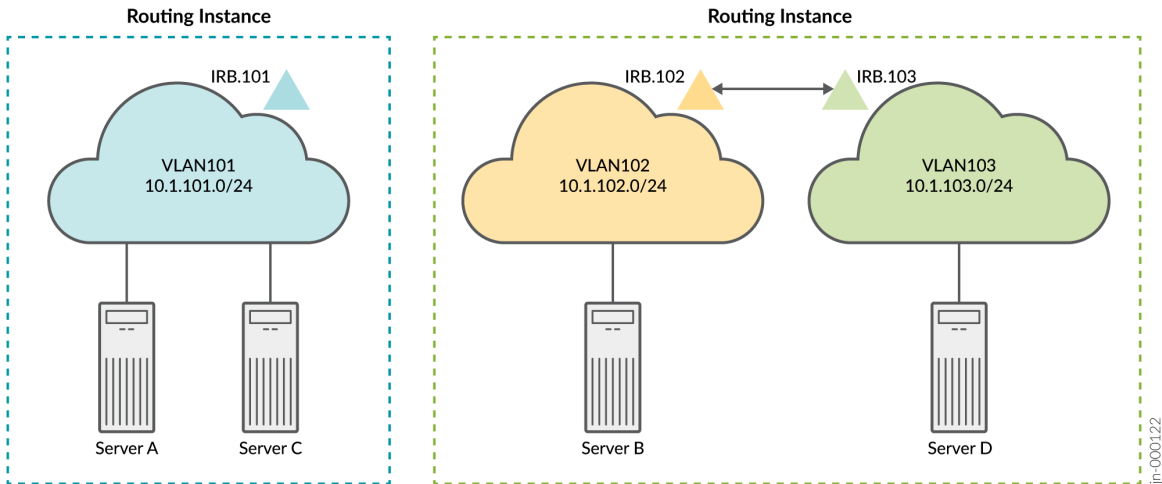


Table 27 on page 534 provides key parameters, including the IRB interfaces, configured for each network. Each VLAN is supported by an IRB interface, over which data packets from the other VLANs are routed.

Table 27: Key VLAN and VXLAN Parameters

Parameters	Servers A and C	Servers B and C
VLAN	v101	v102
		v103
VXLAN VNI	101	102
		103
VLAN ID	101	102
		103

Table 27: Key VLAN and VXLAN Parameters *(Continued)*

Parameters	Servers A and C	Servers B and C
IRB interface	irb.101	irb.102
		irb.103

Keep the following in mind when you configure the parameters in [Table 27 on page 534](#):

- Each VLAN must be associated with a unique IP subnet, and therefore a unique IRB interface.
- Each VLAN must be assigned a unique VXLAN network identifier (VNI).
- Each IRB interface is specified as part of a Layer 3 virtual routing forwarding (VRF) instance, or the interfaces can be lumped together in the default switch instance. This example uses VRF instances to enforce separation between the user community (VLANs).
- The configuration of each IRB interface must include a default gateway address, which is specified with the `virtual-gateway-address` configuration statement at the `[interfaces irb unit logical-unit-number family inet address ip-address]` hierarchy level. Configuring a virtual gateway sets up a redundant default gateway for each IRB interface.

Spine 1: Underlay Network Configuration

IN THIS SECTION

- [CLI Quick Configuration | 535](#)
- [Spine 1: Configure the Underlay Network | 536](#)

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the `[edit]` hierarchy level, and then enter `commit` from configuration mode.

```
set interfaces xe-0/0/0 unit 0 family inet address 10.1.11.1/30
set interfaces xe-0/0/1 unit 0 family inet address 10.1.12.1/30
set interfaces xe-0/0/2 unit 0 family inet address 10.1.13.1/30
```

```

set interfaces xe-0/0/3 unit 0 family inet address 10.1.14.1/30
set interfaces lo0 unit 0 family inet address 10.1.255.1/32 primary
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options router-id 10.1.255.1
set routing-options autonomous-system 65000
set routing-options forwarding-table export load-balancing-policy
set protocols bgp group underlay type external
set protocols bgp group underlay export send-direct
set protocols bgp group underlay local-as 65001
set protocols bgp group underlay multipath multiple-as
set protocols bgp group underlay neighbor 10.1.11.2 peer-as 65011
set protocols bgp group underlay neighbor 10.1.12.2 peer-as 65012
set protocols bgp group underlay neighbor 10.1.13.2 peer-as 65013
set protocols bgp group underlay neighbor 10.1.14.2 peer-as 65014

```

Spine 1: Configure the Underlay Network

Step-by-Step Procedure

To configure the underlay network on Spine 1:

1. Configure the Layer 3 fabric interfaces.

```

[edit]
user@Spine1# set interfaces xe-0/0/0 unit 0 family inet address 10.1.11.1/30
user@Spine1# set interfaces xe-0/0/1 unit 0 family inet address 10.1.12.1/30
user@Spine1# set interfaces xe-0/0/2 unit 0 family inet address 10.1.13.1/30
user@Spine1# set interfaces xe-0/0/3 unit 0 family inet address 10.1.14.1/30

```

2. Specify an IP address for the loopback interface. This IP address serves as the source IP address in the outer header of VXLAN encapsulated packets.

```

[edit]
user@Spine1# set interfaces lo0 unit 0 family inet address 10.1.255.1/32 primary

```

3. Configure the routing options. The configuration includes reference to a load balancing policy to enable the use of equal cost multiple path (ECMP) routing through the underlay.

```
[edit]
user@Spine1# set routing-options router-id 10.1.255.1
user@Spine1# set routing-options autonomous-system 65000
user@Spine1# set routing-options forwarding-table export load-balancing-policy
```

4. Configure a BGP group that includes EBGP peers for the underlay. Note that multipath is included in the BGP configuration to allow multiple equal cost paths to be installed. Normally BGP uses a tie breaking algorithm that selects a single best path,.

```
[edit]
user@Spine1# set protocols bgp group underlay type external
user@Spine1# set protocols bgp group underlay export send-direct
user@Spine1# set protocols bgp group underlay local-as 65001
user@Spine1# set protocols bgp group underlay multipath multiple-as
user@Spine1# set protocols bgp group underlay neighbor 10.1.11.2 peer-as 65011
user@Spine1# set protocols bgp group underlay neighbor 10.1.12.2 peer-as 65012
user@Spine1# set protocols bgp group underlay neighbor 10.1.13.2 peer-as 65013
user@Spine1# set protocols bgp group underlay neighbor 10.1.14.2 peer-as 65014
```

5. Configure the per-packet load-balancing policy.

```
[edit]
user@Spine1# set policy-options policy-statement load-balancing-policy then load-balance per-packet
```

6. Configure a policy to advertise direct interface routes into underlay. At a minimum you must advertise the loopback interfaces (lo0) routes into the underlay.

```
[edit]
user@Spine1# set policy-options policy-statement send-direct term 1 from protocol direct
user@Spine1# set policy-options policy-statement send-direct term 1 then accept
```

Spine 1: Overlay Network Configuration

IN THIS SECTION

- [CLI Quick Configuration | 538](#)
- [Configuring the Overlay Network on Spine 1 | 538](#)

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste the configuration into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [edit] hierarchy level, and then enter `commit` from configuration mode.

```
set protocols bgp group evpn type internal
set protocols bgp group evpn local-address 10.1.255.1
set protocols bgp group evpn family evpn signaling
set protocols bgp group evpn cluster 1.1.1.1
set protocols bgp group evpn multipath
set protocols bgp group evpn neighbor 10.1.255.2
set protocols bgp group evpn neighbor 10.1.255.11
set protocols bgp group evpn neighbor 10.1.255.12
set protocols bgp group evpn neighbor 10.1.255.13
set protocols bgp group evpn neighbor 10.1.255.14
set protocols evpn encapsulation vxlan
set protocols evpn multicast-mode ingress-replication
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 10.1.255.1:1
set switch-options vrf-target target:65000:1
set switch-options vrf-target auto
```

Configuring the Overlay Network on Spine 1

Step-by-Step Procedure

To configure the overlay network on Spine 1:

1. Configure an internal BGP (IBGP) group for the EVPN-VXLAN overlay. Note that the EVPN family is specified to allow advertisement of EVPN routes and we added a peering to Spine2 to allow for spine-to-spine connectivity. As with the underlay, BGP multipath is also enabled in the overlay.

```
[edit]
user@Spine1# set protocols bgp group evpn type internal
user@Spine1# set protocols bgp group evpn local-address 10.1.255.1
user@Spine1# set protocols bgp group evpn family evpn signaling
user@Spine1# set protocols bgp group evpn cluster 1.1.1.1
user@Spine1# set protocols bgp group evpn multipath
user@Spine1# set protocols bgp group evpn neighbor 10.1.255.2
user@Spine1# set protocols bgp group evpn neighbor 10.1.255.11
user@Spine1# set protocols bgp group evpn neighbor 10.1.255.12
user@Spine1# set protocols bgp group evpn neighbor 10.1.255.13
user@Spine1# set protocols bgp group evpn neighbor 10.1.255.14
```

2. Configure VXLAN encapsulation for the Layer 2 frames exchanged between the EVPN neighbors.

```
[edit]
user@Spine1# set protocols evpn encapsulation vxlan
```

3. Specify how multicast traffic is replicated between Juniper Networks switches in an EVPN-VXLAN environment.

```
[edit]
user@Spine1# set protocols evpn multicast-mode ingress-replication
```

4. Configure options for the default routing instance (virtual switch type).

```
[edit]
user@Spine1# set switch-options vtep-source-interface lo0.0
user@Spine1# set switch-options route-distinguisher 10.1.255.1:1
user@Spine1# set switch-options vrf-target target:65000:1
user@Spine1# set switch-options vrf-target auto
```

Spine 1: Access Profile Configuration

IN THIS SECTION

- [CLI Quick Configuration | 540](#)
- [Configuring the Access Profiles for Spine 1 | 541](#)

CLI Quick Configuration

The access profile or access port configuration involves the settings needed to attach server workloads, BMS or VMs, to access (leaf) switches. This step involves definition of the device's VLAN, along with routing instance and IRB configuration that provide user isolation and Layer 3 routing, respectively.

To quickly configure this example, copy the following commands, paste the commands into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [edit] hierarchy level, and then enter `commit` from configuration mode.

NOTE: When [proxy-macip-advertisement](#) is enabled, the Layer 3 gateway advertises MAC and IP routes (MAC+IP type 2 routes) on behalf of Layer 2 VXLAN gateways in EVPN-VXLAN networks. This behavior is not supported on EVPN-MPLS. Starting in Junos OS Release 20.2R2, the warning message, WARNING: Only EVPN VXLAN supports proxy-macip-advertisement configuration, appears when you enable proxy-macip-advertisement. The message appears when you change your configuration, save your configuration, or use the `show` command to display your configuration

```
set interfaces irb unit 101 proxy-macip-advertisement
set interfaces irb unit 101 virtual-gateway-accept-data
set interfaces irb unit 101 family inet address 10.1.101.1/24 virtual-gateway-address
10.1.101.254
set interfaces irb unit 102 proxy-macip-advertisement
set interfaces irb unit 102 virtual-gateway-accept-data
set interfaces irb unit 102 family inet address 10.1.102.1/24 virtual-gateway-address
10.1.102.254
set interfaces irb unit 103 proxy-macip-advertisement
set interfaces irb unit 103 virtual-gateway-accept-data
set interfaces irb unit 103 family inet address 10.1.103.1/24 virtual-gateway-address
10.1.103.254
```

```

set protocols evpn vni-options vni 101 vrf-target target:65000:101
set protocols evpn vni-options vni 102 vrf-target target:65000:102
set protocols evpn vni-options vni 103 vrf-target target:65000:103
set protocols evpn extended-vni-list 101
set protocols evpn extended-vni-list 102
set protocols evpn extended-vni-list 103
set routing-instances serverAC instance-type vrf
set routing-instances serverAC interface irb.101
set routing-instances serverAC route-distinguisher 10.1.255.1:13
set routing-instances serverAC vrf-target target:65000:13
set routing-instances serverBD instance-type vrf
set routing-instances serverBD interface irb.102
set routing-instances serverBD interface irb.103
set routing-instances serverBD route-distinguisher 10.1.255.1:24
set routing-instances serverBD vrf-target target:65000:24
set vlans v101 vlan-id 101
set vlans v101 l3-interface irb.101
set vlans v101 vxlan vni 101
set vlans v102 vlan-id 102
set vlans v102 l3-interface irb.102
set vlans v102 vxlan vni 102
set vlans v103 vlan-id 103
set vlans v103 l3-interface irb.103
set vlans v103 vxlan vni 103

```

Configuring the Access Profiles for Spine 1

Step-by-Step Procedure

To configure profiles for the server networks:

1. Configure the IRB interfaces that support routing among VLANs 101, 102, and 103.

```

[edit]
user@Spine1# set interfaces irb unit 101 proxy-macip-advertisement
user@Spine1# set interfaces irb unit 101 virtual-gateway-accept-data
user@Spine1# set interfaces irb unit 101 family inet address 10.1.101.1/24 virtual-gateway-
address 10.1.101.254
user@Spine1# set interfaces irb unit 102 proxy-macip-advertisement
user@Spine1# set interfaces irb unit 102 virtual-gateway-accept-data
user@Spine1# set interfaces irb unit 102 family inet address 10.1.102.1/24 virtual-gateway-

```



```

address 10.1.102.254
user@Spine1# set interfaces irb unit 103 proxy-macip-advertisement
user@Spine1# set interfaces irb unit 103 virtual-gateway-accept-data
user@Spine1# set interfaces irb unit 103 family inet address 10.1.103.1/24 virtual-gateway-
address 10.1.103.254

```

2. Specify which VNIs are included in the EVPN-VXLAN domain.

```

[edit]
user@Spine1# set protocols evpn extended-vni-list 101
user@Spine1# set protocols evpn extended-vni-list 102
user@Spine1# set protocols evpn extended-vni-list 103

```

3. Configure a route target for each VNI.

```

[edit]
user@Spine1# set protocols evpn vni-options vni 101 vrf-target target:65000:101
user@Spine1# set protocols evpn vni-options vni 102 vrf-target target:65000:102
user@Spine1# set protocols evpn vni-options vni 103 vrf-target target:65000:103

```

NOTE: In the original configuration, the spine devices run Junos OS Release 15.1X53-D30, and the leaf devices run 14.1X53-D30. In these software releases, when you include the vrf-target configuration statement in the [edit protocols evpn vni-options vni] hierarchy level, you must also include the export option. Note that later Junos OS releases do not require this option. As a result the configurations in this updated example omit the export option.

4. Configure the routing instance for servers A and C.

```

[edit]
user@Spine1# set routing-instances serverAC instance-type vrf
user@Spine1# set routing-instances serverAC interface irb.101
user@Spine1# set routing-instances serverAC route-distinguisher 10.1.255.1:13
user@Spine1# set routing-instances serverAC vrf-target target:65000:13

```

5. And now the routing instance for servers B and D.

```
[edit]
user@Spine1# set routing-instances serverBD instance-type vrf
user@Spine1# set routing-instances serverBD interface irb.102
user@Spine1# set routing-instances serverBD interface irb.103
user@Spine1# set routing-instances serverBD route-distinguisher 10.1.255.1:24
user@Spine1# set routing-instances serverBD vrf-target target:65000:24
```

6. Configure VLANs v101, v102, and v103, and associate the corresponding VNIs and IRB interfaces with each VLAN.

```
[edit]
user@Spine1# set vlans v101 vlan-id 101
user@Spine1# set vlans v101 l3-interface irb.101
user@Spine1# set vlans v101 vxlan vni 101
user@Spine1# set vlans v102 vlan-id 102
user@Spine1# set vlans v102 l3-interface irb.102
user@Spine1# set vlans v102 vxlan vni 102
user@Spine1# set vlans v103 vlan-id 103
user@Spine1# set vlans v103 l3-interface irb.103
user@Spine1# set vlans v103 vxlan vni 103
```

Spine 2: Full Configuration

IN THIS SECTION

- [CLI Quick Configuration | 543](#)

CLI Quick Configuration

Spine2's configuration is similar to Spine1 so we are providing the full configuration instead of step-by-step configuration. To quickly configure this example, copy the following commands, paste the commands into a text file, remove any line breaks, change any details necessary to match your

configuration, copy and paste the commands into the CLI at the [edit] hierarchy level, and then enter commit from configuration mode.

```

set interfaces xe-0/0/0 unit 0 family inet address 10.1.21.1/30
set interfaces xe-0/0/1 unit 0 family inet address 10.1.22.1/30
set interfaces xe-0/0/2 unit 0 family inet address 10.1.23.1/30
set interfaces xe-0/0/3 unit 0 family inet address 10.1.24.1/30
set interfaces irb unit 101 proxy-macip-advertisement
set interfaces irb unit 101 virtual-gateway-accept-data
set interfaces irb unit 101 family inet address 10.1.101.2/24 virtual-gateway-address
10.1.101.254
set interfaces irb unit 102 proxy-macip-advertisement
set interfaces irb unit 102 virtual-gateway-accept-data
set interfaces irb unit 102 family inet address 10.1.102.2/24 virtual-gateway-address
10.1.102.254
set interfaces irb unit 103 proxy-macip-advertisement
set interfaces irb unit 103 virtual-gateway-accept-data
set interfaces irb unit 103 family inet address 10.1.103.2/24 virtual-gateway-address
10.1.103.254
set interfaces lo0 unit 0 family inet address 10.1.255.2/32 primary
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-instances serverAC instance-type vrf
set routing-instances serverAC interface irb.101
set routing-instances serverAC route-distinguisher 10.1.255.1:13
set routing-instances serverAC vrf-target target:65000:13
set routing-instances serverBD instance-type vrf
set routing-instances serverBD interface irb.102
set routing-instances serverBD interface irb.103
set routing-instances serverBD route-distinguisher 10.1.255.1:24
set routing-instances serverBD vrf-target target:65000:24
set routing-options router-id 10.1.255.2
set routing-options autonomous-system 65000
set routing-options forwarding-table export load-balancing-policy
set protocols bgp group underlay type external
set protocols bgp group underlay export send-direct
set protocols bgp group underlay local-as 65002
set protocols bgp group underlay multipath multiple-as
set protocols bgp group underlay neighbor 10.1.21.2 peer-as 65011
set protocols bgp group underlay neighbor 10.1.22.2 peer-as 65012
set protocols bgp group underlay neighbor 10.1.23.2 peer-as 65013

```

```

set protocols bgp group underlay neighbor 10.1.24.2 peer-as 65014
set protocols bgp group evpn type internal
set protocols bgp group evpn local-address 10.1.255.2
set protocols bgp group evpn family evpn signaling
set protocols bgp group evpn cluster 2.2.2.2
set protocols bgp group evpn multipath
set protocols bgp group evpn neighbor 10.1.255.1
set protocols bgp group evpn neighbor 10.1.255.11
set protocols bgp group evpn neighbor 10.1.255.12
set protocols bgp group evpn neighbor 10.1.255.13
set protocols bgp group evpn neighbor 10.1.255.14
set protocols evpn encapsulation vxlan
set protocols evpn multicast-mode ingress-replication
set protocols evpn vni-options vni 101 vrf-target target:65000:101
set protocols evpn vni-options vni 102 vrf-target target:65000:102
set protocols evpn vni-options vni 103 vrf-target target:65000:103
set protocols evpn extended-vni-list 101
set protocols evpn extended-vni-list 102
set protocols evpn extended-vni-list 103
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 10.1.255.2:1
set switch-options vrf-target target:65000:1
set switch-options vrf-target auto
set vlans v101 vlan-id 101
set vlans v101 l3-interface irb.101
set vlans v101 vxlan vni 101
set vlans v102 vlan-id 102
set vlans v102 l3-interface irb.102
set vlans v102 vxlan vni 102
set vlans v103 vlan-id 103
set vlans v103 l3-interface irb.103
set vlans v103 vxlan vni 103

```

Leaf 1: Underlay Network Configuration

IN THIS SECTION

- [CLI Quick Configuration | 546](#)
- [Configuring the Underlay Network for Leaf 1 | 546](#)

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste the commands into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [edit] hierarchy level, and then enter `commit` from configuration mode.

```
set interfaces xe-0/0/0 unit 0 family inet address 10.1.11.2/30
set interfaces xe-0/0/1 unit 0 family inet address 10.1.21.2/30
set interfaces lo0 unit 0 family inet address 10.1.255.11/32 primary
set routing-options router-id 10.1.255.11
set routing-options autonomous-system 65000
set routing-options forwarding-table export load-balancing-policy
set protocols bgp group underlay type external
set protocols bgp group underlay export send-direct
set protocols bgp group underlay local-as 65011
set protocols bgp group underlay multipath multiple-as
set protocols bgp group underlay neighbor 10.1.11.1 peer-as 65001
set protocols bgp group underlay neighbor 10.1.21.1 peer-as 65002
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
```

Configuring the Underlay Network for Leaf 1

Step-by-Step Procedure

To configure the underlay network for Leaf 1:

1. Configure the Layer 3 interfaces.

```
[edit]
user@Leaf1# set interfaces xe-0/0/0 unit 0 family inet address 10.1.11.2/30
user@Leaf1# set interfaces xe-0/0/1 unit 0 family inet address 10.1.21.2/30
```

2. Specify an IP address for the loopback interface. This IP address serves as the source IP address in the outer header of any VXLAN-encapsulated packets.

```
[edit]
user@Leaf1# set interfaces lo0 unit 0 family inet address 10.1.255.11/32 primary
```

3. Set the routing options.

```
[edit]
user@Leaf1# set routing-options router-id 10.1.255.11
user@Leaf1# set routing-options autonomous-system 65000
user@Leaf1# set routing-options forwarding-table export load-balancing-policy
```

4. Configure an EBGp group that includes the spines as peers to handle underlay routing.

```
[edit]
user@Leaf1# set protocols bgp group underlay type external
user@Leaf1# set protocols bgp group underlay export send-direct
user@Leaf1# set protocols bgp group underlay local-as 65011
user@Leaf1# set protocols bgp group underlay multipath multiple-as
user@Leaf1# set protocols bgp group underlay neighbor 10.1.11.1 peer-as 65001
user@Leaf1# set protocols bgp group underlay neighbor 10.1.21.1 peer-as 65002
```

5. Configure a policy that spreads traffic across multiple paths between the Juniper Networks switches.

```
[edit]
user@Leaf1# set policy-options policy-statement load-balancing-policy then load-balance per-
packet
```

6. Configure a policy to advertise direct interface routes. At a minimum the underlay must have full reachability to the device loopback addresses.

```
[edit]
user@Leaf1# set policy-options policy-statement send-direct term 1 from protocol direct
user@Leaf1# set policy-options policy-statement send-direct term 1 then accept
```

Leaf 1: Overlay Network Configuration

IN THIS SECTION

- [CLI Quick Configuration | 548](#)
- [Configuring the Overlay Network for Leaf 1 | 548](#)

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste the commands into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [edit] hierarchy level, and then enter `commit` from configuration mode.

```
set protocols bgp group evpn type internal
set protocols bgp group evpn local-address 10.1.255.11
set protocols bgp group evpn family evpn signaling
set protocols bgp group evpn multipath
set protocols bgp group evpn neighbor 10.1.255.1
set protocols bgp group evpn neighbor 10.1.255.2
set protocols evpn encapsulation vxlan
set protocols evpn multicast-mode ingress-replication
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 10.1.255.11:1
set switch-options vrf-target target:65000:1
set switch-options vrf-target auto
```

Configuring the Overlay Network for Leaf 1

Step-by-Step Procedure

To configure the overlay network for Leaf 1:

1. Configure an IBGP group for the EVPN-VXLAN overlay network.

```
[edit]
user@Leaf1# set protocols bgp group evpn type internal
user@Leaf1# set protocols bgp group evpn local-address 10.1.255.11
user@Leaf1# set protocols bgp group evpn family evpn signaling
user@Leaf1# set protocols bgp group evpn multipath
user@Leaf1# set protocols bgp group evpn neighbor 10.1.255.1
user@Leaf1# set protocols bgp group evpn neighbor 10.1.255.2
```

2. Configure VXLAN encapsulation for the data packets exchanged between the EVPN neighbors.

```
[edit]
user@Leaf1# set protocols evpn encapsulation vxlan
```

3. Specify how multicast traffic is replicated between Juniper Networks switches in an EVPN-VXLAN environment.

```
[edit]
user@Leaf1# set protocols evpn multicast-mode ingress-replication
```

4. Configure options for the default routing instance (virtual switch type).

```
[edit]
user@Leaf1# set switch-options vtep-source-interface lo0.0
user@Leaf1# set switch-options route-distinguisher 10.1.255.11:1
user@Leaf1# set switch-options vrf-target target:65000:1
user@Leaf1# set switch-options vrf-target auto
```

Leaf 1: Access Profile Configuration

IN THIS SECTION

- [CLI Quick Configuration | 549](#)
- [Configuring the Access Profile for Leaf 1 | 550](#)

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste the commands into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [edit] hierarchy level, and then enter `commit` from configuration mode.

```
set interfaces xe-0/0/2 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/2 unit 0 family ethernet-switching vlan members 101
set protocols evpn vni-options vni 101 vrf-target target:65000:101
set protocols evpn extended-vni-list 101
set vlans v101 vlan-id 101
set vlans v101 vxlan vni 101
```


Configuring the Access Profile for Leaf 1

Step-by-Step Procedure

To configure the profile for the server network:

1. Configure a Layer 2 Ethernet interface for the connection with the physical server. This interface is associated with VLAN 101. In this example the access interface is configured as a trunk to support VLAN tagging. Untagged access interfaces are also supported.

```
[edit]
user@Leaf1# set interfaces xe-0/0/2 unit 0 family ethernet-switching interface-mode trunk
user@Leaf1# set interfaces xe-0/0/2 unit 0 family ethernet-switching vlan members 101
```

2. Configure a route target for the VNI.

```
[edit]
user@Leaf1# set protocols evpn vni-options vni 101 vrf-target target:65000:101
```

NOTE: In the original configuration, the spine devices run Junos OS Release 15.1X53-D30, and the leaf devices run 14.1X53-D30. In these software releases, when you include the vrf-target configuration statement in the [edit protocols evpn vni-options vni] hierarchy level, you must also include the export option. Note that later Junos OS releases do not require this option, as reflected in the updated configurations used in this example.

3. Specify which VNI is included in the EVPN-VXLAN domains.

```
[edit]
user@Leaf1# set protocols evpn extended-vni-list 101
```

4. Configure VLAN v101. The VLAN is mapped to the same VXLAN VNI you configured on the spine devices. Note that the Layer 3 IRB interface is not specified on the leaf devices. This is because in CBR the leaves perform Layer 2 bridging only.

```
[edit]
user@Leaf1# set vlans v101 vlan-id 101
user@Leaf1# set vlans v101 vxlan vni 101
```

Leaf 2: Full Configuration

IN THIS SECTION

- [CLI Quick Configuration | 551](#)

CLI Quick Configuration

Leaf2's configuration is similar to Leaf1 so we are providing the full configuration instead of step-by-step configuration. To quickly configure this example, copy the following commands, paste the commands into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [edit] hierarchy level, and then enter `commit` from configuration mode.

```
set interfaces xe-0/0/0 unit 0 family inet address 10.1.12.2/30
set interfaces xe-0/0/1 unit 0 family inet address 10.1.22.2/30
set interfaces xe-0/0/2 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/2 unit 0 family ethernet-switching vlan members 102
set interfaces lo0 unit 0 family inet address 10.1.255.12/32 primary
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options router-id 10.1.255.12
set routing-options autonomous-system 65000
set routing-options forwarding-table export load-balancing-policy
set protocols bgp group underlay type external
set protocols bgp group underlay export send-direct
set protocols bgp group underlay local-as 65012
set protocols bgp group underlay multipath multiple-as
set protocols bgp group underlay as-override
set protocols bgp group underlay neighbor 10.1.12.1 peer-as 65001
set protocols bgp group underlay neighbor 10.1.22.1 peer-as 65002
set protocols bgp group evpn type internal
set protocols bgp group evpn local-address 10.1.255.12
set protocols bgp group evpn family evpn signaling
set protocols bgp group evpn multipath
set protocols bgp group evpn neighbor 10.1.255.1
set protocols bgp group evpn neighbor 10.1.255.2
set protocols evpn encapsulation vxlan
```

```

set protocols evpn multicast-mode ingress-replication
set protocols evpn vni-options vni 102 vrf-target target:65000:102
set protocols evpn extended-vni-list 102
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 10.1.255.12:1
set switch-options vrf-target target:65000:1
set switch-options vrf-target auto
set vlans v102 vlan-id 102
set vlans v102 vxlan vni 102

```

Leaf 3: Full Configuration

IN THIS SECTION

- [CLI Quick Configuration | 552](#)

CLI Quick Configuration

Leaf3's configuration is similar to Leaf1 so we are providing the full configuration instead of step-by-step configuration. To quickly configure this example, copy the following commands, paste the commands into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [edit] hierarchy level, and then enter `commit` from configuration mode.

```

set interfaces xe-0/0/0 unit 0 family inet address 10.1.13.2/30
set interfaces xe-0/0/1 unit 0 family inet address 10.1.23.2/30
set interfaces xe-0/0/2 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/2 unit 0 family ethernet-switching vlan members 101
set interfaces lo0 unit 0 family inet address 10.1.255.13/32 primary
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options router-id 10.1.255.13
set routing-options autonomous-system 65000
set routing-options forwarding-table export load-balancing-policy
set protocols bgp group underlay type external
set protocols bgp group underlay export send-direct
set protocols bgp group underlay local-as 65013

```

```

set protocols bgp group underlay multipath multiple-as
set protocols bgp group underlay neighbor 10.1.13.1 peer-as 65001
set protocols bgp group underlay neighbor 10.1.23.1 peer-as 65002
set protocols bgp group evpn type internal
set protocols bgp group evpn local-address 10.1.255.13
set protocols bgp group evpn family evpn signaling
set protocols bgp group evpn multipath
set protocols bgp group evpn neighbor 10.1.255.1
set protocols bgp group evpn neighbor 10.1.255.2
set protocols evpn encapsulation vxlan
set protocols evpn multicast-mode ingress-replication
set protocols evpn vni-options vni 101 vrf-target target:65000:101
set protocols evpn extended-vni-list 101
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 10.1.255.13:1
set switch-options vrf-target target:65000:1
set switch-options vrf-target auto
set vlans v101 vlan-id 101
set vlans v101 vxlan vni 101

```

Leaf 4: Full Configuration

IN THIS SECTION

- [CLI Quick Configuration | 553](#)

CLI Quick Configuration

Leaf4's configuration is similar to Leaf1 so we are providing the full configuration instead of step-by-step configuration. To quickly configure this example, copy the following commands, paste the commands into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [edit] hierarchy level, and then enter `commit` from configuration mode.

```

set interfaces xe-0/0/0 unit 0 family inet address 10.1.14.2/30
set interfaces xe-0/0/1 unit 0 family inet address 10.1.24.2/30
set interfaces xe-0/0/2 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/2 unit 0 family ethernet-switching vlan members 103

```

```

set interfaces lo0 unit 0 family inet address 10.1.255.14/32 primary
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options router-id 10.1.255.14
set routing-options autonomous-system 65000
set routing-options forwarding-table export load-balancing-policy
set protocols bgp group underlay type external
set protocols bgp group underlay export send-direct
set protocols bgp group underlay local-as 65014
set protocols bgp group underlay multipath multiple-as
set protocols bgp group underlay neighbor 10.1.14.1 peer-as 65001
set protocols bgp group underlay neighbor 10.1.24.1 peer-as 65002
set protocols bgp group evpn type internal
set protocols bgp group evpn local-address 10.1.255.14
set protocols bgp group evpn family evpn signaling
set protocols bgp group evpn multipath
set protocols bgp group evpn neighbor 10.1.255.1
set protocols bgp group evpn neighbor 10.1.255.2
set protocols evpn encapsulation vxlan
set protocols evpn multicast-mode ingress-replication
set protocols evpn vni-options vni 103 vrf-target target:65000:103
set protocols evpn extended-vni-list 103
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 10.1.255.14:1
set switch-options vrf-target target:65000:1
set switch-options vrf-target auto
set vlans v103 vlan-id 103
set vlans v103 vxlan vni 103

```

Verification

IN THIS SECTION

- [Verifying the Configuration of the IRB Interfaces | 555](#)
- [Verifying the Routing Instances | 557](#)
- [Verifying Dynamic MAC Addresses Learning | 558](#)
- [Verifying Routes in the Routing Instances | 559](#)
- [Verifying Connectivity | 560](#)

Confirm that the IRB interfaces are working properly:

Verifying the Configuration of the IRB Interfaces

Purpose

Verify the configuration of the IRB interfaces on Spine 1 and Spine 2.

Action

From operational mode, enter the `show interfaces irb` command.

```
user@Spine1> show interfaces irb
Physical interface: irb, Enabled, Physical link is Up
  Interface index: 640, SNMP ifIndex: 505
  Type: Ethernet, Link-level type: Ethernet, MTU: 1514
  Device flags   : Present Running
  Interface flags: SNMP-Traps
  Link type      : Full-Duplex
  Link flags     : None
  Current address: 02:05:86:71:57:00, Hardware address: 02:05:86:71:57:00
  Last flapped   : Never
    Input packets : 0
    Output packets: 0

Logical interface irb.101 (Index 558) (SNMP ifIndex 583)
  Flags: Up SNMP-Traps 0x4004000 Encapsulation: ENET2
  Bandwidth: 1Gbps
  Routing Instance: default-switch Bridging Domain: v101
  Input packets : 7
  Output packets: 13
  Protocol inet, MTU: 1514
  Max nh cache: 75000, New hold nh limit: 75000, Curr nh cnt: 2, Curr new hold cnt: 0, NH drop
  cnt: 0
    Flags: Sendbcst-pkt-to-re, Is-Primary
    Addresses, Flags: Is-Default Is-Preferred Is-Primary
      Destination: 10.1.101/24, Local: 10.1.101.1, Broadcast: 10.1.101.255
      Destination: 10.1.101/24, Local: 10.1.101.254, Broadcast: 10.1.101.255

Logical interface irb.102 (Index 582) (SNMP ifIndex 584)
  Flags: Up SNMP-Traps 0x4000 Encapsulation: ENET2
  Bandwidth: 1Gbps
```

```

Routing Instance: default-switch Bridging Domain: v102
Input packets : 2
Output packets: 6
Protocol inet, MTU: 1514
Max nh cache: 75000, New hold nh limit: 75000, Curr nh cnt: 1, Curr new hold cnt: 0, NH drop
cnt: 0
  Flags: Sendbroadcast-pkt-to-re
  Addresses, Flags: Is-Preferred Is-Primary
    Destination: 10.1.102/24, Local: 10.1.102.1, Broadcast: 10.1.102.255
    Destination: 10.1.102/24, Local: 10.1.102.254, Broadcast: 10.1.102.255

Logical interface irb.103 (Index 580) (SNMP ifIndex 585)
Flags: Up SNMP-Traps 0x4000 Encapsulation: ENET2
Bandwidth: 1Gbps
Routing Instance: default-switch Bridging Domain: v103
Input packets : 2
Output packets: 6
Protocol inet, MTU: 1514
Max nh cache: 75000, New hold nh limit: 75000, Curr nh cnt: 1, Curr new hold cnt: 0, NH drop
cnt: 0
  Flags: Sendbroadcast-pkt-to-re, Is-Primary
  Addresses, Flags: Is-Default Is-Preferred Is-Primary
    Destination: 10.1.103/24, Local: 10.1.103.1, Broadcast: 10.1.103.255
    Destination: 10.1.103/24, Local: 10.1.103.254, Broadcast: 10.1.103.255

```

Meaning

The sample output from Spine1 verifies the following:

- IRB interfaces irb.101, irb.102, and irb.103 are configured.
- The physical interface upon which the IRB interfaces are configured is up and running.
- Each IRB interface is properly mapped to its respective VLAN.
- The configuration of each IRB interface correctly reflects the IP address and destination (virtual gateway address) assigned to it.

Verifying the Routing Instances

Purpose

Verify that the routing instances for servers A and B, and for servers C and D, are properly configured on Spine 1 and Spine 2.

Action

From operational mode, enter the `show route instance routing-instance-name extensive` command routing instances serversAC and serversBD.

```
user@Spine1> show route instance serverAC extensive
serverAC:
  Router ID: 10.1.101.1
  Type: vrf                State: Active
  Interfaces:
    irb.101
  Route-distinguisher: 10.1.255.1:12
  Vrf-import: [ __vrf-import-serverAC-internal__ ]
  Vrf-export: [ __vrf-export-serverAC-internal__ ]
  Vrf-import-target: [ target:65000:12 ]
  Vrf-export-target: [ target:65000:12 ]
  Fast-reroute-priority: low
  Tables:
    serverAC.inet.0      : 3 routes (3 active, 0 holddown, 0 hidden)
    serverAC.iso.0       : 0 routes (0 active, 0 holddown, 0 hidden)
    serverAC.inet6.0     : 1 routes (1 active, 0 holddown, 0 hidden)
    serverAC.mdt.0       : 0 routes (0 active, 0 holddown, 0 hidden)
```

```
user@Spine1> show route instance serverBD extensive
serverBD:
  Router ID: 10.1.102.1
  Type: vrf                State: Active
  Interfaces:
    irb.102
    irb.103
  Route-distinguisher: 10.1.255.1:34
  Vrf-import: [ __vrf-import-serverBD-internal__ ]
  Vrf-export: [ __vrf-export-serverBD-internal__ ]
```



```

Vrf-import-target: [ target:65000:34 ]
Vrf-export-target: [ target:65000:34 ]
Fast-reroute-priority: low
Tables:
  serverBD.inet.0      : 6 routes (6 active, 0 holddown, 0 hidden)
  serverBD.iso.0       : 0 routes (0 active, 0 holddown, 0 hidden)
  serverBD.inet6.0     : 1 routes (1 active, 0 holddown, 0 hidden)
  serverBD.mdt.0       : 0 routes (0 active, 0 holddown, 0 hidden)

```

Meaning

In the sample output from Spine1, the routing instances for servers A and C and for servers B and D shows the loopback interface and IRB interfaces that are associated with each group. The output also shows the actual route distinguisher and import and export policy configurations.

Verifying Dynamic MAC Addresses Learning

Purpose

Verify that for VLANs v101, v102, and v103, a dynamic MAC address is installed in the Ethernet switching tables on all leaves.

Action

From operational mode, enter the `show ethernet-switching table` command.

```
user@Leaf1> show ethernet-switching table
```

```

MAC flags (S - static MAC, D - dynamic MAC, L - locally learned, P - Persistent static
          SE - statistics enabled, NM - non configured MAC, R - remote PE MAC, O - ovsdb MAC)

```

```
Ethernet switching table : 5 entries, 5 learned
```

```
Routing instance : default-switch
```

Vlan name	MAC address	MAC flags	Logical interface	SVLBNH/ VENH Index	Active source
v101	00:00:5e:00:01:01	DR	esi.1746		
	05:00:00:fd:e8:00:00:00:65:00				
v101	00:50:56:93:87:58	D	xe-0/0/2.0		
v101	00:50:56:93:ab:f6	D	vtep.32770		
10.1.255.13					

v101	02:05:86:71:27:00	D	vtep.32771
10.1.255.1			
v101	02:05:86:71:5f:00	D	vtep.32769
10.1.255.2			

Meaning

The sample output from Leaf 1 indicates that it has learned the MAC address 00:00:5e:00:01:01 for its virtual gateway (IRB). This is the MAC address that the attached servers use to reach their default gateway. Because the same virtual IP/MAC is configured on both spines, the virtual IP is treated as an ESI LAG to support active forwarding to both spines without the risk of packet loops. The output also indicates that Leaf 1 learned the IRB MAC addresses for Spine 1 and Spine 2, which function as virtual tunnel endpoints (VTEPs).

Verifying Routes in the Routing Instances

Purpose

Verify the correct routes are in the routing instances.

Action

From operational mode, enter the `show route table routing-instance-name.inet.0` command.

```
user@Spine1> show route table serverAC.inet.0

serverAC.inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.1.101.0/24      *[Direct/0] 2d 01:34:44
                  > via irb.101
10.1.101.1/32     *[Local/0] 2d 01:34:44
                  Local via irb.101
10.1.101.254/32   *[Local/0] 2d 01:34:44
                  Local via irb.101

user@Spine1> show route table serverBD.inet.0

serverBD.inet.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```

10.1.102.0/24      *[Direct/0] 2d 01:34:51
                   > via irb.102
10.1.102.1/32     *[Local/0] 2d 01:34:51
                   Local via irb.102
10.1.102.254/32   *[Local/0] 2d 01:34:51
                   Local via irb.102
10.1.103.0/24     *[Direct/0] 2d 01:34:51
                   > via irb.103
10.1.103.1/32     *[Local/0] 2d 01:34:51
                   Local via irb.103
10.1.103.254/32   *[Local/0] 2d 01:34:51
                   Local via irb.103

```

Meaning

The sample output from Spine 1 indicates that routing instance for servers A and C has the IRB interface routes associated with VLAN 101 and routing instance for server B and D has the IRB interfaces routes associated with VLANs 102 and 103.

Based on the routes in each table it is clear that servers A and C in VLAN 101 are not able to reach servers in C and D in VLANs 102 or 103. Also clear from the output is that the common table that houses routes for servers B and D allows Layer 3 communications through their IRB interfaces.

Verifying Connectivity

Purpose

Verify that servers A and C can ping each other and servers B and D can ping each other.

Action

Run the ping command from the servers.

```

user@serverA> ping 10.1.101.103 count 2
PING 10.1.101.103 (10.1.101.103): 56 data bytes
64 bytes from 10.1.101.103: icmp_seq=0 ttl=64 time=103.749 ms
64 bytes from 10.1.101.103: icmp_seq=1 ttl=64 time=116.325 ms

--- 10.1.101.103 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss

```

```

round-trip min/avg/max/stddev = 103.749/110.037/116.325/6.288 ms

user@serverB> ping 10.1.103.101 count 2
PING 10.1.103.101 (10.1.103.101): 56 data bytes
64 bytes from 10.1.103.101: icmp_seq=0 ttl=63 time=103.346 ms
64 bytes from 10.1.103.101: icmp_seq=1 ttl=63 time=102.355 ms

--- 10.1.103.101 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 102.355/102.851/103.346/0.495 ms

```

Meaning

The sample output shows server A can ping server C, and server B can ping server D. Servers A and C should not be able to ping servers B and D, and servers B and D should not be able to ping servers A and C.

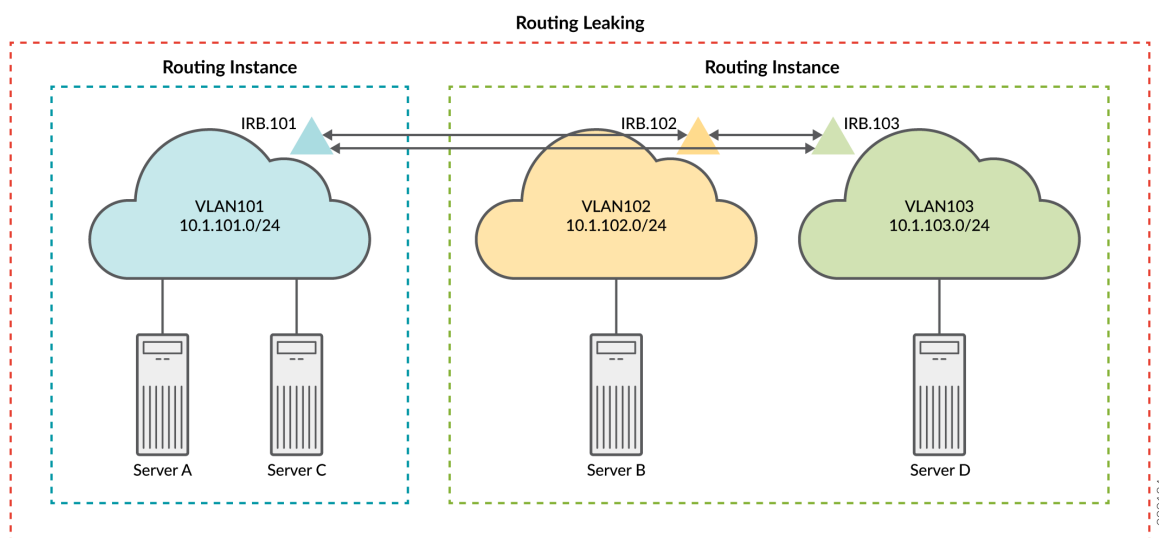
Spine 1 and 2: Route Leaking (Optional)

IN THIS SECTION

- [CLI Quick Configuration | 562](#)

Figure 48: EVPN-VXLAN Logical Topology with Route Leaking

Referring back to [Figure 47 on page 533](#), recall that you configured three VLANs and two routing instances to provide connectivity for servers A and C in VLAN 101, and for servers B and D in VLANs 102 and 103. In this section you modify the configuration to leak routes between the two routing instances. shows the resulting logical connectivity. Once the IRB routes are leaked the servers in VLAN 101 are expected to be able to reach the servers in both VLANs 102 and 103 using Layer 3 connectivity.



CLI Quick Configuration

At this stage you have deployed a CRB based EVPN fabric and have confirmed expected connectivity. That is, servers A and C can communicate at Layer 2, while servers B and D, on VLANs 102 and 103 respectively, communicate through IRB routing in their shared routing instance. What if you want all servers to be able to ping each other? One option to solve this problem is to leak the routes between the routing instances. See [auto-export](#) for more information on leaking routes between VRFs. To quickly configure this example, copy the following commands, paste the commands into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [edit] hierarchy level, and then enter `commit` from configuration mode.

```
set policy-options policy-statement serverAC_vrf_imp term 1 from community serverBD
set policy-options policy-statement serverAC_vrf_imp term 1 then accept
set policy-options policy-statement serverBD_vrf_imp term 1 from community serverAC
set policy-options policy-statement serverBD_vrf_imp term 1 then accept
set policy-options community serverAC members target:65000:13
set policy-options community serverBD members target:65000:24
set routing-instances serverAC routing-options auto-export
set routing-instances serverAC vrf-import serverAC_vrf_imp
set routing-instances serverBD routing-options auto-export
set routing-instances serverBD vrf-import serverBD_vrf_imp
```

Verification with Route Leaking (Optional)

IN THIS SECTION

- [Verifying Routes with Route Leaking \(Optional\) | 563](#)
- [Verifying Connectivity with Route Leaking \(Optional\) | 564](#)

Verifying Routes with Route Leaking (Optional)

Purpose

Verify the correct routes are in the routing instances.

Action

From operational mode, enter the `show route table routing-instance-name.inet.0` command.

```
user@Spine1> show route table serverAC.inet.0

serverAC.inet.0: 9 destinations, 9 routes (9 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.1.101.0/24      *[Direct/0] 2d 02:18:50
                   > via irb.101
10.1.101.1/32     *[Local/0] 2d 02:18:50
                   Local via irb.101
10.1.101.254/32   *[Local/0] 2d 02:18:50
                   Local via irb.101
10.1.102.0/24     *[Direct/0] 00:31:21
                   > via irb.102
10.1.102.1/32     *[Local/0] 00:31:21
                   Local via irb.102
10.1.102.254/32   *[Local/0] 00:31:21
                   Local via irb.102
10.1.103.0/24     *[Direct/0] 00:31:21
                   > via irb.103
10.1.103.1/32     *[Local/0] 00:31:21
                   Local via irb.103
```

```

10.1.103.254/32    *[Local/0] 00:31:21
                  Local via irb.103

user@Spine1> show route table serverBD.inet.0

serverBD.inet.0: 9 destinations, 9 routes (9 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.1.101.0/24      *[Direct/0] 00:32:00
                  > via irb.101
10.1.101.1/32      *[Local/0] 00:32:00
                  Local via irb.101
10.1.101.254/32    *[Local/0] 00:32:00
                  Local via irb.101
10.1.102.0/24      *[Direct/0] 2d 02:19:29
                  > via irb.102
10.1.102.1/32      *[Local/0] 2d 02:19:29
                  Local via irb.102
10.1.102.254/32    *[Local/0] 2d 02:19:29
                  Local via irb.102
10.1.103.0/24      *[Direct/0] 2d 02:19:29
                  > via irb.103
10.1.103.1/32      *[Local/0] 2d 02:19:29
                  Local via irb.103
10.1.103.254/32    *[Local/0] 2d 02:19:29
                  Local via irb.103

```

Meaning

The sample output from Spine 1 indicates that both routing instances now have the IRB interface routes associated with all three VLANs. Because you copied routes between the instance tables the net result is the same as if you configured all three VLANs in a common routing instance. Thus, you can expect full Layer 3 connectivity between servers in all three VLANs.

Verifying Connectivity with Route Leaking (Optional)

Purpose

Verify that servers A and C can ping servers B and D.

Action

Run the ping command from the servers.

```
user@serverA> ping 10.1.102.101 count 2
PING 10.1.102.101 (10.1.102.101): 56 data bytes
64 bytes from 10.1.102.101: icmp_seq=0 ttl=63 time=102.448 ms
64 bytes from 10.1.102.101: icmp_seq=1 ttl=63 time=102.384 ms

--- 10.1.102.101 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 102.384/102.416/102.448/0.032 ms

user@serverA> ping 10.1.103.101 count 2
PING 10.1.103.101 (10.1.103.101): 56 data bytes
64 bytes from 10.1.103.101: icmp_seq=0 ttl=63 time=103.388 ms
64 bytes from 10.1.103.101: icmp_seq=1 ttl=63 time=102.623 ms

--- 10.1.103.101 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 102.623/103.006/103.388/0.382 ms

user@serverC> ping 10.1.102.101 count 2
PING 10.1.102.101 (10.1.102.101): 56 data bytes
64 bytes from 10.1.102.101: icmp_seq=0 ttl=63 time=167.580 ms
64 bytes from 10.1.102.101: icmp_seq=1 ttl=63 time=168.075 ms

--- 10.1.102.101 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 167.580/167.827/168.075/0.248 ms

user@serverC> ping 10.1.103.101 count 2
PING 10.1.103.101 (10.1.103.101): 56 data bytes
64 bytes from 10.1.103.101: icmp_seq=0 ttl=63 time=103.673 ms
64 bytes from 10.1.103.101: icmp_seq=1 ttl=63 time=115.090 ms

--- 10.1.103.101 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 103.673/109.382/115.090/5.709 ms
```


Meaning

The sample output shows server A can ping server B and server D, and server C can ping server B and server D.

Release History Table

Release	Description
15.1X53-D30	Starting with Junos OS Release 15.1X53-D30, you can use integrated routing and bridging (IRB) interfaces to route between VLANs using Virtual Extensible LAN (VXLAN) encapsulation.

Example: Configuring a QFX5110 Switch as a Layer 3 VXLAN Gateway in an EVPN-VXLAN Centrally-Routed Bridging Overlay

IN THIS SECTION

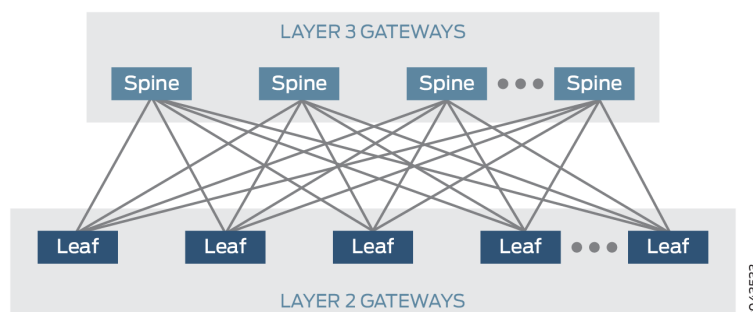
- [Requirements | 567](#)
- [Overview and Topology | 568](#)
- [Basic Underlay Network Configuration | 571](#)
- [Basic EVPN-VXLAN Overlay Network Configuration | 572](#)
- [Basic Customer Profile Configuration | 574](#)
- [Route Leaking Configuration | 577](#)

Ethernet VPN (EVPN) is a control plane technology that enables hosts (physical [bare-metal] servers and virtual machines [VMs]) to be placed anywhere in a network and remain connected to the same logical Layer 2 overlay network. Virtual Extensible LAN (VXLAN) is a tunneling protocol that creates the data plane for the Layer 2 overlay network.

The physical underlay network over which EVPN-VXLAN is commonly deployed is a two-layer IP fabric, which includes spine and leaf devices as shown in [Figure 49 on page 567](#). In the underlay network, the

spine devices provide connectivity between the leaf devices, and the leaf devices provide connectivity to the attached physical servers and VMs on virtualized servers.

Figure 49: Two-Layer IP Fabric



In an EVPN-VXLAN centrally-routed bridging overlay (EVPN-VXLAN topology with a two-layer IP fabric), the leaf devices function as Layer 2 VXLAN gateways that handle traffic within a VLAN, and the spine devices function as Layer 3 VXLAN gateways that handle traffic between VLANs using integrated routing and bridging (IRB) interfaces.

Prior to Junos OS Release 17.3R1, a QFX5110 switch can function only as a Layer 2 VXLAN gateway in a centrally-routed bridging overlay, all of which is deployed within a data center. Starting with Junos OS Release 17.3R1, the QFX5110 switch can also function as a Layer 3 VXLAN gateway in a centrally-routed bridging overlay.

This topic provides a sample configuration of a QFX5110 switch that functions as a spine device or Layer 3 VXLAN gateway in a centrally-routed bridging overlay. This example shows how to configure Layer 3 VXLAN gateways with IRB interfaces and default gateways.

Requirements

This example uses the following hardware and software components:

- Two QFX5110 switches that function as spine devices (spine 1 and spine 2). These devices provide Layer 3 VXLAN gateway functionality.

NOTE: This example focuses on the configuration of the QFX5110 switch that functions as spine 1. For spine 1, a basic configuration is provided for the IP/BGP underlay network, the EVPN-VXLAN overlay network, customer-specific profiles, and route leaking. This example does not include all features that can be used in an EVPN-VXLAN network. The configuration for spine 1 essentially serves as a template for the configuration of spine 2. For the

configuration of spine 2, where appropriate, you can replace spine 1-specific information with the information specific to spine 2, add additional commands, and so on.

- Two QFX5200 switches that function as leaf devices (leaf 1 and leaf 2). These devices provide Layer 2 VXLAN gateway functionality.
- Junos OS Release 17.3R1 or later software running on the QFX5110 and QFX5200 switches.
- Physical servers in VLAN v100, and a physical server and a virtualized servers on which VMs are installed in VLAN v200.

Overview and Topology

In this example, a service provider supports ABC Corporation, which has multiple sites. Physical servers in site 100 must communicate with a physical servers and VMs in site 200. To enable this communication in the centrally-routed bridging overlay shown in [Figure 50 on page 569](#), on the

QFX5110 switches that function as Layer 3 VXLAN gateways, or spine devices, you configure the key software entities shown in [Table 28 on page 569](#).

Figure 50: Centrally-Routed Bridging Overlay

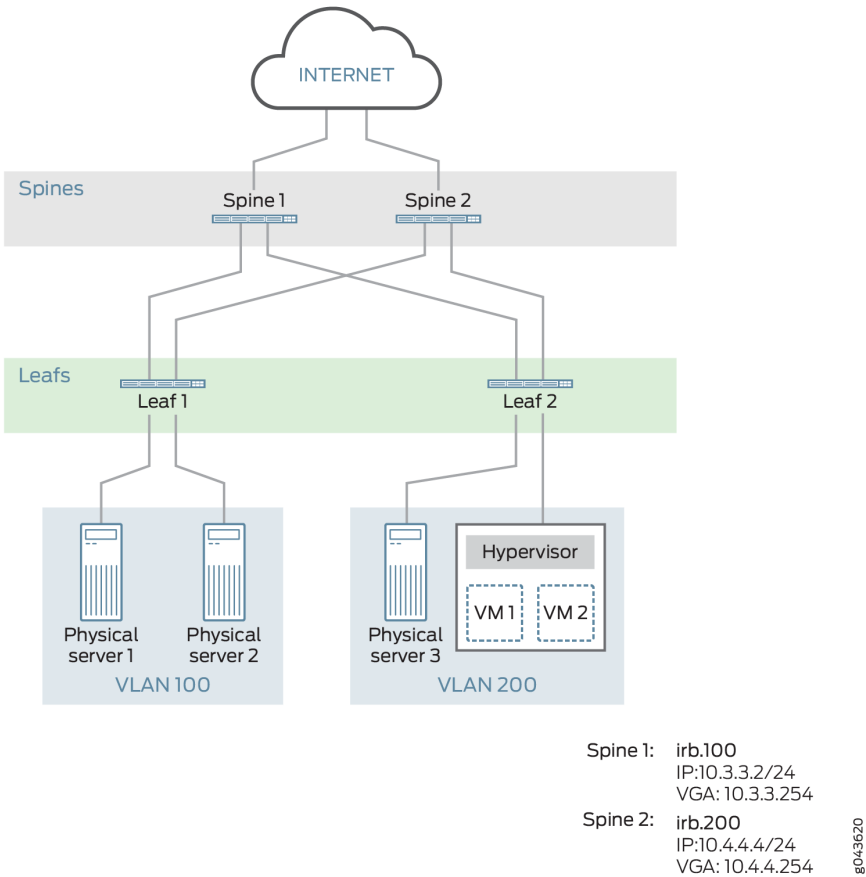


Table 28: Layer 3 Inter-VLAN Routing Entities Configured on Spine 1 and Spine 2

Entity	Configuration on Spine 1 and Spine 2
VLANs	v100
	v200
VRF instances	vrf_vlan100
	vrf_vlan200

Table 28: Layer 3 Inter-VLAN Routing Entities Configured on Spine 1 and Spine 2 (Continued)

Entity	Configuration on Spine 1 and Spine 2
IRB interfaces	irb.100
	10.3.3.2/24 (IRB IP address)
	10.3.3.254 (virtual gateway address)
	irb.200
	10.4.4.4/24 (IRB IP address)
	10.4.4.254 (virtual gateway address)

As outlined in [Table 28 on page 569](#), on both spine devices, you configure VLAN v100 for site 100 and VLAN v200 for site 200. To segregate the Layer 3 routes associated with VLANs v100 and v200, you create VPN routing and forwarding (VRF) instances vrf_vlan100 and vrf_vlan200 on both spine devices. To route traffic between the VLANs, you configure IRB interfaces irb.100 and irb.200 on both spine devices, and associate VRF routing instance vrf_vlan100 with IRB interface irb.100, and VRF routing instance vrf_vlan200 with IRB interface irb.200.

NOTE: QFX5110 switches do not support the configuration of an IRB interface with a unique MAC address.

The physical servers in VLANs v100 and v200 are non-virtualized. As a result, we strongly recommend that you configure IRB interfaces irb.100 and irb.200 to function as default Layer 3 gateways that handle the inter-VLAN traffic of the physical servers. To that end, the configuration of each IRB interface also includes a virtual gateway address (VGA), which configures each IRB interface as a default gateway. In addition, this example assumes that each physical server is configured to use a particular default gateway. For general information about default gateways and how inter-VLAN traffic flows between a physical server to another physical server or VM in a different VLAN in a centrally-routed bridging overlay, see ["Using a Default Layer 3 Gateway to Route Traffic in an EVPN-VXLAN Overlay Network" on page 508](#).

NOTE: When configuring a VGA for an IRB interface, keep in mind that the IRB IP address and VGA must be different.

NOTE: If a QFX5110 switch running Junos OS Release 17.3R1 or later software functions as both a Layer 3 VXLAN gateway and a Dynamic Host Configuration Protocol (DHCP) relay in an EVPN-VXLAN topology, the DHCP server response time for an IP address might take up to a few minutes. The lengthy response time might occur if a DHCP client receives and later releases an IP address on an EVPN-VXLAN IRB interface configured on the QFX5110 switch and the binding between the DHCP client and the IP address is not deleted.

As outlined in [Table 28 on page 569](#), a separate VRF routing instance is configured for each VLAN. To enable communication between the hosts in VLANs v100 and v200, this example shows how to export unicast routes from the routing table for routing instance vrf_vlan100 and import the routes into the routing table for vrf_vlan200 and vice versa. This feature is also known as route leaking.

Basic Underlay Network Configuration

IN THIS SECTION

- [CLI Quick Configuration | 571](#)
- [Configuring a Basic Underlay Network | 572](#)

CLI Quick Configuration

To quickly configure a basic underlay network, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

```
set interfaces et-0/0/0 unit 0 family inet address 10.1.1.1/24
set interfaces et-0/0/1 unit 0 family inet address 10.2.2.1/24
set routing-options router-id 10.2.3.11
set routing-options autonomous-system 64500
set protocols bgp group pe neighbor 10.2.3.12
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface et-0/0/0.0
set protocols ospf area 0.0.0.0 interface et-0/0/1.0
```

Configuring a Basic Underlay Network

Step-by-Step Procedure

To configure a basic underlay network on spine 1:

1. Configure the interfaces that connect to the leaf devices.

```
[edit interfaces]
user@switch# set et-0/0/0 unit 0 family inet address 10.1.1.1/24
user@switch# set et-0/0/1 unit 0 family inet address 10.2.2.1/24
```

2. Configure the router ID and autonomous system number for spine 1.

```
[edit routing-options]
user@switch# set router-id 10.2.3.11
user@switch# set autonomous-system 64500
```

3. Configure a BGP group that includes spine 2 as a peer that also handles underlay functions.

```
[edit protocols]
user@switch# set bgp group pe neighbor 10.2.3.12
```

4. Configure OSPF as the routing protocol for the underlay network.

```
[edit protocols]
user@switch# set ospf area 0.0.0.0 interface lo0.0 passive
user@switch# set ospf area 0.0.0.0 interface et-0/0/0.0
user@switch# set ospf area 0.0.0.0 interface et-0/0/1.0
```

Basic EVPN-VXLAN Overlay Network Configuration

IN THIS SECTION

- [CLI Quick Configuration | 573](#)
- [Configuring a Basic EVPN-VXLAN Overlay Network | 573](#)

CLI Quick Configuration

To quickly configure a basic overlay network, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

```
set forwarding-options vxlan-routing interface-num 8192
set forwarding-options vxlan-routing next-hop 16384
set protocols bgp group pe type internal
set protocols bgp group pe local-address 10.2.3.11
set protocols bgp group pe family evpn signaling
set protocols evpn encapsulation vxlan
set protocols evpn extended-vni-list all
set protocols evpn default-gateway no-gateway-community
set switch-options route-distinguisher 10.2.3.11:1
set switch-options vrf-target target:1111:11
set switch-options vtep-source-interface lo0.0
```

Configuring a Basic EVPN-VXLAN Overlay Network

Step-by-Step Procedure

To configure a basic EVPN-VXLAN overlay network on spine 1:

1. Increase the number of physical interfaces and next hops that the QFX5110 switch allocates for use in an EVPN-VXLAN overlay network.

```
[edit forwarding-options]
set vxlan-routing interface-num 8192
set vxlan-routing next-hop 16384
```

2. Configure an IBGP overlay between spine 1 and the connected leaf devices, specify a local IP address for spine 1, and include the EVPN signaling Network Layer Reachability Information (NLRI) to the pe BGP group.

```
[edit protocols]
user@switch# set bgp group pe type internal
user@switch# set bgp group pe local-address 10.2.3.11
user@switch# set bgp group pe family evpn signaling
```


3. Configure VXLAN encapsulation for the data packets exchanged between the EVPN neighbors, and specify that all VXLAN network identifiers (VNIs) are part of the virtual routing and forwarding (VRF) instance. Also, specify that the MAC address of the IRB interface and the MAC address of the corresponding default gateway are advertised to the Layer 2 VXLAN gateways without the extended community option of default -gateway.

```
[edit protocols]
user@switch# set evpn encapsulation vxlan
user@switch# set evpn extended-vni-list all
user@switch# set evpn default-gateway no-gateway-community
```

4. Configure switch options to set a route distinguisher and VRF target for the VRF routing instance, and associate interface lo0 with the virtual tunnel endpoint (VTEP).

```
[edit switch-options]
user@switch# set route-distinguisher 10.2.3.11:1
user@switch# set vrf-target target:1111:11
user@switch# set vtep-source-interface lo0.0
```

Basic Customer Profile Configuration

IN THIS SECTION

- [CLI Quick Configuration | 574](#)
- [Configuring a Basic Customer Profile | 575](#)

CLI Quick Configuration

To quickly configure a basic customer profile for ABC Corporation sites 100 and 200, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

```
set interfaces xe-0/0/32:1 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/32:1 unit 0 family ethernet-switching vlan members v100
set interfaces xe-0/0/32:1 unit 0 family ethernet-switching vlan members v200
set interfaces irb unit 100 family inet address 10.3.3.2/24 virtual-gateway-address 10.3.3.254
```

```

set interfaces irb unit 100 proxy-macip-advertisement
set interfaces irb unit 200 family inet address 10.4.4.4/24 virtual-gateway-address 10.4.4.254
set interfaces irb unit 200 proxy-macip-advertisement
set interfaces lo0 unit 0 family inet address 10.2.3.11/32 primary
set interfaces lo0 unit 1 family inet address 10.2.3.24/32 primary
set interfaces lo0 unit 2 family inet address 10.2.3.25/32 primary
set routing-instances vrf_vlan100 instance-type vrf
set routing-instances vrf_vlan100 interface irb.100
set routing-instances vrf_vlan100 interface lo0.1
set routing-instances vrf_vlan100 route-distinguisher 10.2.3.11:2
set routing-instances vrf_vlan100 vrf-import import-inet
set routing-instances vrf_vlan100 vrf-export export-inet1
set routing-instances vrf_vlan200 instance-type vrf
set routing-instances vrf_vlan200 interface irb.200
set routing-instances vrf_vlan200 interface lo0.2
set routing-instances vrf_vlan200 route-distinguisher 10.2.3.11:3
set routing-instances vrf_vlan200 vrf-import import-inet
set routing-instances vrf_vlan200 vrf-export export-inet2
set vlans v100 vlan-id 100
set vlans v100 l3-interface irb.100
set vlans v100 vxlan vni 100
set vlans v200 vlan-id 200
set vlans v200 l3-interface irb.200
set vlans v200 vxlan vni 200

```

Configuring a Basic Customer Profile

Step-by-Step Procedure

To configure a basic customer profile for ABC Corporation sites 100 and 200 on spine 1:

1. Configure a Layer 2 interface, and specify the interface as a member of VLANs v100 and v200.

```

[edit interfaces]
user@switch# set xe-0/0/32:1 unit 0 family ethernet-switching interface-mode trunk
user@switch# set xe-0/0/32:1 unit 0 family ethernet-switching vlan members v100
user@switch# set xe-0/0/32:1 unit 0 family ethernet-switching vlan members v200

```

2. Create IRB interfaces, and configure the interfaces to act as default Layer 3 virtual gateways, which route traffic from physical servers in VLAN v100 to physical servers and VMs in VLAN v200 and vice

versa. Also, on the IRB interfaces, enable the Layer 3 VXLAN gateway to advertise MAC+IP type 2 routes on behalf of the Layer 2 VXLAN gateways.

```
[edit interfaces]
user@switch# set irb unit 100 family inet address 10.3.3.2/24 virtual-gateway-address
10.3.3.254
user@switch# set irb unit 100 proxy-macip-advertisement
user@switch# set irb unit 200 family inet address 10.4.4.4/24 virtual-gateway-address
10.4.4.254
user@switch# set irb unit 200 proxy-macip-advertisement
```

NOTE: QFX5110 switches do not support the configuration of an IRB interface with a unique MAC address.

NOTE: When configuring a VGA for an IRB interface, keep in mind that the VGA and IRB IP address must be different.

3. Configure a loopback interface (lo0) for spine 1 and a logical loopback address (lo0.x) for each VRF routing instance.

```
[edit interfaces]
user@switch# set lo0 unit 0 family inet address 10.2.3.11/32 primary
user@switch# set lo0 unit 1 family inet address 10.2.3.24/32 primary
user@switch# set lo0 unit 2 family inet address 10.2.3.25/32 primary
```

4. Configure VRF routing instances for VLANs v100 and v200. In each routing instance, associate an IRB interface, a loopback interface, and an identifier attached to the route. Also specify that each routing instance exports its overlay routes to the VRF table for the other routing instance and imports overlay routes from the VRF table for the other routing instance into its VRF table.

```
[edit routing-instances]
user@switch# set vrf_vlan100 instance-type vrf
user@switch# set vrf_vlan100 interface irb.100
user@switch# set vrf_vlan100 interface lo0.1
user@switch# set vrf_vlan100 route-distinguisher 10.2.3.11:2
user@switch# set vrf_vlan100 vrf-import import-inet
user@switch# set vrf_vlan100 vrf-export export-inet1
```

```

user@switch# set vrf_vlan200 instance-type vrf
user@switch# set vrf_vlan200 interface irb.200
user@switch# set vrf_vlan200 interface lo0.2
user@switch# set vrf_vlan200 route-distinguisher 10.2.3.11:3
user@switch# set vrf_vlan200 vrf-import import-inet
user@switch# set vrf_vlan200 vrf-export export-inet2

```

5. Configure VLANs v100 and v200, and associate an IRB interface and VNI with each VLAN.

```

[edit vlans]
user@switch# set v100 vlan-id 100
user@switch# set v100 l3-interface irb.100
user@switch# set v100 vxlan vni 100
user@switch# set v200 vlan-id 200
user@switch# set v200 l3-interface irb.200
user@switch# set v200 vxlan vni 200

```

Route Leaking Configuration

IN THIS SECTION

- [Procedure | 577](#)

Procedure

CLI Quick Configuration

To quickly configure route leaking, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

```

set policy-options policy-statement export-inet1 term 1 from interface irb.100
set policy-options policy-statement export-inet1 term 1 then community add com200
set policy-options policy-statement export-inet1 term 1 then accept
set policy-options policy-statement export-inet2 term 1 from interface irb.200
set policy-options policy-statement export-inet2 term 1 then community add com100
set policy-options policy-statement export-inet2 term 1 then accept

```

```

set policy-options policy-statement import-inet term 1 from community com100
set policy-options policy-statement import-inet term 1 from community com200
set policy-options policy-statement import-inet term 1 then accept
set policy-options community com100 members target:1:100
set policy-options community com200 members target:1:200
set routing-instances vrf_vlan100 vrf-import import-inet
set routing-instances vrf_vlan100 vrf-export export-inet1
set routing-instances vrf_vlan100 routing-options auto-export family inet unicast
set routing-instances vrf_vlan200 vrf-import import-inet
set routing-instances vrf_vlan200 vrf-export export-inet2
set routing-instances vrf_vlan200 routing-options auto-export family inet unicast

```

Step-by-Step Procedure

To configure route leaking on spine 1:

1. Configure a routing policy that specifies that routes learned through IRB interface irb.100 are exported and then imported into the routing table for vrf_vlan200. Configure another routing policy that specifies that routes learned through IRB interface irb.200 are exported and then imported into the routing table for vrf_vlan100.

```

[edit policy-options]
user@switch# set policy-statement export-inet1 term 1 from interface irb.100
user@switch# set policy-statement export-inet1 term 1 then community add com200
user@switch# set policy-statement export-inet1 term 1 then accept
user@switch# set policy-statement export-inet2 term 1 from interface irb.200
user@switch# set policy-statement export-inet2 term 1 then community add com100
user@switch# set policy-statement export-inet2 term 1 then accept
user@switch# set policy-statement import-inet term 1 from community com100
user@switch# set policy-statement import-inet term 1 from community com200
user@switch# set policy-statement import-inet term 1 then accept
user@switch# set community com100 members target:1:100
user@switch# set community com200 members target:1:200

```

2. In the VRF routing instances for VLANs v100 and v200, apply the routing policies configured in step 1.

```

[edit routing-instances]
user@switch# set vrf_vlan100 vrf-import import-inet
user@switch# set vrf_vlan100 vrf-export export-inet1

```

```
user@switch# set vrf_vlan200 vrf-import import-inet
user@switch# set vrf_vlan200 vrf-export export-inet2
```

3. Specify that unicast routes are to be exported from the vrf_vlan100 routing table into the vrf_vlan200 routing table and vice versa.

```
[edit routing-instances]
user@switch# set vrf_vlan100 routing-options auto-export family inet unicast
user@switch# set vrf_vlan200 routing-options auto-export family inet unicast
```

RELATED DOCUMENTATION

[Example: Configuring a QFX5110 Switch as a Layer 3 VXLAN Gateway in an EVPN-VXLAN Centrally-Routed Bridging Overlay | 566](#)

Configuring an EVPN-VXLAN Centrally-Routed Bridged Overlay

IN THIS CHAPTER

- [Example: Configure an EVPN-VXLAN Centrally-Routed Bridging \(CRB\) Using MX Routers as Spines | 580](#)

Example: Configure an EVPN-VXLAN Centrally-Routed Bridging (CRB) Using MX Routers as Spines

IN THIS SECTION

- [Requirements | 580](#)
- [Overview | 581](#)
- [Topology | 582](#)
- [Configuration | 582](#)
- [Verification | 601](#)

This example shows how to configure EVPN and VXLAN on an IP fabric to support optimal forwarding of Ethernet frames, provide network segmentation on a broad scale, enable control plane-based MAC learning, and many other advantages. This example is based on a Centrally-routed with bridging (CRB) EVPN architecture in a 5-stage Clos fabric.

Requirements

The original example used the following hardware and software components:

- Two Juniper Networks MX Series routers to act as IP gateways for the EVPN overlay

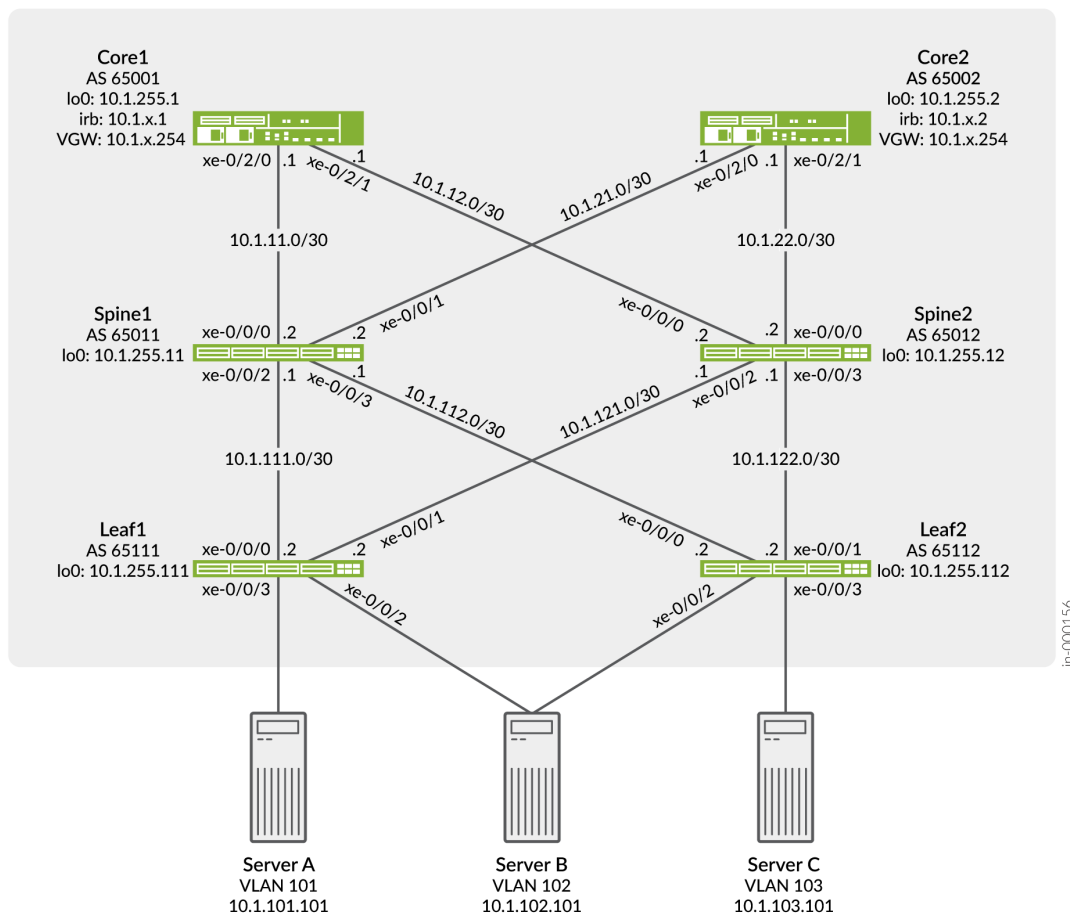
- Four Juniper Networks QFX5100 switches. Two of these switches act as PE devices in the EVPN topology, and the other two switches act as pure IP transport for the underlay.
- Junos OS Release 16.1 or later.
 - Updated and revalidated using Junos OS Release 21.3R1.9
- Starting with Junos OS Release 17.3R1, EVPN-VXLAN is also supported on EX9200 switches. Previously, only MPLS encapsulation was supported. In this example, the EX9200 switch would function as an IP gateway for the EVPN overlay. There are some configuration differences between MX Series routers and EX9200 switches. The configuration section later in this topic has more information about the configuration that is specific to an EX9200.
- See the [hardware summary](#) for a list of supported platforms.

Overview

Ethernet VPNs (EVPNs) enable you to connect groups of dispersed customer sites using Layer 2 virtual bridges, and Virtual Extensible LANs (VXLANs) enable you to stretch Layer 2 connection over an intervening Layer 3 network, while providing network segmentation like a VLAN, but without the scaling limitation of traditional VLANs. EVPN with VXLAN encapsulation handles Layer 2 connectivity at the scale required by cloud service providers, and replaces limiting protocols like STP, freeing up your Layer 3 network to use more robust routing protocols.

This example configuration shows how to configure EVPN with VXLAN encapsulation. In this example, the MX Series routers are named Core-1 and Core-2. The QFX5100 switches are named Leaf-1, Leaf-2, Spine-1, and Spine-2. The core routers act as IP gateways for the EVPN overlay, the leaf switches act as PE devices in the EVPN topology, and the spine switches act as pure IP transport for the underlay (also known as a "lean spine").

Topology



Configuration

IN THIS SECTION

- [CLI Quick Configuration | 583](#)
- [Configuring Leaf-1 | 592](#)
- [Configuring Spine-1 | 596](#)
- [Configuring Core-1 | 597](#)

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

Leaf-1

```
set system host-name leaf-1
set chassis aggregated-devices ethernet device-count 2
set interfaces xe-0/0/0 unit 0 family inet address 10.1.111.2/30
set interfaces xe-0/0/1 unit 0 family inet address 10.1.121.2/30
set interfaces xe-0/0/2 ether-options 802.3ad ae0
set interfaces xe-0/0/3 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/3 unit 0 family ethernet-switching vlan members v101
set interfaces ae0 esi 00:01:01:01:01:01:01:01:01
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp system-id 00:00:00:01:01:01
set interfaces ae0 unit 0 family ethernet-switching interface-mode access
set interfaces ae0 unit 0 family ethernet-switching vlan members v102
set interfaces lo0 unit 0 family inet address 10.1.255.111/32
set policy-options policy-statement lo0 from family inet
set policy-options policy-statement lo0 from protocol direct
set policy-options policy-statement lo0 from route-filter 0.0.0.0/0 prefix-length-range /32-/32
set policy-options policy-statement lo0 then accept
set policy-options policy-statement load-balance term 1 then load-balance per-packet
set policy-options policy-statement vrf-imp term t1 from community com101
set policy-options policy-statement vrf-imp term t1 then accept
set policy-options policy-statement vrf-imp term t2 from community com102
set policy-options policy-statement vrf-imp term t2 then accept
set policy-options policy-statement vrf-imp term t3 from community com103
set policy-options policy-statement vrf-imp term t3 then accept
set policy-options policy-statement vrf-imp term t5 then reject
set policy-options community com101 members target:65000:101
set policy-options community com102 members target:65000:102
set policy-options community com103 members target:65000:103
set routing-options router-id 10.1.255.111
set routing-options autonomous-system 65000
set routing-options forwarding-table export load-balance
set routing-options forwarding-table ecmp-fast-reroute
set protocols bgp group underlay type external
```

```

set protocols bgp group underlay export lo0
set protocols bgp group underlay local-as 65111
set protocols bgp group underlay multipath multiple-as
set protocols bgp group underlay neighbor 10.1.111.1 peer-as 65011
set protocols bgp group underlay neighbor 10.1.121.1 peer-as 65012
set protocols bgp group EVPN_VXLAN_CORE type internal
set protocols bgp group EVPN_VXLAN_CORE local-address 10.1.255.111
set protocols bgp group EVPN_VXLAN_CORE family evpn signaling
set protocols bgp group EVPN_VXLAN_CORE neighbor 10.1.255.1
set protocols bgp group EVPN_VXLAN_CORE neighbor 10.1.255.2
set protocols evpn encapsulation vxlan
set protocols evpn multicast-mode ingress-replication
set protocols evpn vni-options vni 101 vrf-target target:65000:101
set protocols evpn vni-options vni 102 vrf-target target:65000:102
set protocols evpn extended-vni-list 101
set protocols evpn extended-vni-list 102
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 10.1.255.111:1
set switch-options vrf-import vrf-imp
set switch-options vrf-target target:65000:1
set vlans v101 vlan-id 101
set vlans v101 vxlan vni 101
set vlans v102 vlan-id 102
set vlans v102 vxlan vni 102

```

Leaf-2

```

set system host-name leaf-2
set chassis aggregated-devices ethernet device-count 2
set interfaces xe-0/0/0 unit 0 family inet address 10.1.112.2/30
set interfaces xe-0/0/1 unit 0 family inet address 10.1.122.2/30
set interfaces xe-0/0/2 ether-options 802.3ad ae0
set interfaces xe-0/0/3 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/3 unit 0 family ethernet-switching vlan members v103
set interfaces ae0 esi 00:01:01:01:01:01:01:01:01
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp system-id 00:00:00:01:01:01
set interfaces ae0 unit 0 family ethernet-switching interface-mode access
set interfaces ae0 unit 0 family ethernet-switching vlan members v102
set interfaces lo0 unit 0 family inet address 10.1.255.112/32
set policy-options policy-statement lo0 from family inet

```

```
set policy-options policy-statement lo0 from protocol direct
set policy-options policy-statement lo0 from route-filter 0.0.0.0/0 prefix-length-range /32-/32
set policy-options policy-statement lo0 then accept
set policy-options policy-statement load-balance term 1 then load-balance per-packet
set policy-options policy-statement vrf-imp term t1 from community com101
set policy-options policy-statement vrf-imp term t1 then accept
set policy-options policy-statement vrf-imp term t2 from community com102
set policy-options policy-statement vrf-imp term t2 then accept
set policy-options policy-statement vrf-imp term t3 from community com103
set policy-options policy-statement vrf-imp term t3 then accept
set policy-options policy-statement vrf-imp term t5 then reject
set policy-options community com101 members target:65000:101
set policy-options community com102 members target:65000:102
set policy-options community com103 members target:65000:103
set routing-options router-id 10.1.255.112
set routing-options autonomous-system 65000
set routing-options forwarding-table export load-balance
set routing-options forwarding-table ecmp-fast-reroute
set protocols bgp group underlay type external
set protocols bgp group underlay export lo0
set protocols bgp group underlay local-as 65112
set protocols bgp group underlay multipath multiple-as
set protocols bgp group underlay neighbor 10.1.112.1 peer-as 65011
set protocols bgp group underlay neighbor 10.1.122.1 peer-as 65012
set protocols bgp group EVPN_VXLAN_CORE type internal
set protocols bgp group EVPN_VXLAN_CORE local-address 10.1.255.112
set protocols bgp group EVPN_VXLAN_CORE family evpn signaling
set protocols bgp group EVPN_VXLAN_CORE neighbor 10.1.255.1
set protocols bgp group EVPN_VXLAN_CORE neighbor 10.1.255.2
set protocols evpn encapsulation vxlan
set protocols evpn multicast-mode ingress-replication
set protocols evpn vni-options vni 102 vrf-target target:65000:102
set protocols evpn vni-options vni 103 vrf-target target:65000:103
set protocols evpn extended-vni-list 102
set protocols evpn extended-vni-list 103
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 10.1.255.112:1
set switch-options vrf-import vrf-imp
set switch-options vrf-target target:65000:1
set vlans v102 vlan-id 102
set vlans v102 vxlan vni 102
```

```
set vlans v103 vlan-id 103
set vlans v103 vxlan vni 103
```

Spine-1

```
set system host-name spine-1
set interfaces xe-0/0/0 unit 0 family inet address 10.1.11.2/30
set interfaces xe-0/0/1 unit 0 family inet address 10.1.21.2/30
set interfaces xe-0/0/2 unit 0 family inet address 10.1.111.1/30
set interfaces xe-0/0/3 unit 0 family inet address 10.1.112.1/30
set interfaces lo0 unit 0 family inet address 10.1.255.11/32
set policy-options policy-statement lo0 from family inet
set policy-options policy-statement lo0 from protocol direct
set policy-options policy-statement lo0 from route-filter 0.0.0.0/0 prefix-length-range /32-/32
set policy-options policy-statement lo0 then accept
set policy-options policy-statement load-balance term 1 then load-balance per-packet
set routing-options router-id 10.1.255.11
set routing-options autonomous-system 65000
set routing-options forwarding-table export load-balance
set routing-options forwarding-table ecmp-fast-reroute
set protocols bgp group underlay type external
set protocols bgp group underlay export lo0
set protocols bgp group underlay local-as 65011
set protocols bgp group underlay multipath multiple-as
set protocols bgp group underlay neighbor 10.1.11.1 peer-as 65001
set protocols bgp group underlay neighbor 10.1.21.1 peer-as 65002
set protocols bgp group underlay neighbor 10.1.111.2 peer-as 65111
set protocols bgp group underlay neighbor 10.1.112.2 peer-as 65112
```

Spine-2

```
set system host-name spine-2
set interfaces xe-0/0/0 unit 0 family inet address 10.1.12.2/30
set interfaces xe-0/0/1 unit 0 family inet address 10.1.22.2/30
set interfaces xe-0/0/2 unit 0 family inet address 10.1.121.1/30
set interfaces xe-0/0/3 unit 0 family inet address 10.1.122.1/30
set interfaces lo0 unit 0 family inet address 10.1.255.12/32
set policy-options policy-statement lo0 from family inet
set policy-options policy-statement lo0 from protocol direct
set policy-options policy-statement lo0 from route-filter 0.0.0.0/0 prefix-length-range /32-/32
set policy-options policy-statement lo0 then accept
```

```

set policy-options policy-statement load-balance term 1 then load-balance per-packet
set routing-options router-id 10.1.255.12
set routing-options autonomous-system 65000
set routing-options forwarding-table export load-balance
set routing-options forwarding-table ecmp-fast-reroute
set protocols bgp group underlay type external
set protocols bgp group underlay export lo0
set protocols bgp group underlay local-as 65012
set protocols bgp group underlay multipath multiple-as
set protocols bgp group underlay neighbor 10.1.12.1 peer-as 65001
set protocols bgp group underlay neighbor 10.1.22.1 peer-as 65002
set protocols bgp group underlay neighbor 10.1.121.2 peer-as 65111
set protocols bgp group underlay neighbor 10.1.122.2 peer-as 65112

```

Core-1

```

set system host-name core-1
set interfaces xe-0/2/0 unit 0 family inet address 10.1.11.1/30
set interfaces xe-0/2/1 unit 0 family inet address 10.1.12.1/30
set interfaces irb unit 101 virtual-gateway-accept-data
set interfaces irb unit 101 family inet address 10.1.101.1/24 virtual-gateway-address
10.1.101.254
set interfaces irb unit 102 virtual-gateway-accept-data
set interfaces irb unit 102 family inet address 10.1.102.1/24 virtual-gateway-address
10.1.102.254
set interfaces irb unit 103 virtual-gateway-accept-data
set interfaces irb unit 103 family inet address 10.1.103.1/24 virtual-gateway-address
10.1.103.254
set interfaces lo0 unit 0 family inet address 10.1.255.1/32
set policy-options policy-statement VS_VLAN101_IMP term ESI from community comm-leaf
set policy-options policy-statement VS_VLAN101_IMP term ESI then accept
set policy-options policy-statement VS_VLAN101_IMP term VS_VLAN101 from community comm-VS_VLAN101
set policy-options policy-statement VS_VLAN101_IMP term VS_VLAN101 then accept
set policy-options policy-statement VS_VLAN102_IMP term ESI from community comm-leaf
set policy-options policy-statement VS_VLAN102_IMP term ESI then accept
set policy-options policy-statement VS_VLAN102_IMP term VS_VLAN102 from community comm-VS_VLAN102
set policy-options policy-statement VS_VLAN102_IMP term VS_VLAN102 then accept
set policy-options policy-statement VS_VLAN103_IMP term ESI from community comm-leaf
set policy-options policy-statement VS_VLAN103_IMP term ESI then accept
set policy-options policy-statement VS_VLAN103_IMP term VS_VLAN103 from community comm-VS_VLAN103
set policy-options policy-statement VS_VLAN103_IMP term VS_VLAN103 then accept
set policy-options policy-statement lo0 from family inet

```

```

set policy-options policy-statement lo0 from protocol direct
set policy-options policy-statement lo0 from route-filter 0.0.0.0/0 prefix-length-range /32-/32
set policy-options policy-statement lo0 then accept
set policy-options policy-statement load-balance term 1 then load-balance per-packet
set policy-options community comm-VS_VLAN101 members target:65000:101
set policy-options community comm-VS_VLAN102 members target:65000:102
set policy-options community comm-VS_VLAN103 members target:65000:103
set policy-options community comm-leaf members target:65000:1
set routing-instances VRF_Tenant_A instance-type vrf
set routing-instances VRF_Tenant_A interface irb.101
set routing-instances VRF_Tenant_A route-distinguisher 10.1.255.1:1010
set routing-instances VRF_Tenant_A vrf-target target:65000:101
set routing-instances VRF_Tenant_B instance-type vrf
set routing-instances VRF_Tenant_B interface irb.102
set routing-instances VRF_Tenant_B route-distinguisher 10.1.255.1:1020
set routing-instances VRF_Tenant_B vrf-target target:65000:102
set routing-instances VRF_Tenant_C instance-type vrf
set routing-instances VRF_Tenant_C interface irb.103
set routing-instances VRF_Tenant_C route-distinguisher 10.1.255.1:1030
set routing-instances VRF_Tenant_C vrf-target target:65000:103
set routing-instances VS_VLAN101 instance-type virtual-switch
set routing-instances VS_VLAN101 protocols evpn encapsulation vxlan
set routing-instances VS_VLAN101 protocols evpn extended-vni-list 101
set routing-instances VS_VLAN101 protocols evpn multicast-mode ingress-replication
set routing-instances VS_VLAN101 vtep-source-interface lo0.0
set routing-instances VS_VLAN101 bridge-domains bd101 vlan-id 101
set routing-instances VS_VLAN101 bridge-domains bd101 routing-interface irb.101
set routing-instances VS_VLAN101 bridge-domains bd101 vxlan vni 101
set routing-instances VS_VLAN101 route-distinguisher 10.1.255.1:101
set routing-instances VS_VLAN101 vrf-import VS_VLAN101_IMP
set routing-instances VS_VLAN101 vrf-target target:65000:101
set routing-instances VS_VLAN102 instance-type virtual-switch
set routing-instances VS_VLAN102 protocols evpn encapsulation vxlan
set routing-instances VS_VLAN102 protocols evpn extended-vni-list 102
set routing-instances VS_VLAN102 protocols evpn multicast-mode ingress-replication
set routing-instances VS_VLAN102 vtep-source-interface lo0.0
set routing-instances VS_VLAN102 bridge-domains bd102 vlan-id 102
set routing-instances VS_VLAN102 bridge-domains bd102 routing-interface irb.102
set routing-instances VS_VLAN102 bridge-domains bd102 vxlan vni 102
set routing-instances VS_VLAN102 route-distinguisher 10.1.255.1:102
set routing-instances VS_VLAN102 vrf-import VS_VLAN102_IMP
set routing-instances VS_VLAN102 vrf-target target:65000:102
set routing-instances VS_VLAN103 instance-type virtual-switch

```

```

set routing-instances VS_VLAN103 protocols evpn encapsulation vxlan
set routing-instances VS_VLAN103 protocols evpn extended-vni-list 103
set routing-instances VS_VLAN103 protocols evpn multicast-mode ingress-replication
set routing-instances VS_VLAN103 vtep-source-interface lo0.0
set routing-instances VS_VLAN103 bridge-domains bd103 vlan-id 103
set routing-instances VS_VLAN103 bridge-domains bd103 routing-interface irb.103
set routing-instances VS_VLAN103 bridge-domains bd103 vxlan vni 103
set routing-instances VS_VLAN103 route-distinguisher 10.1.255.1:103
set routing-instances VS_VLAN103 vrf-import VS_VLAN103_IMP
set routing-instances VS_VLAN103 vrf-target target:65000:103
set routing-options router-id 10.1.255.1
set routing-options autonomous-system 65000
set routing-options forwarding-table export load-balance
set routing-options forwarding-table ecmp-fast-reroute
set protocols bgp group underlay type external
set protocols bgp group underlay export lo0
set protocols bgp group underlay local-as 65001
set protocols bgp group underlay multipath multiple-as
set protocols bgp group underlay neighbor 10.1.11.2 peer-as 65011
set protocols bgp group underlay neighbor 10.1.12.2 peer-as 65012
set protocols bgp group EVPN_VXLAN type internal
set protocols bgp group EVPN_VXLAN local-address 10.1.255.1
set protocols bgp group EVPN_VXLAN family evpn signaling
set protocols bgp group EVPN_VXLAN cluster 1.1.1.1
set protocols bgp group EVPN_VXLAN multipath
set protocols bgp group EVPN_VXLAN neighbor 10.1.255.111
set protocols bgp group EVPN_VXLAN neighbor 10.1.255.112
set protocols bgp group EVPN_VXLAN neighbor 10.1.255.2

```

Core-2

```

set system host-name core-2
set interfaces xe-0/2/0 unit 0 family inet address 10.1.21.1/30
set interfaces xe-0/2/1 unit 0 family inet address 10.1.22.1/30
set interfaces irb unit 101 virtual-gateway-accept-data
set interfaces irb unit 101 family inet address 10.1.101.2/24 virtual-gateway-address
10.1.101.254
set interfaces irb unit 102 virtual-gateway-accept-data
set interfaces irb unit 102 family inet address 10.1.102.2/24 virtual-gateway-address
10.1.102.254
set interfaces irb unit 103 virtual-gateway-accept-data
set interfaces irb unit 103 family inet address 10.1.103.2/24 virtual-gateway-address

```



```

10.1.103.254
set interfaces lo0 unit 0 family inet address 10.1.255.2/32
set policy-options policy-statement VS_VLAN101_IMP term ESI from community comm-leaf
set policy-options policy-statement VS_VLAN101_IMP term ESI then accept
set policy-options policy-statement VS_VLAN101_IMP term VS_VLAN101 from community comm-VS_VLAN101
set policy-options policy-statement VS_VLAN101_IMP term VS_VLAN101 then accept
set policy-options policy-statement VS_VLAN102_IMP term ESI from community comm-leaf
set policy-options policy-statement VS_VLAN102_IMP term ESI then accept
set policy-options policy-statement VS_VLAN102_IMP term VS_VLAN102 from community comm-VS_VLAN102
set policy-options policy-statement VS_VLAN102_IMP term VS_VLAN102 then accept
set policy-options policy-statement VS_VLAN103_IMP term ESI from community comm-leaf
set policy-options policy-statement VS_VLAN103_IMP term ESI then accept
set policy-options policy-statement VS_VLAN103_IMP term VS_VLAN103 from community comm-VS_VLAN103
set policy-options policy-statement VS_VLAN103_IMP term VS_VLAN103 then accept
set policy-options policy-statement lo0 from family inet
set policy-options policy-statement lo0 from protocol direct
set policy-options policy-statement lo0 from route-filter 0.0.0.0/0 prefix-length-range /32-/32
set policy-options policy-statement lo0 then accept
set policy-options policy-statement load-balance term 1 then load-balance per-packet
set policy-options community comm-VS_VLAN101 members target:65000:101
set policy-options community comm-VS_VLAN102 members target:65000:102
set policy-options community comm-VS_VLAN103 members target:65000:103
set policy-options community comm-leaf members target:65000:1
set routing-instances VRF_Tenant_A instance-type vrf
set routing-instances VRF_Tenant_A interface irb.101
set routing-instances VRF_Tenant_A route-distinguisher 10.1.255.2:1010
set routing-instances VRF_Tenant_A vrf-target target:65000:101
set routing-instances VRF_Tenant_B instance-type vrf
set routing-instances VRF_Tenant_B interface irb.102
set routing-instances VRF_Tenant_B route-distinguisher 10.1.255.2:1020
set routing-instances VRF_Tenant_B vrf-target target:65000:102
set routing-instances VRF_Tenant_C instance-type vrf
set routing-instances VRF_Tenant_C interface irb.103
set routing-instances VRF_Tenant_C route-distinguisher 10.1.255.2:1030
set routing-instances VRF_Tenant_C vrf-target target:65000:103
set routing-instances VS_VLAN101 instance-type virtual-switch
set routing-instances VS_VLAN101 protocols evpn encapsulation vxlan
set routing-instances VS_VLAN101 protocols evpn extended-vni-list 101
set routing-instances VS_VLAN101 protocols evpn multicast-mode ingress-replication
set routing-instances VS_VLAN101 vtep-source-interface lo0.0
set routing-instances VS_VLAN101 bridge-domains bd101 vlan-id 101
set routing-instances VS_VLAN101 bridge-domains bd101 routing-interface irb.101
set routing-instances VS_VLAN101 bridge-domains bd101 vxlan vni 101

```

```

set routing-instances VS_VLAN101 route-distinguisher 10.1.255.2:101
set routing-instances VS_VLAN101 vrf-import VS_VLAN101_IMP
set routing-instances VS_VLAN101 vrf-target target:65000:101
set routing-instances VS_VLAN102 instance-type virtual-switch
set routing-instances VS_VLAN102 protocols evpn encapsulation vxlan
set routing-instances VS_VLAN102 protocols evpn extended-vni-list 102
set routing-instances VS_VLAN102 protocols evpn multicast-mode ingress-replication
set routing-instances VS_VLAN102 vtep-source-interface lo0.0
set routing-instances VS_VLAN102 bridge-domains bd102 vlan-id 102
set routing-instances VS_VLAN102 bridge-domains bd102 routing-interface irb.102
set routing-instances VS_VLAN102 bridge-domains bd102 vxlan vni 102
set routing-instances VS_VLAN102 route-distinguisher 10.1.255.2:102
set routing-instances VS_VLAN102 vrf-import VS_VLAN102_IMP
set routing-instances VS_VLAN102 vrf-target target:65000:102
set routing-instances VS_VLAN103 instance-type virtual-switch
set routing-instances VS_VLAN103 protocols evpn encapsulation vxlan
set routing-instances VS_VLAN103 protocols evpn extended-vni-list 103
set routing-instances VS_VLAN103 protocols evpn multicast-mode ingress-replication
set routing-instances VS_VLAN103 vtep-source-interface lo0.0
set routing-instances VS_VLAN103 bridge-domains bd103 vlan-id 103
set routing-instances VS_VLAN103 bridge-domains bd103 routing-interface irb.103
set routing-instances VS_VLAN103 bridge-domains bd103 vxlan vni 103
set routing-instances VS_VLAN103 route-distinguisher 10.1.255.2:103
set routing-instances VS_VLAN103 vrf-import VS_VLAN103_IMP
set routing-instances VS_VLAN103 vrf-target target:65000:103
set routing-options router-id 10.1.255.2
set routing-options autonomous-system 65000
set routing-options forwarding-table export load-balance
set routing-options forwarding-table ecmp-fast-reroute
set protocols bgp group underlay type external
set protocols bgp group underlay export lo0
set protocols bgp group underlay local-as 65002
set protocols bgp group underlay multipath multiple-as
set protocols bgp group underlay neighbor 10.1.21.2 peer-as 65011
set protocols bgp group underlay neighbor 10.1.22.2 peer-as 65012
set protocols bgp group EVPN_VXLAN type internal
set protocols bgp group EVPN_VXLAN local-address 10.1.255.2
set protocols bgp group EVPN_VXLAN family evpn signaling
set protocols bgp group EVPN_VXLAN cluster 2.2.2.2
set protocols bgp group EVPN_VXLAN multipath
set protocols bgp group EVPN_VXLAN neighbor 10.1.255.111

```

```
set protocols bgp group EVPN_VXLAN neighbor 10.1.255.112
set protocols bgp group EVPN_VXLAN neighbor 10.1.255.1
```

EX9200 Configuration

On EX9200 switches, the `vlan` statement is used instead of `bridge-domains`, and the `l3-interface` statement is used instead of `routing-interface`.

The following example shows how to configure these statements. All other configuration shown for MX Series routers in this example also applies to EX9200 switches.

```
set routing-instances VS_VLAN300 vlan vlan1300 vlan-id 300
set routing-instances VS_VLAN300 vlan vlan1300 l3-interface irb.1300
```

NOTE: In this example, wherever `bridge-domains` or `routing-interface` statements are used, to configure on EX9200 switches, use `vlan` and `l3-interface` instead.

Configuring Leaf-1

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

NOTE: The steps for configuring Leaf-2 are similar to Leaf-1 and therefore we will only show the step-by-step procedures for Leaf-1.

To configure Leaf-1:

1. Set the system hostname.

```
[edit]
user@leaf-1# set system host-name leaf-1
```

2. Configure routing options. The *load-balance* export policy is configured in the next step.

```
[edit]
user@leaf-1# set routing-options router-id 10.1.255.111
user@leaf-1# set routing-options autonomous-system 65000
user@leaf-1# set routing-options forwarding-table export load-balance
user@leaf-1# set routing-options forwarding-table ecmp-fast-reroute
```

3. Configure the load balancing policy.

```
[edit policy-options policy-statement load-balance]
user@leaf-1# set term 1 then load-balance per-packet
```

4. Configure the underlay EBGP to the spine devices. The *lo0* export policy is configured in the next step.

```
[edit]
user@leaf-1# set protocols bgp group underlay type external
user@leaf-1# set protocols bgp group underlay export lo0
user@leaf-1# set protocols bgp group underlay local-as 65111
user@leaf-1# set protocols bgp group underlay multipath multiple-as
user@leaf-1# set protocols bgp group underlay neighbor 10.1.111.1 peer-as 65011
user@leaf-1# set protocols bgp group underlay neighbor 10.1.121.1 peer-as 65012
```

5. Configure a policy to advertise the loopback address into the underlay. In this example you write a portable policy that is loopback address agnostic, by matching only direct routes with a /32 prefix length. The result is a policy that matches any loopback address and is reusable across all devices in the topology.

```
[edit policy-options policy-statement lo0]
user@leaf-1# set from family inet
user@leaf-1# set from protocol direct
user@leaf-1# set from route-filter 0.0.0.0/0 prefix-length-range /32-/32
user@leaf-1# set then accept
```

6. Configure switch options The virtual tunnel endpoint interface is lo0.0, which must be reachable through the underlay routing protocol. The route distinguisher must be unique across all switches in the network to ensure all route advertisements within MP-BGP overlay are globally unique. The VRF table target on the QFX Series switch is, at a minimum, the community the switch sends

attaches to all ESI (Type-1) routes. The `vrf-import vrf-imp` statement defines the target community list, which is imported into the `default-switch.evpn.0` instance from the `bgp.evpn.0` table.

```
[edit]
user@leaf-1# set switch-options vtep-source-interface lo0.0
user@leaf-1# set switch-options route-distinguisher 10.1.255.111:1
user@leaf-1# set switch-options vrf-import vrf-imp
user@leaf-1# set switch-options vrf-target target:65000:1
```

7. Configure the VRF table import policy.

```
[edit]
user@leaf-1# set policy-options policy-statement vrf-imp term t1 from community com101
user@leaf-1# set policy-options policy-statement vrf-imp term t1 then accept
user@leaf-1# set policy-options policy-statement vrf-imp term t2 from community com102
user@leaf-1# set policy-options policy-statement vrf-imp term t2 then accept
user@leaf-1# set policy-options policy-statement vrf-imp term t3 from community com103
user@leaf-1# set policy-options policy-statement vrf-imp term t3 then accept
user@leaf-1# set policy-options policy-statement vrf-imp term t5 then reject
```

8. Configure the related communities.

```
[edit]
user@leaf-1# set policy-options community com101 members target:65000:101
user@leaf-1# set policy-options community com102 members target:65000:102
user@leaf-1# set policy-options community com103 members target:65000:103
```

9. Configure the extended virtual network identifier (VNI) list to establish the VNIs you want to be part of the EVPN domain. You also configure ingress replication; in EVPN-VXLAN ingress-replication is used to handle multicast without requiring a multicast capable underlay. Different route targets are specified for each VXLAN network identifier instance under `vni-routing-options`.

```
[edit]
user@leaf-1# set protocols evpn encapsulation vxlan
user@leaf-1# set protocols evpn multicast-mode ingress-replication
user@leaf-1# set protocols evpn vni-options vni 101 vrf-target target:65000:101
user@leaf-1# set protocols evpn vni-options vni 102 vrf-target target:65000:102
```

```
user@leaf-1# set protocols evpn extended-vni-list 101
user@leaf-1# set protocols evpn extended-vni-list 102
```

10. Map locally significant VLAN IDs to globally significant VXLAN network identifiers.

```
[edit]
user@leaf-1# set vlans v101 vlan-id 101
user@leaf-1# set vlans v101 vxlan vni 101
user@leaf-1# set vlans v102 vlan-id 102
user@leaf-1# set vlans v102 vxlan vni 102
```

11. Configure the EVPN MP-BGP overlay sessions.

```
[edit]
user@leaf-1# set protocols bgp group EVPN_VXLAN_CORE type internal
user@leaf-1# set protocols bgp group EVPN_VXLAN_CORE local-address 10.1.255.111
user@leaf-1# set protocols bgp group EVPN_VXLAN_CORE family evpn signaling
user@leaf-1# set protocols bgp group EVPN_VXLAN_CORE neighbor 10.1.255.1
user@leaf-1# set protocols bgp group EVPN_VXLAN_CORE neighbor 10.1.255.2
```

12. Configure the fabric interfaces.

```
[edit]
user@leaf-1# set interfaces xe-0/0/0 unit 0 family inet address 10.1.111.2/30
user@leaf-1# set interfaces xe-0/0/1 unit 0 family inet address 10.1.121.2/30
user@leaf-1# set interfaces xe-0/0/2 ether-options 802.3ad ae0
user@leaf-1# set interfaces xe-0/0/3 unit 0 family ethernet-switching interface-mode trunk
user@leaf-1# set interfaces xe-0/0/3 unit 0 family ethernet-switching vlan members v101
```

13. Configure the LACP-enabled LAG interface. The ESI value is globally unique across the entire EVPN domain. The all-active configuration statement ensures that all PE routers to which this multihomed tenant is attached to can forward traffic from the CE device, such that all CE links are actively used.

```
[edit]
user@leaf-1# set interfaces ae0 esi 00:01:01:01:01:01:01:01:01
user@leaf-1# set interfaces ae0 esi all-active
user@leaf-1# set interfaces ae0 aggregated-ether-options lacp active
user@leaf-1# set interfaces ae0 aggregated-ether-options lacp system-id 00:00:00:01:01:01
```

```
user@leaf-1# set interfaces ae0 unit 0 family ethernet-switching interface-mode access
user@leaf-1# set interfaces ae0 unit 0 family ethernet-switching vlan members v102
```

14. Configure the loopback interface address.

```
[edit]
user@leaf-1# set interfaces lo0 unit 0 family inet address 10.1.255.111/32
```

Configuring Spine-1

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

NOTE: The steps for configuring Spine-2 are similar to Spine-1 and therefore we will only show the step-by-step procedures for Spine-1.

To configure Spine-1:

1. Set the system hostname.

```
[edit]
user@spine-1# set system host-name spine-1
```

2. Configure the routing options.

```
[edit]
user@spine-1# set routing-options router-id 10.1.255.11
user@spine-1# set routing-options autonomous-system 65000
user@spine-1# set routing-options forwarding-table export load-balance
user@spine-1# set routing-options forwarding-table ecmp-fast-reroute
```

3. Configure a load balancing policy.

```
[edit policy-options policy-statement load-balance]
user@spine-1# set term 1 then load-balance per-packet
```

4. Configure the EBGP underlay with peering to the leaf and core devices. The *lo0* policy that advertises the lo0 address is applied in this step; the configuration of the policy itself is shown in the next step.

```
[edit]
user@spine-1# set protocols bgp group underlay type external
user@spine-1# set protocols bgp group underlay export lo0
user@spine-1# set protocols bgp group underlay local-as 65011
user@spine-1# set protocols bgp group underlay multipath multiple-as
user@spine-1# set protocols bgp group underlay neighbor 10.1.11.1 peer-as 65001
user@spine-1# set protocols bgp group underlay neighbor 10.1.21.1 peer-as 65002
user@spine-1# set protocols bgp group underlay neighbor 10.1.111.2 peer-as 65111
user@spine-1# set protocols bgp group underlay neighbor 10.1.112.2 peer-as 65112
```

5. Configure the *lo0* policy .

```
[edit policy-options policy-statement lo0]
user@spine-1# set from family inet
user@spine-1# set from protocol direct
user@spine-1# set from route-filter 0.0.0.0/0 prefix-length-range /32-/32
user@spine-1# set then accept
```

Configuring Core-1

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

NOTE: The steps for configuring Core-2 are similar to Core-1 and therefore we will only show the step-by-step procedures for Core-1.

To configure Core-1:

1. Set the system hostname.

```
[edit]
user@core-1# set system host-name core-1
```

2. Configure the routing options. The *load-balance* policy is applied during this step. You configure the policy in the next step

```
[edit]
user@core-1# set routing-options router-id 10.1.255.1
user@core-1# set routing-options autonomous-system 65000
user@core-1# set routing-options forwarding-table export load-balance
user@core-1# set routing-options forwarding-table ecmp-fast-reroute
```

3. Configure the *load-balance* policy .

```
[edit policy-options policy-statement load-balance]
user@core-1# set term 1 then load-balance per-packet
```

4. Configure the BGP underlay peering. The *lo0* policy that advertises the loopback address is applied during this step. You configure the policy in the next step.

```
[edit]
user@core-1# set protocols bgp group underlay type external
user@core-1# set protocols bgp group underlay export lo0
user@core-1# set protocols bgp group underlay local-as 65001
user@core-1# set protocols bgp group underlay multipath multiple-as
user@core-1# set protocols bgp group underlay neighbor 10.1.11.2 peer-as 65011
user@core-1# set protocols bgp group underlay neighbor 10.1.12.2 peer-as 65012
```

5. Configure policy *lo0*.

```
[edit policy-options policy-statement lo0]
user@core-1# set from family inet
user@core-1# set from protocol direct
```

```

user@core-1# set lo0 from route-filter 0.0.0.0/0 prefix-length-range /32-/32
user@core-1# set lo0 then accept

```

6. A large portion of Core-1's configuration takes place in the [routing-instance] hierarchy. Configure the virtual routers and configure a unique VRF table import policy for each virtual switch.

```

[edit]
user@core-1# set routing-instances VRF_Tenant_A instance-type vrf
user@core-1# set routing-instances VRF_Tenant_A interface irb.101
user@core-1# set routing-instances VRF_Tenant_A route-distinguisher 10.1.255.1:1010
user@core-1# set routing-instances VRF_Tenant_A vrf-target target:65000:101
user@core-1# set routing-instances VRF_Tenant_B instance-type vrf
user@core-1# set routing-instances VRF_Tenant_B interface irb.102
user@core-1# set routing-instances VRF_Tenant_B route-distinguisher 10.1.255.1:1020
user@core-1# set routing-instances VRF_Tenant_B vrf-target target:65000:102
user@core-1# set routing-instances VRF_Tenant_C instance-type vrf
user@core-1# set routing-instances VRF_Tenant_C interface irb.103
user@core-1# set routing-instances VRF_Tenant_C route-distinguisher 10.1.255.1:1030
user@core-1# set routing-instances VRF_Tenant_C vrf-target target:65000:103
user@core-1# set routing-instances VS_VLAN101 instance-type virtual-switch
user@core-1# set routing-instances VS_VLAN101 protocols evpn encapsulation vxlan
user@core-1# set routing-instances VS_VLAN101 protocols evpn extended-vni-list 101
user@core-1# set routing-instances VS_VLAN101 protocols evpn multicast-mode ingress-
replication
user@core-1# set routing-instances VS_VLAN101 vtep-source-interface lo0.0
user@core-1# set routing-instances VS_VLAN101 bridge-domains bd101 vlan-id 101
user@core-1# set routing-instances VS_VLAN101 bridge-domains bd101 routing-interface irb.101
user@core-1# set routing-instances VS_VLAN101 bridge-domains bd101 vxlan vni 101
user@core-1# set routing-instances VS_VLAN101 route-distinguisher 10.1.255.1:101
user@core-1# set routing-instances VS_VLAN101 vrf-import VS_VLAN101_IMP
user@core-1# set routing-instances VS_VLAN101 vrf-target target:65000:101
user@core-1# set routing-instances VS_VLAN102 instance-type virtual-switch
user@core-1# set routing-instances VS_VLAN102 protocols evpn encapsulation vxlan
user@core-1# set routing-instances VS_VLAN102 protocols evpn extended-vni-list 102
user@core-1# set routing-instances VS_VLAN102 protocols evpn multicast-mode ingress-
replication
user@core-1# set routing-instances VS_VLAN102 vtep-source-interface lo0.0
user@core-1# set routing-instances VS_VLAN102 bridge-domains bd102 vlan-id 102
user@core-1# set routing-instances VS_VLAN102 bridge-domains bd102 routing-interface irb.102
user@core-1# set routing-instances VS_VLAN102 bridge-domains bd102 vxlan vni 102
user@core-1# set routing-instances VS_VLAN102 route-distinguisher 10.1.255.1:102
user@core-1# set routing-instances VS_VLAN102 vrf-import VS_VLAN102_IMP

```

```

user@core-1# set routing-instances VS_VLAN102 vrf-target target:65000:102
user@core-1# set routing-instances VS_VLAN103 instance-type virtual-switch
user@core-1# set routing-instances VS_VLAN103 protocols evpn encapsulation vxlan
user@core-1# set routing-instances VS_VLAN103 protocols evpn extended-vni-list 103
user@core-1# set routing-instances VS_VLAN103 protocols evpn multicast-mode ingress-
replication
user@core-1# set routing-instances VS_VLAN103 vtep-source-interface lo0.0
user@core-1# set routing-instances VS_VLAN103 bridge-domains bd103 vlan-id 103
user@core-1# set routing-instances VS_VLAN103 bridge-domains bd103 routing-interface irb.103
user@core-1# set routing-instances VS_VLAN103 bridge-domains bd103 vxlan vni 103
user@core-1# set routing-instances VS_VLAN103 route-distinguisher 10.1.255.1:103
user@core-1# set routing-instances VS_VLAN103 vrf-import VS_VLAN103_IMP
user@core-1# set routing-instances VS_VLAN103 vrf-target target:65000:103

```

7. Configure the policy for each routing instance.

```

[edit policy-options]
user@core-1# set policy-statement VS_VLAN101_IMP term ESI from community comm-leaf
user@core-1# set policy-statement VS_VLAN101_IMP term ESI then accept
user@core-1# set policy-statement VS_VLAN101_IMP term VS_VLAN101 from community comm-
VS_VLAN101
user@core-1# set policy-statement VS_VLAN101_IMP term VS_VLAN101 then accept
user@core-1# set policy-statement VS_VLAN102_IMP term ESI from community comm-leaf
user@core-1# set policy-statement VS_VLAN102_IMP term ESI then accept
user@core-1# set policy-statement VS_VLAN102_IMP term VS_VLAN102 from community comm-
VS_VLAN102
user@core-1# set policy-statement VS_VLAN102_IMP term VS_VLAN102 then accept
user@core-1# set policy-statement VS_VLAN103_IMP term ESI from community comm-leaf
user@core-1# set policy-statement VS_VLAN103_IMP term ESI then accept
user@core-1# set policy-statement VS_VLAN103_IMP term VS_VLAN103 from community comm-
VS_VLAN103
user@core-1# set policy-statement VS_VLAN103_IMP term VS_VLAN103 then accept

```

8. Configure the communities . Make sure that the *comm-leaf* policy accepts routes tagged with target 65000:1. This ensures that all virtual switches import the Type-1 ESI routes from all leafs.

```

[edit]
user@core-1# set policy-options community comm-VS_VLAN101 members target:65000:101
user@core-1# set policy-options community comm-VS_VLAN102 members target:65000:102

```

```

user@core-1# set policy-options community comm-VS_VLAN103 members target:65000:103
user@core-1# set policy-options community comm-leaf members target:65000:1

```

9. Configure the IRB interfaces. Every IRB has a virtual gateway address, which is a shared MAC address and IP address across Core-1 and Core-2.

```

[edit interfaces irb]
user@core-1# set unit 101 virtual-gateway-accept-data
user@core-1# set unit 101 family inet address 10.1.101.1/24 virtual-gateway-address 10.1.101.254
user@core-1# set unit 102 virtual-gateway-accept-data
user@core-1# set unit 102 family inet address 10.1.102.1/24 virtual-gateway-address 10.1.102.254
user@core-1# set unit 103 virtual-gateway-accept-data
user@core-1# set unit 103 family inet address 10.1.103.1/24 virtual-gateway-address 10.1.103.254

```

10. Configure the IBGP overlay sessions towards Leaf-1 and Leaf-2. We've include a peering between the Core devices for route sharing between Core devices.

```

[edit]
user@core-1# set protocols bgp group EVPN_VXLAN type internal
user@core-1# set protocols bgp group EVPN_VXLAN local-address 10.1.255.1
user@core-1# set protocols bgp group EVPN_VXLAN family evpn signaling
user@core-1# set protocols bgp group EVPN_VXLAN cluster 1.1.1.1
user@core-1# set protocols bgp group EVPN_VXLAN multipath
user@core-1# set protocols bgp group EVPN_VXLAN neighbor 10.1.255.111
user@core-1# set protocols bgp group EVPN_VXLAN neighbor 10.1.255.112
user@core-1# set protocols bgp group EVPN_VXLAN neighbor 10.1.255.2

```

Verification

IN THIS SECTION

- [Verifying MAC Reachability to a Single-Homed CE Device \(Leaf-1\) | 602](#)
- [Verifying MAC Reachability to a Single-Homed CE Device \(Type-2\) | 603](#)
- [Verifying the Imported Route | 604](#)

- [Verifying the Layer 2 Address Learning Daemon Copy | 605](#)
- [Verifying the Kernel-Level Forwarding Table | 606](#)
- [Verifying MAC Reachability to a Multihomed CE Device | 608](#)
- [Verifying EVPN, Layer 2 Address Learning Daemon, and the Kernel-Forwarding Tables for Multihomed CE Device | 610](#)

Verifying MAC Reachability to a Single-Homed CE Device (Leaf-1)

Purpose

Verify MAC reachability to Tenant_A. This user is single-homed to Leaf-1. First, verify that the MAC address is learned locally on Leaf-1. Leaf-1 generates the Type-2 route only after it learns the MAC address.

Action

Verify that the MAC address is learned locally on Leaf-1.

```
lab@leaf-1> show ethernet-switching table vlan-id 101

MAC flags (S - static MAC, D - dynamic MAC, L - locally learned, P - Persistent static
          SE - statistics enabled, NM - non configured MAC, R - remote PE MAC, O - ovsdb MAC)

Ethernet switching table : 4 entries, 4 learned
Routing instance : default-switch
  Vlan      MAC          MAC    Logical      SVLBNH/      Active
  name      address       flags   interface    VENH Index   source
  v101      00:00:5e:00:01:01  DRP     esi.1749
05:00:00:fd:e8:00:00:00:65:00
  v101      2c:6b:f5:54:95:f0  DR       vtep.32770
10.1.255.2
  v101      2c:6b:f5:ef:73:f0  DR       vtep.32769
10.1.255.1
  v101      56:04:15:00:bb:02  D        xe-0/0/3.0
```

Meaning

The output shows that MAC 56:04:15:00:bb:02 is successfully learned from the Tenant_A CE device, which is Server A on the xe-0/0/3.0 interface.

Verifying MAC Reachability to a Single-Homed CE Device (Type-2)

Purpose

Verify MAC reachability to a single-homed CE device (Type-2)

Action

Verify the generation of the Type-2 route to Core-1.

```
lab@leaf-1> show route advertising-protocol bgp 10.1.255.1 evpn-mac-address 56:04:15:00:bb:02
```

```
bgp.evpn.0: 50 destinations, 91 routes (50 active, 0 holddown, 0 hidden)
```

Prefix	Nexthop	MED	Lclpref	AS path
2:10.1.255.111:1::101::56:04:15:00:bb:02/304	MAC/IP			
*	Self		100	I
2:10.1.255.111:1::101::56:04:15:00:bb:02::10.1.101.101/304	MAC/IP			
*	Self		100	I

```
default-switch.evpn.0: 47 destinations, 87 routes (47 active, 0 holddown, 0 hidden)
```

Prefix	Nexthop	MED	Lclpref	AS path
2:10.1.255.111:1::101::56:04:15:00:bb:02/304	MAC/IP			
*	Self		100	I
2:10.1.255.111:1::101::56:04:15:00:bb:02::10.1.101.101/304	MAC/IP			
*	Self		100	I

```
__default_evpn__.evpn.0: 3 destinations, 4 routes (3 active, 0 holddown, 0 hidden)
```

Meaning

The output shows that the MAC and MAC/IP are being advertised.

On Core-1, the Type-2 route is received into bgp.evpn.0.

```
lab@core-1> show route receive-protocol bgp 10.1.255.111 evpn-mac-address 56:04:15:00:bb:02
extensive table bgp.evpn.0

bgp.evpn.0: 52 destinations, 68 routes (52 active, 0 holddown, 0 hidden)
* 2:10.1.255.111:1::101::56:04:15:00:bb:02/304 MAC/IP (2 entries, 1 announced)
  Import Accepted
  Route Distinguisher: 10.1.255.111:1
  Route Label: 101
  ESI: 00:00:00:00:00:00:00:00:00:00
  Nexthop: 10.1.255.111
  Localpref: 100
  AS path: I
  Communities: target:65000:101 encapsulation:vxlan(0x8)

* 2:10.1.255.111:1::101::56:04:15:00:bb:02::10.1.101.101/304 MAC/IP (2 entries, 1 announced)
  Import Accepted
  Route Distinguisher: 10.1.255.111:1
  Route Label: 101
  ESI: 00:00:00:00:00:00:00:00:00:00
  Nexthop: 10.1.255.111
  Localpref: 100
  AS path: I
  Communities: target:65000:101 encapsulation:vxlan(0x8)
```

The output shows the Type-2 routes for 56:04:15:00:bb:02. The route distinguisher is from Leaf-1 and is set to 10.1.255.111:1.

Verifying the Imported Route

Purpose

Verify that the EVPN Type-2 route is imported.

Action

On Core-1, verify whether EVPN Type-2 routes are successfully imported from the bgp.evpn.0 table into the EVPN switch instance.

Meaning

The output shows that, in Tenant_A's virtual switch, the Type-2 route is advertised with the correct target, target:1:101. Use the extensive option to review the Type-2 route in greater detail.

```
lab@core-1> show route table VS_VLAN101.evpn.0 evpn-mac-address 56:04:15:00:bb:02
```

```
VS_VLAN101.evpn.0: 18 destinations, 25 routes (18 active, 0 holddown, 0 hidden)
```

```
+ = Active Route, - = Last Active, * = Both
```

```
2:10.1.255.111:1::101::56:04:15:00:bb:02/304 MAC/IP
```

```
*[BGP/170] 1w1d 20:50:01, localpref 100, from 10.1.255.111
```

```
AS path: I, validation-state: unverified
```

```
> to 10.1.11.2 via xe-0/2/0.0
```

```
[BGP/170] 3d 02:56:43, localpref 100, from 10.1.255.2
```

```
AS path: I, validation-state: unverified
```

```
> to 10.1.11.2 via xe-0/2/0.0
```

```
2:10.1.255.111:1::101::56:04:15:00:bb:02::10.1.101.101/304 MAC/IP
```

```
*[BGP/170] 1w1d 20:50:01, localpref 100, from 10.1.255.111
```

```
AS path: I, validation-state: unverified
```

```
> to 10.1.11.2 via xe-0/2/0.0
```

```
[BGP/170] 3d 02:56:43, localpref 100, from 10.1.255.2
```

```
AS path: I, validation-state: unverified
```

```
> to 10.1.11.2 via xe-0/2/0.0
```

The output shows that Core-1 receives two copies. The first is the advertisement from Leaf-1 (Source: 10.1.255.111). The second is the advertisement from Core-2 (Source: 10.1.255.2).

Verifying the Layer 2 Address Learning Daemon Copy

Purpose

Verify the Layer 2 address learning daemon copy.

Action

Verify the Layer 2 address learning daemon copy by entering the `show bridge-mac table` command.

Meaning

The output shows that 56:04:15:00:bb:02 is reachable through the vtep.32771 logical interface to Leaf-1.

```
lab@core-1> show bridge mac-table instance VS_VLAN101
```

```
MAC flags      (S -static MAC, D -dynamic MAC, L -locally learned, C -Control MAC
               O -OVSDB MAC, SE -Statistics enabled, NM -Non configured MAC, R -Remote PE MAC, P -Pinned
               MAC, FU - Fast Update)
```

```
Routing instance : VS_VLAN101
```

```
Bridging domain : bd101, VLAN : 101
```

MAC address	MAC flags	Logical interface	Active source
00:00:5e:00:01:01	DRP	esi.722	05:00:00:fd:e8:00:00:00:65:00
2c:6b:f5:54:95:f0	DR	vtep.32779	10.1.255.2
56:04:15:00:bb:02	DR	vtep.32771	10.1.255.111

NOTE: On EX9200 switches, the `show ethernet-switching table-instance instance-name` command corresponds to the `show bridge mac-table instance instance-name` command used here for MX Series routers

Verifying the Kernel-Level Forwarding Table

Purpose

Verify the kernel-level forwarding table, next hop identifier, and Layer 2 MAC table and hardware.

Action

Query the kernel-level forwarding table, correlate the index next hop identifier with the correct virtual network identifier, and review the Layer 2 MAC table and hardware.

Meaning

Tenant_A's MAC, 56:04:15:00:bb:02, is reachable through index 687.

```
lab@core-1> show route forwarding-table family bridge vpn VS_VLAN101
Routing table: VS_VLAN101.evpn-vxlan
VPLS:
Destination      Type RtRef Next hop          Type Index   NhRef Netif
default          perm   0                dscd   664     1
vtep.32771       intf   0                comp    687     7
vtep.32774       intf   0                comp    691     4
vtep.32779       intf   0                comp    716     7

Routing table: VS_VLAN101.evpn-vxlan
Bridging domain: bd101.evpn-vxlan
VPLS:
Enabled protocols: Bridging, ACKed by all peers, EVPN VXLAN,
Destination      Type RtRef Next hop          Type Index   NhRef Netif
00:00:5e:00:01:01/48 user    0                indr  1048579   2
                                comp    722     2
2c:6b:f5:54:95:f0/48 user    0                comp    716     7
56:04:15:00:bb:02/48 user    0                comp    687     7
0x30003/51       user    0                comp    705     2
```

Correlate index 687 (NH-Id) with the correct virtual network identifier 101 and remote VTEP-ID of 10.1.255.111.

```
lab@core-1> show l2-learning vxlan-tunnel-end-point remote
Logical System Name      Id  SVTEP-IP      IFL  L3-Idx  SVTEP-Mode  ELP-SVTEP-IP
<default>                0   10.1.255.1    lo0.0  0
RVTEP-IP                 L2-RTT                IFL-Idx  Interface  NH-Id  RVTEP-Mode  ELP-
IP      Flags
10.1.255.2               VS_VLAN101            377      vtep.32779  716    RNVE
VNID                     MC-Group-IP
101                      0.0.0.0
RVTEP-IP                 L2-RTT                IFL-Idx  Interface  NH-Id  RVTEP-Mode  ELP-
IP      Flags
10.1.255.111            VS_VLAN101            369      vtep.32771  687    RNVE
VNID                     MC-Group-IP
101                      0.0.0.0
RVTEP-IP                 L2-RTT                IFL-Idx  Interface  NH-Id  RVTEP-Mode  ELP-
```

IP	Flags							
10.1.255.112	VS_VLAN101	372	vtep.32774	691	RNVE			
10.1.255.2	VS_VLAN102	376	vtep.32778	715	RNVE			
VNID	MC-Group-IP							
102	0.0.0.0							
RVTEP-IP	L2-RTT	IFL-Idx	Interface	NH-Id	RVTEP-Mode	ELP-		
IP	Flags							
10.1.255.111	VS_VLAN102	370	vtep.32772	688	RNVE			
VNID	MC-Group-IP							
102	0.0.0.0							
RVTEP-IP	L2-RTT	IFL-Idx	Interface	NH-Id	RVTEP-Mode	ELP-		
IP	Flags							
10.1.255.112	VS_VLAN102	373	vtep.32775	695	RNVE			
VNID	MC-Group-IP							
102	0.0.0.0							
RVTEP-IP	L2-RTT	IFL-Idx	Interface	NH-Id	RVTEP-Mode	ELP-		
IP	Flags							
10.1.255.2	VS_VLAN103	375	vtep.32777	714	RNVE			
VNID	MC-Group-IP							
103	0.0.0.0							
RVTEP-IP	L2-RTT	IFL-Idx	Interface	NH-Id	RVTEP-Mode	ELP-		
IP	Flags							
10.1.255.111	VS_VLAN103	371	vtep.32773	689	RNVE			
10.1.255.112	VS_VLAN103	374	vtep.32776	692	RNVE			
VNID	MC-Group-IP							
103	0.0.0.0							

NOTE: On EX9200 switches, the show ethernet-switching command corresponds to the show l2-learning command show here for MX Series routers.

Verifying MAC Reachability to a Multihomed CE Device

Purpose

Verify MAC reachability to the multihomed Tenant_B CE device on Leaf-1 and Leaf-2.

Action

Verify that Leaf-1 and Leaf-2 are advertising both Type-1 and Type-2 reachability towards the multihomed CE device.

```
lab@leaf-1> show ethernet-switching table vlan-id 102
```

MAC flags (S - static MAC, D - dynamic MAC, L - locally learned, P - Persistent static
SE - statistics enabled, NM - non configured MAC, R - remote PE MAC, O - ovsdb MAC)

Ethernet switching table : 4 entries, 4 learned

Routing instance : default-switch

Vlan name	MAC address	MAC flags	Logical interface	SVLBNH/ VENH Index	Active source
v102	00:00:5e:00:01:01	DR	esi.1748		
05:00:00:fd:e8:00:00:00:66:00					
v102	2c:6b:f5:43:12:c0	DL	ae0.0		
v102	2c:6b:f5:54:95:f0	D	vtep.32770		
10.1.255.2					
v102	2c:6b:f5:ef:73:f0	D	vtep.32769		
10.1.255.1					

```
lab@leaf-2> show ethernet-switching table vlan-id 102
```

MAC flags (S - static MAC, D - dynamic MAC, L - locally learned, P - Persistent static
SE - statistics enabled, NM - non configured MAC, R - remote PE MAC, O - ovsdb MAC)

Ethernet switching table : 4 entries, 4 learned

Routing instance : default-switch

Vlan name	MAC address	MAC flags	Logical interface	SVLBNH/ VENH Index	Active source
v102	00:00:5e:00:01:01	DR	esi.1749		
05:00:00:fd:e8:00:00:00:66:00					
v102	2c:6b:f5:43:12:c0	DR	ae0.0		
v102	2c:6b:f5:54:95:f0	D	vtep.32769		
10.1.255.2					
v102	2c:6b:f5:ef:73:f0	D	vtep.32770		
10.1.255.1					

Meaning

The output shows that 2c:6b:f5:43:12:c0 represents the MAC of the Tenant_B attached to Leaf-1 and Leaf-2.

Verifying EVPN, Layer 2 Address Learning Daemon, and the Kernel-Forwarding Tables for Multihomed CE Device

Purpose

Verify the Tenant B's EVPN table, and Core-1's Layer 2 address learning daemon table and kernel-forwarding table.

Action

In Core-1, display the Tenant B's EVPN table.

```
lab@core-1> show route table VS_VLAN102.evpn.0

VS_VLAN102.evpn.0: 20 destinations, 29 routes (20 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:10.1.255.2:0::050000fde80000006600::FFFF:FFFF/192 AD/ESI
    *[BGP/170] 2d 23:43:32, localpref 100, from 10.1.255.2
    AS path: I, validation-state: unverified
    to 10.1.11.2 via xe-0/2/0.0
    > to 10.1.12.2 via xe-0/2/1.0
1:10.1.255.111:0::0101010101010101::FFFF:FFFF/192 AD/ESI
    *[BGP/170] 00:14:59, localpref 100, from 10.1.255.111
    AS path: I, validation-state: unverified
    > to 10.1.11.2 via xe-0/2/0.0
    to 10.1.12.2 via xe-0/2/1.0
    [BGP/170] 00:14:58, localpref 100, from 10.1.255.2
    AS path: I, validation-state: unverified
    > to 10.1.11.2 via xe-0/2/0.0
    to 10.1.12.2 via xe-0/2/1.0
1:10.1.255.111:1::0101010101010101::0/192 AD/EVI
    *[BGP/170] 00:15:00, localpref 100, from 10.1.255.111
    AS path: I, validation-state: unverified
    > to 10.1.11.2 via xe-0/2/0.0
    to 10.1.12.2 via xe-0/2/1.0
    [BGP/170] 00:14:59, localpref 100, from 10.1.255.2
```

```

        AS path: I, validation-state: unverified
        > to 10.1.11.2 via xe-0/2/0.0
        to 10.1.12.2 via xe-0/2/1.0
1:10.1.255.112:0::0101010101010101::FFFF:FFFF/192 AD/ESI
    *[BGP/170] 00:10:13, localpref 100, from 10.1.255.112
        AS path: I, validation-state: unverified
        to 10.1.11.2 via xe-0/2/0.0
        > to 10.1.12.2 via xe-0/2/1.0
    [BGP/170] 00:10:13, localpref 100, from 10.1.255.2
        AS path: I, validation-state: unverified
        to 10.1.11.2 via xe-0/2/0.0
        > to 10.1.12.2 via xe-0/2/1.0
1:10.1.255.112:1::0101010101010101::0/192 AD/EVI
    *[BGP/170] 00:10:14, localpref 100, from 10.1.255.112
        AS path: I, validation-state: unverified
        to 10.1.11.2 via xe-0/2/0.0
        > to 10.1.12.2 via xe-0/2/1.0
    [BGP/170] 00:10:14, localpref 100, from 10.1.255.2
        AS path: I, validation-state: unverified
        to 10.1.11.2 via xe-0/2/0.0
        > to 10.1.12.2 via xe-0/2/1.0
2:10.1.255.1:102::102::00:00:5e:00:01:01/304 MAC/IP
    *[EVPN/170] 2d 23:44:03
        Indirect
2:10.1.255.1:102::102::2c:6b:f5:ef:73:f0/304 MAC/IP
    *[EVPN/170] 2d 23:44:03
        Indirect
2:10.1.255.2:102::102::00:00:5e:00:01:01/304 MAC/IP
    *[BGP/170] 2d 23:43:32, localpref 100, from 10.1.255.2
        AS path: I, validation-state: unverified
        to 10.1.11.2 via xe-0/2/0.0
        > to 10.1.12.2 via xe-0/2/1.0
2:10.1.255.2:102::102::2c:6b:f5:54:95:f0/304 MAC/IP
    *[BGP/170] 2d 23:43:32, localpref 100, from 10.1.255.2
        AS path: I, validation-state: unverified
        > to 10.1.11.2 via xe-0/2/0.0
        to 10.1.12.2 via xe-0/2/1.0
2:10.1.255.111:1::102::2c:6b:f5:43:12:c0/304 MAC/IP
    *[BGP/170] 00:14:49, localpref 100, from 10.1.255.111
        AS path: I, validation-state: unverified
        to 10.1.11.2 via xe-0/2/0.0
        > to 10.1.12.2 via xe-0/2/1.0
    [BGP/170] 00:14:49, localpref 100, from 10.1.255.2

```

```

        AS path: I, validation-state: unverified
        to 10.1.11.2 via xe-0/2/0.0
    > to 10.1.12.2 via xe-0/2/1.0
2:10.1.255.112:1::102::2c:6b:f5:43:12:c0/304 MAC/IP
    *[BGP/170] 00:09:24, localpref 100, from 10.1.255.112
        AS path: I, validation-state: unverified
    > to 10.1.11.2 via xe-0/2/0.0
        to 10.1.12.2 via xe-0/2/1.0
    [BGP/170] 00:09:24, localpref 100, from 10.1.255.2
        AS path: I, validation-state: unverified
    > to 10.1.11.2 via xe-0/2/0.0
        to 10.1.12.2 via xe-0/2/1.0
2:10.1.255.1:102::102::00:00:5e:00:01:01::10.1.102.254/304 MAC/IP
    *[EVPN/170] 2d 23:44:03
        Indirect
2:10.1.255.1:102::102::2c:6b:f5:ef:73:f0::10.1.102.1/304 MAC/IP
    *[EVPN/170] 2d 23:44:03
        Indirect
2:10.1.255.2:102::102::00:00:5e:00:01:01::10.1.102.254/304 MAC/IP
    *[BGP/170] 2d 23:43:32, localpref 100, from 10.1.255.2
        AS path: I, validation-state: unverified
    > to 10.1.11.2 via xe-0/2/0.0
        to 10.1.12.2 via xe-0/2/1.0
2:10.1.255.2:102::102::2c:6b:f5:54:95:f0::10.1.102.2/304 MAC/IP
    *[BGP/170] 2d 23:43:32, localpref 100, from 10.1.255.2
        AS path: I, validation-state: unverified
    > to 10.1.11.2 via xe-0/2/0.0
        to 10.1.12.2 via xe-0/2/1.0
2:10.1.255.112:1::102::2c:6b:f5:43:12:c0::10.1.102.101/304 MAC/IP
    *[BGP/170] 00:06:19, localpref 100, from 10.1.255.112
        AS path: I, validation-state: unverified
    > to 10.1.11.2 via xe-0/2/0.0
        to 10.1.12.2 via xe-0/2/1.0
    [BGP/170] 00:06:18, localpref 100, from 10.1.255.2
        AS path: I, validation-state: unverified
    > to 10.1.11.2 via xe-0/2/0.0
        to 10.1.12.2 via xe-0/2/1.0
3:10.1.255.1:102::102::10.1.255.1/248 IM
    *[EVPN/170] 2d 23:45:49
        Indirect
3:10.1.255.2:102::102::10.1.255.2/248 IM
    *[BGP/170] 2d 23:44:03, localpref 100, from 10.1.255.2
        AS path: I, validation-state: unverified

```

```

> to 10.1.11.2 via xe-0/2/0.0
  to 10.1.12.2 via xe-0/2/1.0
3:10.1.255.111:1::102::10.1.255.111/248 IM
*[BGP/170] 00:14:58, localpref 100, from 10.1.255.111
  AS path: I, validation-state: unverified
> to 10.1.11.2 via xe-0/2/0.0
  to 10.1.12.2 via xe-0/2/1.0
[BGP/170] 00:14:58, localpref 100, from 10.1.255.2
  AS path: I, validation-state: unverified
> to 10.1.11.2 via xe-0/2/0.0
  to 10.1.12.2 via xe-0/2/1.0
3:10.1.255.112:1::102::10.1.255.112/248 IM
*[BGP/170] 00:10:17, localpref 100, from 10.1.255.112
  AS path: I, validation-state: unverified
> to 10.1.11.2 via xe-0/2/0.0
  to 10.1.12.2 via xe-0/2/1.0
[BGP/170] 00:10:17, localpref 100, from 10.1.255.2
  AS path: I, validation-state: unverified
> to 10.1.11.2 via xe-0/2/0.0
  to 10.1.12.2 via xe-0/2/1.0

```

Display Core-1's Layer 2 address learning daemon table.

```
lab@core-1> show bridge mac-table instance VS_VLAN102
```

MAC flags (S -static MAC, D -dynamic MAC, L -locally learned, C -Control MAC
 O -OVSDB MAC, SE -Statistics enabled, NM -Non configured MAC, R -Remote PE MAC, P -Pinned
 MAC, FU - Fast Update)

Routing instance : VS_VLAN102

Bridging domain : bd102, VLAN : 102

MAC address	MAC flags	Logical interface	Active source
00:00:5e:00:01:01	DRP	esi.708	05:00:00:fd:e8:00:00:00:66:00
2c:6b:f5:43:12:c0	DR	esi.719	00:01:01:01:01:01:01:01:01:01
2c:6b:f5:54:95:f0	DR	vtep.32772	10.1.255.2

NOTE: On EX9200 switches, the `show ethernet-switching table-instance instance-name` command corresponds to the `show bridge mac-table instance instance-name` command shown here for MX Series routers

Display Core-1's kernel forwarding table.

```
lab@core-1> show route forwarding-table vpn VS_VLAN102
Routing table: VS_VLAN102.evpn-vxlan
VPLS:
Destination      Type RtRef Next hop          Type Index   NhRef Netif
default          perm   0                dscd   544     1
vtep.32772       intf   0                comp   688     7
vtep.32775       intf   0                comp   716     5
vtep.32778       intf   0                comp   722     5

Routing table: VS_VLAN102.evpn-vxlan
Bridging domain: bd102.evpn-vxlan
VPLS:
Enabled protocols: Bridging, ACKed by all peers, EVPN VXLAN,
Destination      Type RtRef Next hop          Type Index   NhRef Netif
00:00:5e:00:01:01/48 user    0                indr  1048574  2
                                comp    708     2
2c:6b:f5:43:12:c0/48 user    0                indr  1048578  3
                                comp    719     2
2c:6b:f5:54:95:f0/48 user    0                comp   688     7
0x30004/51       user    0                comp   702     2
```

Meaning

For the Tenant_B CE device, four different routes are listed for ESI 00:01:01:01:01:01:01:01:

- 1:10.1.255.111:0::010101010101010101::FFFF:FFFF/192 AD/ESI

This per-Ethernet Segment A-D Type-1 EVPN route originated from Leaf-1. The route distinguisher is obtained from global-level routing-options. Core-1 receives this Type-1 route, originated from Leaf-1, from both Leaf-1 and Leaf-2.

- 1:10.1.255.111:1::010101010101010101::0/192 AD/EVI

This is the per-EVI A-D Type-1 EVPN route. The route distinguisher is obtained from the routing instance, or in the case of QFX5100, the switch-options. Core-1 receives this Type-1 route, originated from Leaf-1, from both Leaf-1 and Leaf-2.

- 1:10.1.255.112:0:0101010101010101::FFFF:FFFF/192 AD/ESI

This is the per-Ethernet Segment A-D Type-1 EVPN route originated from Leaf-2. The route distinguisher is obtained from global-level routing-options. Core-1 receives this Type-1 route, originated from Leaf-2, from both Leaf-2 and Leaf-1.

- 1:10.1.255.112:1:0101010101010101::0/192 AD/EVI

This is the per-EVI A-D Type-1 EVPN route. The route distinguisher is obtained from the routing instance, or in the case of QFX5100, switch-options. Core-1 receives this Type-1 route, originated from Leaf-2, from both Leaf-2 and Leaf-1.

The Type-2 routes for the two physical and one virtual MAC associated with the Tenant_B multihomed CE device are originated as expected.

From the output we cannot yet determine what VTEPs are used to forward to ESI 00:01:01:01:01:01:01:01. To determine the VTEPS, display the VXLAN tunnel endpoint ESIs.

```
lab@core-1> show l2-learning vxlan-tunnel-end-point esi
```

ESI	RTT	VLNBH	INH	ESI-IFL	LOC-IFL	#RVTEPs
00:01:01:01:01:01:01:01	VS_VLAN101	718	1048577	esi.718		2

Aliasing

RVTEP-IP	RVTEP-IFL	VENH	MASK-ID	FLAGS	MAC-COUNT
10.1.255.112	vtep.32779	723	1	2	0
10.1.255.111	vtep.32774	714	0	2	0

...

NOTE: On EX9200 switches, the show ethernet-switching command corresponds to the show l2-learning command show here for MX Series routers.

The output shows active load-balancing on the VTEP interfaces to both Leaf-1 and Leaf-2 for MACs on this ESI, which validates the all-active configuration on Leaf-1 and Leaf-2.

RELATED DOCUMENTATION

[Understanding EVPN with VXLAN Data Plane Encapsulation](#) | 398

https://www.juniper.net/documentation/en_US/release-independent/nce/topics/concept/nce-evpn-vxlan-campus-arch.html

Configuring an EVPN-VXLAN Edge-Routed Bridging Overlay

IN THIS CHAPTER

- [Example: Configuring an EVPN-VXLAN Edge-Routed Bridging Overlay Within a Data Center | 617](#)
- [Example: Configuring a QFX5110 Switch as Layer 2 and 3 VXLAN Gateways in an EVPN-VXLAN Edge-Routed Bridging Overlay | 630](#)

Example: Configuring an EVPN-VXLAN Edge-Routed Bridging Overlay Within a Data Center

IN THIS SECTION

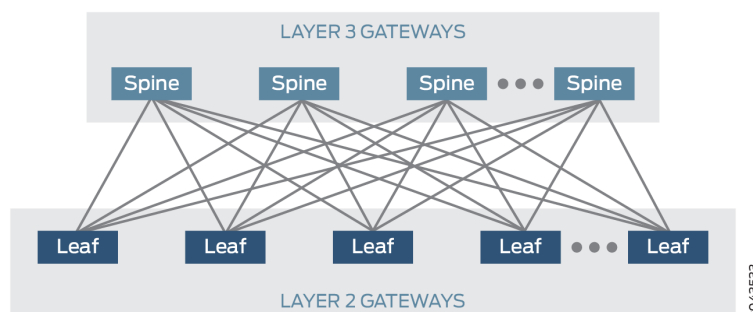
- [Requirements | 618](#)
- [Overview and Topology | 619](#)
- [Configuration | 623](#)
- [Verification | 628](#)

Ethernet VPN (EVPN) is a control plane technology that enables hosts (physical servers and virtual machines [VMs]) to be placed anywhere in a network and remain connected to the same logical Layer 2 overlay network. Virtual Extensible LAN (VXLAN) is a tunneling protocol that creates the data plane for the Layer 2 overlay network.

The physical underlay network over which EVPN-VXLAN is commonly deployed is a two-layer IP fabric, which includes spine and leaf devices as shown in [Figure 51 on page 618](#). The spine devices—for example, QFX10000 switches—provide connectivity between the leaf devices, and the leaf devices—for example, QFX5100 switches—provide connectivity to attached hosts. In the overlay network, the leaf devices function as Layer 2 gateways that handle traffic within a VXLAN, and the spine devices function

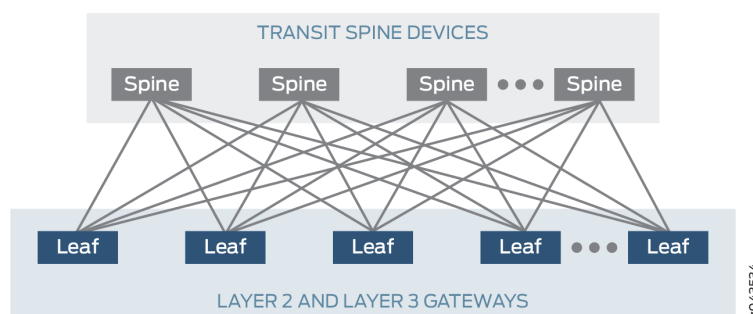
as Layer 3 gateways that handle traffic between VXLANs through the use of integrated routing and bridging (IRB) interfaces. For more information about configuring an EVPN-VXLAN centrally-routed bridging overlay (an EVPN-VXLAN topology with a two-layer IP fabric), see ["Example: Configure an EVPN-VXLAN Centrally-Routed Bridging \(CRB\) Overlay"](#) on page 530.

Figure 51: Two-Layer IP Fabric



You can also deploy EVPN-VXLAN over a physical underlay network in which the IP fabric is collapsed into a single layer of QFX10000 switches that function as leaf devices. In this fabric, which is shown in [Figure 52 on page 618](#), the leaf devices serve as both Layer 2 and Layer 3 gateways. In this topology, transit spine devices provide Layer 3 routing functionality only.

Figure 52: Collapsed IP Fabric



This example describes how to configure an EVPN-VXLAN edge-routed bridging overlay (EVPN-VXLAN topology with a collapsed IP fabric), in particular, the Layer 3 gateway, on a leaf device.

Requirements

This example uses the following hardware and software components:

- Two routers that function as transit spine devices.

- Three QFX10000 switches running Junos OS Release 15.1X53-D60 or later software. These switches are leaf devices that provide both Layer 2 and Layer 3 gateway functionality.

NOTE: This example focuses on the configuration of the Layer 2 overlay network on a leaf device. The transit spine devices used in this example provide Layer 3 functionality only. As a result, this example does not include the configuration of these devices.

Further, this example provides the configuration for leaf 1 only. The configuration for leaf 1 essentially serves as a template for the configuration of the other leaf devices. For the configuration of the other leaf devices, where appropriate, you can replace leaf 1-specific information with the information specific to the device you are configuring, add additional commands, and so on.

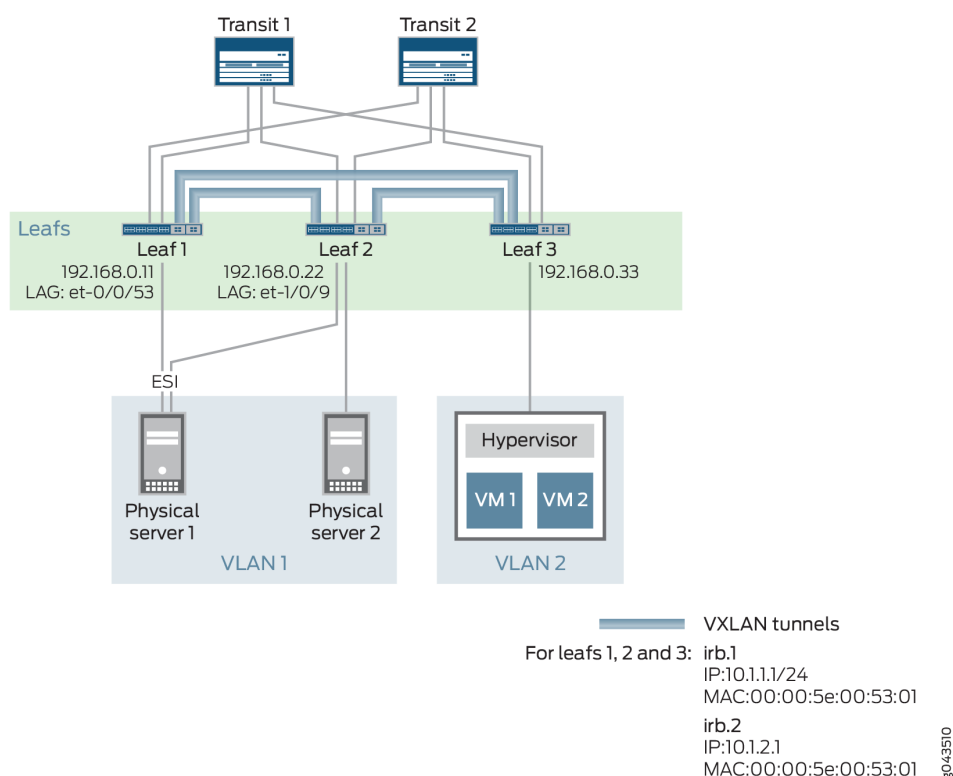
- Two physical (bare-metal) servers and one server with VMs that are supported by a hypervisor.

Overview and Topology

The edge-routed bridging overlay shown in [Figure 53 on page 620](#) includes two transit spine devices, an IP fabric, which includes three leaf devices that function as both Layer 2 and Layer 3 gateways, two physical servers, and one virtualized server on which VMs and a hypervisor are installed. Physical server 1 is connected to leaf 1 and leaf 2 through a link aggregation group (LAG) interface. On both leaf devices, the interface is assigned the same Ethernet segment identifier (ESI) and set to multihoming active-active mode.

All leaf devices are in the same autonomous system (65200).

Figure 53: Edge-Routed Bridging Overlay Within a Data Center



In this topology, an application on physical server 1 needs to communicate with VM 1 on the virtualized server. Physical servers 1 and 2 are included in VLAN 1, and the virtualized server is included in VLAN 2. For communication between VLANs 1 and 2 to occur, two IRB interfaces—irb.1, which is associated with VLAN 1, and irb.2, which is associated with VLAN 2—must be configured on each leaf device.

The most significant difference between the configuration of an edge-routed bridging overlay and a centrally-routed bridging overlay is the configuration of the Layer 3 gateway. Therefore, this example focuses on the EVPN-VXLAN configuration, in particular, the Layer 3 gateway configuration on the leaf devices.

For the edge-routed bridging overlay, you can configure the IRB interfaces within an EVPN instance using one of the following methods:

- **Method 1**—For each IRB interface on a particular leaf device, for example, leaf 1, the following is specified:
 - A unique IP address.
 - The *same* MAC address.

For example:

irb.1	IP address: 10.1.1.1/24	MAC address: 00:00:5e:00:53:01
irb.2	IP address: 10.1.2.1/24	MAC address: 00:00:5e:00:53:01

- **Method 2**—For each IRB interface on leaf 1, the following is specified:

- A unique IP address.
- A *unique* MAC address.

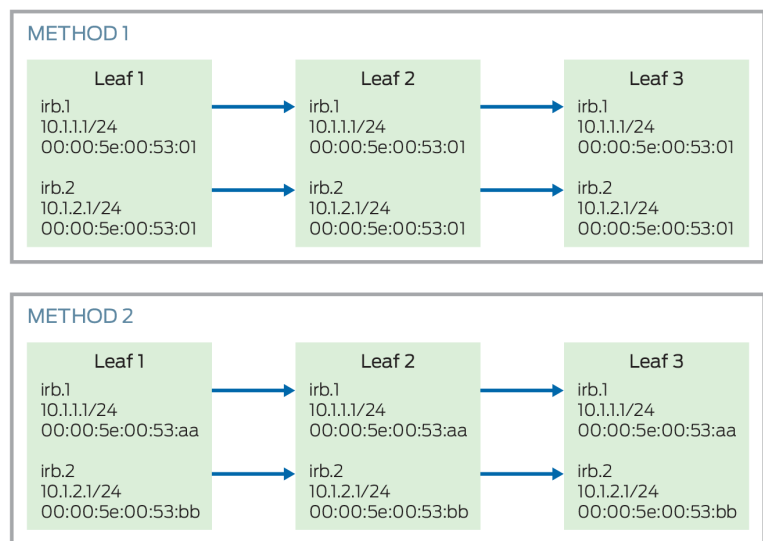
For example:

irb.1	IP address: 10.1.1.1/24	MAC address: 00:00:5e:00:53:aa
irb.2	IP address: 10.1.2.1/24	MAC address: 00:00:5e:00:53:bb

Regardless of the method that you use to configure the IRB interfaces on leaf 1, if irb.1 and irb.2 are also configured on other leafs, for example, leafs 2 and 3, you must specify the same configurations that you

specified on leaf 1 for those IRB interfaces on leafs 2 and 3. For example, [Figure 54 on page 622](#) shows the configurations for irb.1 and irb.2 on leafs 1, 2, and 3 for both methods.

Figure 54: Method 1 and 2 IRB Interface Configurations on Multiple Leaf Devices



NOTE: In this example, method 1 is used to configure the IRB interfaces.

As shown in this example, with the same MAC address configured for each IRB interface on each leaf device, each host uses the same MAC address when sending inter-VLAN traffic regardless of where the host is located or which leaf device receives the traffic. For example, in the topology shown in [Figure 53 on page 620](#), multi-homed physical server 1 in VLAN 1 sends a packet to VM 1 in VLAN 2. If leaf 1 is down, leaf 2 continues to forward the inter-VLAN traffic even without the configuration of a redundant default gateway MAC address.

Note that the configuration of IRB interfaces used in this example does not include a virtual gateway address (VGA) and a corresponding MAC address that establishes redundant default gateway functionality, which is mentioned above. By configuring the same MAC address for each IRB interface on each leaf device, hosts use the local leaf device configured with the common MAC address as the default Layer 3 gateway. Therefore, you eliminate the need to advertise a redundant default gateway and dynamically synchronize the MAC addresses of the redundant default gateway throughout the EVPN control plane. As a result, when configuring each leaf device, you must disable the advertisement of the redundant default gateway by including the `default-gateway do-not-advertise` configuration statement in the `[edit protocols evpn]` hierarchy level in your configuration.

NOTE: Although the IRB interface configuration used in this example does not include a VEA, you can configure it as needed to make EVPN-VXLAN work properly in your edge-routed bridging overlay. If you configure a VEA for each IRB interface, you specify the same IP address for each VEA on each leaf device instead of configuring the same MAC address for each IRB interface on each leaf device as is shown in this example.

When it comes to handling the replication of broadcast, unknown unicast, and multicast (BUM) traffic, note that the configuration on leaf 1:

- includes the `set protocols evpn multicast-mode ingress-replication` command. This command causes leaf 1, which is a hardware VTEP, to handle the replication and sending of BUM traffic instead of a multicast client in the EVPN-VXLAN topology.

Configuration

IN THIS SECTION

- [CLI Quick Configuration | 623](#)
- [Configuring EVPN-VXLAN on Leaf 1 | 625](#)

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

Leaf 1

```
set interfaces et-0/0/53 ether-options 802.3ad ae202
set interfaces ae202 esi 00:11:22:33:44:55:66:77:88:99
set interfaces ae202 esi all-active
set interfaces ae202 aggregated-ether-options lacp active
set interfaces ae202 aggregated-ether-options lacp system-id 00:00:00:04:04:04
set interfaces ae202 unit 0 family ethernet-switching interface-mode trunk
set interfaces ae202 unit 0 family ethernet-switching vlan members 1-2
set interfaces irb unit 1 family inet address 10.1.1.1/24
set interfaces irb unit 1 mac 00:00:5e:00:53:01
```

```

set interfaces irb unit 2 family inet address 10.1.2.1/24
set interfaces irb unit 2 mac 00:00:5e:00:53:01
set interfaces lo0 unit 0 family inet address 192.168.0.11/32
set interfaces lo0 unit 1 family inet address 192.168.10.11/32
set protocols bgp group overlay-evpn type internal
set protocols bgp group overlay-evpn local-address 192.168.0.11
set protocols bgp group overlay-evpn family evpn signaling
set protocols bgp group overlay-evpn local-as 65200
set protocols bgp group overlay-evpn multipath
set protocols bgp group overlay-evpn neighbor 192.168.0.22
set protocols bgp group overlay-evpn neighbor 192.168.0.33
set protocols evpn encapsulation vxlan
set protocols evpn extended-vni-list 1001
set protocols evpn extended- vni-list 1002
set protocols evpn multicast-mode ingress-replication
set protocols evpn default-gateway do-not-advertise
set protocols evpn vni-options vni 1001 vrf-target export target:1:1001
set protocols evpn vni-options vni 1002 vrf-target export target:1:1002
set policy-options community comm-leaf_eesi members target 9999:9999
set policy-options community com1001 members target:1:1001
set policy-options community com1002 members target:1:1002
set policy-options policy-statement LEAF-IN term import_leaf_eesi from community comm-leaf_eesi
set policy-options policy-statement LEAF-IN term import_leaf_eesi then accept
set policy-options policy-statement vrf-1-to-200 term import_vni1001 from community com1001
set policy-options policy-statement vrf-1-to-200 term import_vni1001 then accept
set policy-options policy-statement vrf-1-to-200 term import_vni1002 from community com1002
set policy-options policy-statement vrf-1-to-200 term import_vni1002 then accept
set routing-instances VRF_1 instance-type vrf
set routing-instances VRF_1 interface irb.1
set routing-instances VRF_1 interface irb.2
set routing-instances VRF_1 interface lo0.1
set routing-instances VRF_1 route-distinguisher 192.168.0.11:1
set routing-instances VRF_1 vrf-target target:1:1
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 192.168.0.11:5000
set switch-options vrf-import LEAF-IN
set switch-options vrf-import vrf-1-to-200
set switch-options vrf-target target:9999:9999
set vlans bd1 vlan-id 1
set vlans bd1 l3-interface irb.1
set vlans bd1 vxlan vni 1001
set vlans bd2 vlan-id 2

```

```
set vlans bd2 l3-interface irb.2
set vlans bd2 vxlan vni 1002
```

Configuring EVPN-VXLAN on Leaf 1

Step-by-Step Procedure

1. Enable physical server 1 to be multihomed to leaf 1 and leaf 2 by configuring an aggregated Ethernet interface, specifying an ESI for the interface, and setting the mode so that the connections to both leaf devices are active.

NOTE: When configuring the ae202 interface on leaf 2, you must specify the same ESI (00:11:22:33:44:55:66:77:88:99) that is specified for the same interface on leaf 1.

```
[edit]
user@switch# set interfaces et-0/0/53 ether-options 802.3ad ae202
user@switch# set interfaces ae202 esi 00:11:22:33:44:55:66:77:88:99
user@switch# set interfaces ae202 esi all-active
user@switch# set interfaces ae202 aggregated-ether-options lacp active
user@switch# set interfaces ae202 aggregated-ether-options lacp system-id 00:00:00:04:04:04
user@switch# set interfaces ae202 unit 0 family ethernet-switching interface-mode trunk
user@switch# set interfaces ae202 unit 0 family ethernet-switching vlan members 1-2
```

2. Configure two IRB interfaces, each with unique IP addresses and the same MAC address.

```
[edit]
user@switch# set interfaces irb unit 1 family inet address 10.1.1.1/24
user@switch# set interfaces irb unit 1 mac 00:00:5e:00:53:01
user@switch# set interfaces irb unit 2 family inet address 10.1.2.1/24
user@switch# set interfaces irb unit 2 mac 00:00:5e:00:53:01
```

3. Configure a loopback interface (lo0.0) for the leaf device and a logical loopback address (lo0.1) for the EVPN routing instance (VRF-1).

```
[edit]
user@switch# set interfaces lo0 unit 0 family inet address 192.168.0.11/32
user@switch# set interfaces lo0 unit 1 family inet address 192.168.10.11/32
```

4. Set up the IBGP overlay network.

```
[edit]
user@switch# set protocols bgp group overlay-evpn type internal
user@switch# set protocols bgp group overlay-evpn local-address 192.168.0.11
user@switch# set protocols bgp group overlay-evpn family evpn signaling
user@switch# set protocols bgp group overlay-evpn local-as 65200
user@switch# set protocols bgp group overlay-evpn multipath
user@switch# set protocols bgp group overlay-evpn neighbor 192.168.0.22
user@switch# set protocols bgp group overlay-evpn neighbor 192.168.0.33
```

5. Set up the EVPN-VXLAN domain, which entails determining which VNIs are included in the domain, specifying that leaf 1, which is a hardware VTEP, handles the replication and sending of BUM traffic, disabling the advertisement of the redundant default gateway throughout the EVPN control plane, and specifying a route target for each VNI.

```
[edit]
user@switch# set protocols evpn encapsulation vxlan
user@switch# set protocols evpn extended-vni-list 1001
user@switch# set protocols evpn extended- vni-list 1002
user@switch# set protocols evpn multicast-mode ingress-replication
user@switch# set protocols evpn default-gateway do-not-advertise
user@switch# set protocols evpn vni-options vni 1001 vrf-target export target:1:1001
user@switch# set protocols evpn vni-options vni 1002 vrf-target export target:1:1002
```

6. Set up communities for the VNIs, and create policies that import and accept the overlay routes.

```
[edit]
user@switch# set policy-options community comm-leaf_esi members target 9999:9999
user@switch# set policy-options community com1001 members target:1:1001
user@switch# set policy-options community com1002 members target:1:1002
user@switch# set policy-options policy-statement LEAF-IN term import_leaf_esi from community
comm-leaf_esi
user@switch# set policy-options policy-statement LEAF-IN term import_leaf_esi then accept
user@switch# set policy-options policy-statement vrf-1-to-200 term import_vni1001 from
community com1001
user@switch# set policy-options policy-statement vrf-1-to-200 term import_vni1001 then accept
user@switch# set policy-options policy-statement vrf-1-to-200 term import_vni1002 from
```

```
community com1002
user@switch# set policy-options policy-statement vrf-1-to-200 term import_vni1002 then accept
```

7. Set up an EVPN routing instance.

```
[edit]
user@switch# set routing-instances VRF_1 instance-type vrf
user@switch# set routing-instances VRF_1 interface irb.1
user@switch# set routing-instances VRF_1 interface irb.2
user@switch# set routing-instances VRF_1 interface lo0.1
user@switch# set routing-instances VRF_1 route-distinguisher 192.168.0.11:1
user@switch# set routing-instances VRF_1 vrf-target target:1:1
```

NOTE: In the above EVPN routing instance configuration, a unique logical loopback interface (lo0.1) is specified, and an IP address for the interface is specified using the `set interfaces lo0 unit logical-unit-number family inet address ip-address/prefix` command. All items configured in the above routing instance except for the logical loopback interface are required for EVPN. However, the configuration of a logical loopback interface and associated IP address are required to ensure that VXLAN control packets are properly processed.

8. Configure the switch options to use loopback interface lo0.0 as the source interface of the VTEP, set a route distinguisher, and import the route targets for the three communities into the EVPN (MAC) table.

```
[edit]
user@switch# set switch-options vtep-source-interface lo0.0
user@switch# set switch-options route-distinguisher 192.168.0.11:5000
user@switch# set switch-options vrf-import LEAF-IN
user@switch# set switch-options vrf-import vrf-1-to-200
user@switch# set switch-options vrf-target target:9999:9999
```

9. Configure VLANs to which IRB interfaces and VXLAN VNIs are associated.

```
[edit]
user@switch# set vlans bd1 vlan-id 1
user@switch# set vlans bd1 l3-interface irb.1
user@switch# set vlans bd1 vxlan vni 1001
user@switch# set vlans bd2 vlan-id 2
```

```
user@switch# set vlans bd2 l3-interface irb.2
user@switch# set vlans bd2 vxlan vni 1002
```

Verification

IN THIS SECTION

- [Verifying the IRB Interfaces | 628](#)
- [Verifying the VTEP Interfaces | 629](#)
- [Verifying the EVPN Routing Instance | 629](#)

The section describes the following verifications for this example:

Verifying the IRB Interfaces

Purpose

Verify that the IRB interfaces are up and running.

Action

Display the status of the IRB interfaces:

```
user@leaf1> show interfaces irb terse
```

Interface	Admin	Link	Proto	Local	Remote
irb	up	up			
irb.1	up	up	inet	10.1.1.1/24	
irb.2	up	up	inet	10.1.2.1/24	

Meaning

The IRB interfaces are up and running.

Verifying the VTEP Interfaces

Purpose

Verify the status of the VTEP interfaces.

Action

Display the status of the VTEP interfaces:

```
user@leaf1> show interfaces vtep terse
Interface          Admin Link Proto  Local          Remote
vtep               up    up
vtep.32769         up    up  eth-switch
vtep.32770         up    up  eth-switch
vtep.32771         up    up  eth-switch
```

Meaning

The interfaces for each of the VTEPs is up. Therefore, the VTEP interfaces are functioning normally.

Verifying the EVPN Routing Instance

Purpose

Verify the routing table for VRF_1.

Action

Verify the routing table for the EVPN routing instance VRF_1.

```
user@leaf1> show route table VRF_1.inet.0
VRF_1.inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.1.1.0/24        *[Direct/0] 00:07:38
                   > via irb.1
10.1.1.1/32        *[Local/0] 00:07:38
                   Local via irb.110.1.2.0/24    *[Direct/0] 00:07:38
                   > via irb.2
```



```

10.1.2.1/32      *[Local/0] 00:07:38
                  Local via irb.2
192.168.10.11/32 *[Direct/0] 00:07:38
                  > via lo0.1

```

Meaning

The EVPN routing instance is functioning correctly.

Example: Configuring a QFX5110 Switch as Layer 2 and 3 VXLAN Gateways in an EVPN-VXLAN Edge-Routed Bridging Overlay

IN THIS SECTION

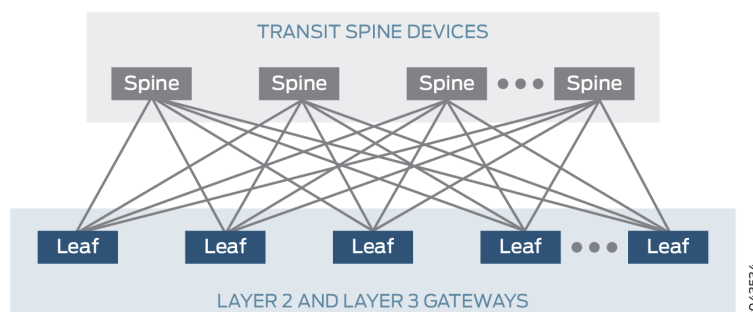
- [Requirements | 631](#)
- [Overview and Topology | 632](#)
- [Basic Underlay Network Configuration | 634](#)
- [Basic EVPN-VXLAN Overlay Network Configuration | 635](#)
- [Basic Customer Profile Configuration | 637](#)
- [Route Leaking Configuration | 640](#)

Ethernet VPN (EVPN) is a control plane technology that enables hosts (physical [bare-metal] servers and virtual machines [VMs]) to be placed anywhere in a network and remain connected to the same logical Layer 2 overlay network. Virtual Extensible LAN (VXLAN) is a tunneling protocol that creates the data plane for the Layer 2 overlay network.

You can deploy EVPN-VXLAN over a physical underlay network in which the IP fabric is collapsed into a single layer of QFX5110 switches that function as leaf devices. As shown in [Figure 55 on page 631](#), the leaf devices serve as both Layer 2 and Layer 3 VXLAN gateways. In the EVPN-VXLAN edge-routed bridging overlay (EVPN-VXLAN topology with a collapsed IP fabric), Layer 2 VXLAN gateways handle traffic within a VLAN, and Layer 3 VXLAN gateways handle traffic between VLANs using integrated routing and bridging (IRB) interfaces.

Figure 55 on page 631 also shows transit spine devices, which provide Layer 3 routing functionality only.

Figure 55: Single Layer of Leaf Devices



Starting with Junos OS Release 17.3R1, the QFX5110 switch can function as a leaf device, which acts as Layer 2 and 3 VXLAN gateways in an EVPN-VXLAN edge-routed bridging overlay.

This topic provides a sample configuration of a QFX5110 switch that functions as a leaf device in an edge-routed bridging overlay.

Requirements

This example uses the following hardware and software components:

- Two routers that function as transit spine devices.
- Three QFX5110 switches running Junos OS Release 17.3R1 or later. These switches act as leaf devices (leaf 1, leaf 2, and leaf 3) that provide Layer 2 and 3 VXLAN gateway functionality.

NOTE: This example focuses on the configuration of the QFX5110 switch that functions as leaf 1. A basic configuration is provided for the IP/BGP underlay network, the EVPN-VXLAN overlay network, a customer-specific profile, and route leaking. This example does not include all features that can be used in an EVPN-VXLAN network. The configuration for leaf 1 essentially serves as a template for the configuration of the other leaf devices. For the configuration of the other leaf devices, where appropriate, you can replace leaf 1-specific information with the information specific to the device you are configuring, add additional commands, and so on.

- Two physical servers and one virtualized server with VMs that are supported by a hypervisor.

Overview and Topology

In this example, a service provider supports ABC Corporation, which has multiple sites. Physical servers in site 100 must communicate with VMs in site 200. To enable this communication in the edge-routed bridging overlay shown in [Figure 56 on page 632](#), you configure the key software entities in [Table 29 on page 632](#) on the QFX5110 switches that function as Layer 2 and 3 VXLAN gateways, or leaf devices.

Figure 56: Sample Edge-Routed Bridging Overlay

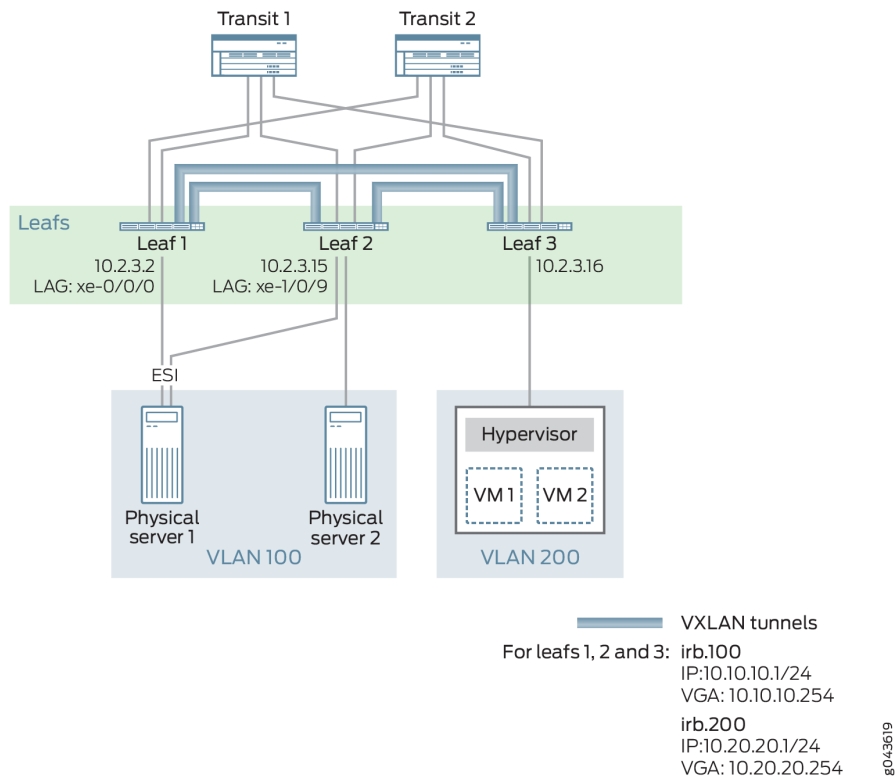


Table 29: Layer 3 Inter-VLAN Routing Entities Configured on Leaf 1, Leaf 2, and Leaf 3

Entities	Configuration on Leaf 1, Leaf 2, and Leaf 3
VLANs	v100
	v200

Table 29: Layer 3 Inter-VLAN Routing Entities Configured on Leaf 1, Leaf 2, and Leaf 3 (Continued)

Entities	Configuration on Leaf 1, Leaf 2, and Leaf 3
VRF instances	vrf_vlan100 vrf_vlan200
IRB interfaces	<div> irb.100 10.10.10.1/24 (IRB IP address) 10.10.10.254 (virtual gateway address) </div> <div> irb.200 10.20.20.1/24 (IRB IP address) 10.20.20.254 (virtual gateway address) </div>

As outlined in [Table 29 on page 632](#), you configure VLAN v100 for site 100 and VLAN v200 for site 200 on each leaf device. To segregate the Layer 3 routes for VLANs v100 and v200, you create VPN routing and forwarding (VRF) instances vrf_vlan100 and vrf_vlan200 on each leaf device. To route traffic between the VLANs, you configure IRB interfaces irb.100 and irb.200, and associate VRF instance vrf_vlan100 with IRB interface irb.100, and VRF instance vrf_vlan200 with IRB interface irb.200.

The physical servers in VLAN v100 are non-virtualized. As a result, we strongly recommend that you configure IRB interfaces irb.100 and irb.200 to function as default Layer 3 gateways that handle the inter-VLAN traffic of the physical servers. To that end, the configuration of each IRB interface also includes a virtual gateway address (VGA), which configures an IRB interface as a default Layer 3 gateway. In addition, this example assumes that each physical server is configured to use a particular default gateway. For more information about default gateways and how inter-VLAN traffic flows between a physical server to another physical server or VM in another VLAN in an edge-routed bridging overlay, see ["Using a Default Layer 3 Gateway to Route Traffic in an EVPN-VXLAN Overlay Network" on page 508](#).

NOTE: When configuring a VGA for an IRB interface, keep in mind that the IRB IP address and VGA must be different.

As outlined in [Table 29 on page 632](#), a separate VRF routing instance is configured for each VLAN. To enable the communication between hosts in VLANs v100 and v200, this example shows how to export

unicast routes from the routing table for vrf_vlan100 and import the routes into the routing table for vrf_vlan200 and vice versa. This feature is also known as route leaking.

Basic Underlay Network Configuration

IN THIS SECTION

- [CLI Quick Configuration | 634](#)
- [Configuring the Underlay Network | 634](#)

CLI Quick Configuration

To quickly configure a basic underlay network, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

```
set routing-options router-id 10.2.3.2
set routing-options autonomous-system 64500
set protocols bgp group pe neighbor 10.2.3.15
set protocols bgp group pe neighbor 10.2.3.16
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface em0.0 disable
set protocols ospf area 0.0.0.0 interface lo0.0 passive
```

Configuring the Underlay Network

Step-by-Step Procedure

To configure a basic underlay network on leaf 1:

1. Configure the router ID and autonomous system number for leaf 1.

```
[edit routing-options]
user@switch# set router-id 10.2.3.2
user@switch# set autonomous-system 64500
```

2. Configure a BGP group that includes leaf 2 and leaf 3 as peers that also handle underlay functions.

```
[edit protocols]
user@switch# set bgp group pe neighbor 10.2.3.15
user@switch# set bgp group pe neighbor 10.2.3.16
```

3. Configure OSPF as the routing protocol for the underlay network.

```
[edit protocols]
user@switch# set ospf area 0.0.0.0 interface all
user@switch# set ospf area 0.0.0.0 interface em0.0 disable
user@switch# set ospf area 0.0.0.0 interface lo0.0 passive
```

Basic EVPN-VXLAN Overlay Network Configuration

IN THIS SECTION

- [CLI Quick Configuration | 635](#)
- [Configuring a Basic EVPN-VXLAN Underlay Network | 636](#)

CLI Quick Configuration

To quickly configure a basic overlay network, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

```
set forwarding-options vxlan-routing interface-num 8192
set forwarding-options vxlan-routing next-hop 16384
set protocols bgp group pe type internal
set protocols bgp group pe local-address 10.2.3.2
set protocols bgp group pe family evpn signaling
set protocols evpn encapsulation vxlan
set protocols evpn extended-vni-list all
set protocols evpn default-gateway no-gateway-community
set switch-options route-distinguisher 10.2.3.2:1
```

```
set switch-options vrf-target target:1111:11
set switch-options vtep-source-interface lo0.0
```

Configuring a Basic EVPN-VXLAN Underlay Network

Step-by-Step Procedure

To configure a basic EVPN-VXLAN overlay network on leaf 1:

1. Increase the number of physical interfaces and next hops that the QFX5110 switch allocates for use in an EVPN-VXLAN topology.

```
[edit forwarding-options]
set vxlan-routing interface-num 8192
set vxlan-routing next-hop 16384
```

2. Configure an IBGP overlay between leaf 1 and the other two leaf devices, specify a local IP address for leaf 1, and include the EVPN signaling Network Layer Reachability Information (NLRI) to the BGP group.

```
[edit protocols]
user@switch# set bgp group pe type internal
user@switch# set bgp group pe local-address 10.2.3.2
user@switch# set bgp group pe family evpn signaling
```

3. Configure VXLAN encapsulation for the data packets exchanged between the EVPN neighbors, and specify that all VXLAN network identifiers (VNIs) are part of the virtual routing and forwarding (VRF) instance. Also, specify that the MAC address of the IRB interface and the MAC address of the corresponding default gateway are advertised without the extended community option of default - gateway.

```
[edit protocols]
user@switch# set evpn encapsulation vxlan
user@switch# set evpn extended-vni-list all
user@switch# set evpn default-gateway no-gateway-community
```

4. Configure switch options to set a route distinguisher and VRF target for the VRF routing instance, and associate interface lo0 with the virtual tunnel endpoint (VTEP).

```
[edit switch-options]
user@switch# set route-distinguisher 10.2.3.2:1
user@switch# set vrf-target target:1111:11
user@switch# set vtep-source-interface lo0.0
```

Basic Customer Profile Configuration

IN THIS SECTION

- [CLI Quick Configuration | 637](#)
- [Configuring a Basic Customer Profile | 638](#)

CLI Quick Configuration

To quickly configure a basic customer profile for ABC Corporation sites 100 and 200, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

```
set interfaces xe-0/0/0 ether-options 802.3ad ae0
set interfaces ae0 esi 00:00:00:ab:cd:00:01:00:00:01
set interfaces ae0 esi all-active
set interfaces xe-0/0/10 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/10 unit 0 family ethernet-switching vlan members v100
set interfaces xe-0/0/11 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/11 unit 0 family ethernet-switching vlan members v200
set interfaces irb unit 100 family inet address 10.10.10.1/24 virtual-gateway-address 10.10.10.254
set interfaces irb unit 200 family inet address 10.20.20.1/24 virtual-gateway-address 10.20.20.254
set interfaces lo0 unit 0 family inet address 10.2.3.2/32 primary
set interfaces lo0 unit 1 family inet address 10.2.3.24/32 primary
set interfaces lo0 unit 2 family inet address 10.2.3.25/32 primary
set routing-instances vrf_vlan100 instance-type vrf
set routing-instances vrf_vlan100 interface irb.100
```



```

set routing-instances vrf_vlan100 interface lo0.1
set routing-instances vrf_vlan100 route-distinguisher 10.2.3.11:2
set routing-instances vrf_vlan200 instance-type vrf
set routing-instances vrf_vlan200 interface irb.200
set routing-instances vrf_vlan200 interface lo0.2
set routing-instances vrf_vlan200 route-distinguisher 10.2.3.11:3
set vlans v100 vlan-id 100
set vlans v100 l3-interface irb.100
set vlans v100 vxlan vni 100
set vlans v200 vlan-id 200
set vlans v200 l3-interface irb.200
set vlans v200 vxlan vni 200

```

Configuring a Basic Customer Profile

Step-by-Step Procedure

To configure a basic customer profile for ABC Corporation sites 100 and 200 on leaf 1:

1. Enable physical server 1 to be multihomed to leaf 1 and leaf 2 by configuring an aggregated Ethernet interface, specifying an ESI for the interface, and setting the mode so that the connections to both leaf devices are active.

```

[edit interfaces]
user@switch# set xe-0/0/0 ether-options 802.3ad ae0
user@switch# set ae0 esi 00:11:22:33:44:55:66:77:88:99
user@switch# set ae0 esi all-active

```

NOTE: When configuring the ae202 interface on leaf 2, you must specify the same ESI (00:11:22:33:44:55:66:77:88:99) that is specified for the same interface on leaf 1.

2. Configure Layer 2 interfaces, and specify each interface as a member of VLAN v100 or v200.

```

[edit interfaces]
user@switch# set xe-0/0/10 unit 0 family ethernet-switching interface-mode trunk
user@switch# set xe-0/0/10 unit 0 family ethernet-switching vlan members v100
user@switch# set xe-0/0/11 unit 0 family ethernet-switching interface-mode trunk
user@switch# set xe-0/0/11 unit 0 family ethernet-switching vlan members v200

```

3. Configure IRB interfaces and associated VGAs (default Layer 3 virtual gateways), which enable the communication between physical servers, or physical servers and VMs, in different VLANs.

```
[edit interfaces]
user@switch# set irb unit 100 family inet address 10.10.10.1/24 virtual-gateway-address 10.10.10.254
user@switch# set irb unit 200 family inet address 10.20.20.1/24 virtual-gateway-address 10.20.20.254
```

NOTE: When configuring a VGA for an IRB interface, keep in mind that the IRB IP address and VGA must be different.

4. Configure a loopback interface (lo0) for leaf 1 and a logical loopback address (lo0.x) for each VRF routing instance.

```
[edit interfaces]
user@switch# set lo0 unit 0 family inet address 10.2.3.2/32 primary
user@switch# set lo0 unit 1 family inet address 10.2.3.24/32 primary
user@switch# set lo0 unit 2 family inet address 10.2.3.25/32 primary
```

5. Configure a VRF routing instance for VLAN v100 and another VRF routing instance for VLAN v200. In each routing instance, associate an IRB interface, a loopback interface, and an identifier attached to the route.

```
[edit routing-instances]
user@switch# set vrf_vlan100 instance-type vrf
user@switch# set vrf_vlan100 interface irb.100
user@switch# set vrf_vlan100 interface lo0.1
user@switch# set vrf_vlan100 route-distinguisher 10.2.3.2:2
user@switch# set vrf_vlan200 instance-type vrf
user@switch# set vrf_vlan200 interface irb.200
user@switch# set vrf_vlan200 interface lo0.2
user@switch# set vrf_vlan200 route-distinguisher 10.2.3.2:3
```

6. Configure VLANs v100 and v200, and associate an IRB interface and VNI with each VLAN.

```
[edit vlans]
user@switch# set v100 vlan-id 100
user@switch# set v100 l3-interface irb.100
user@switch# set v100 vxlan vni 100
user@switch# set v200 vlan-id 200
user@switch# set v200 l3-interface irb.200
user@switch# set v200 vxlan vni 200
```

Route Leaking Configuration

IN THIS SECTION

- [CLI Quick Configuration | 640](#)
- [Configuring Route Leaking | 641](#)

CLI Quick Configuration

To quickly configure route leaking, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

```
set policy-options policy-statement export-inet1 term 1 from interface irb.100
set policy-options policy-statement export-inet1 term 1 then community add com200
set policy-options policy-statement export-inet1 term 1 then accept
set policy-options policy-statement export-inet2 term 1 from interface irb.200
set policy-options policy-statement export-inet2 term 1 then community add com100
set policy-options policy-statement export-inet2 term 1 then accept
set policy-options policy-statement import-inet term 1 from community com100
set policy-options policy-statement import-inet term 1 from community com200
set policy-options policy-statement import-inet term 1 then accept
set policy-options community com100 members target:1:100
set policy-options community com200 members target:1:200
set routing-instances vrf_vlan100 vrf-import import-inet
set routing-instances vrf_vlan100 vrf-export export-inet1
set routing-instances vrf_vlan100 routing-options auto-export family inet unicast
set routing-instances vrf_vlan200 vrf-import import-inet
```

```
set routing-instances vrf_vlan200 vrf-export export-inet2
set routing-instances vrf_vlan200 routing-options auto-export family inet unicast
```

Configuring Route Leaking

Step-by-Step Procedure

To configure route leaking on leaf 1:

1. Configure a routing policy that specifies that routes learned through IRB interface irb.100 are exported and then imported into the routing table for vrf_vlan200. Configure another routing policy that specifies that routes learned through IRB interface irb.200 are exported and then imported into the routing table for vrf_vlan100.

```
[edit policy-options]
user@switch# set policy-statement export-inet1 term 1 from interface irb.100
user@switch# set policy-statement export-inet1 term 1 then community add com200
user@switch# set policy-statement export-inet1 term 1 then accept
user@switch# set policy-statement export-inet2 term 1 from interface irb.200
user@switch# set policy-statement export-inet2 term 1 then community add com100
user@switch# set policy-statement export-inet2 term 1 then accept
user@switch# set policy-statement import-inet term 1 from community com100
user@switch# set policy-statement import-inet term 1 from community com200
user@switch# set policy-statement import-inet term 1 then accept
user@switch# set community com100 members target:1:100
user@switch# set community com200 members target:1:200
```

2. In the VRF routing instances for VLANs v100 and v200, apply the routing policies configured in step 1.

```
[edit routing-instances]
user@switch# set vrf_vlan100 vrf-import import-inet
user@switch# set vrf_vlan100 vrf-export export-inet1
user@switch# set vrf_vlan200 vrf-import import-inet
user@switch# set vrf_vlan200 vrf-export export-inet2
```

- 3. Specify that unicast routes are to be exported from the vrf_vlan100 routing table into the vrf_vlan200 routing table and vice versa.

```
[edit routing-instances]
user@switch# set vrf_vlan100 routing-options auto-export family inet unicast
user@switch# set vrf_vlan200 routing-options auto-export family inet unicast
```

Release History Table

Release	Description
17.3R1	Starting with Junos OS Release 17.3R1, the QFX5110 switch can function as a leaf device, which acts as Layer 2 and 3 VXLAN gateways in an EVPN-VXLAN edge-routed bridging overlay.

RELATED DOCUMENTATION

[Example: Configuring a QFX5110 Switch as a Layer 3 VXLAN Gateway in an EVPN-VXLAN Centrally-Routed Bridging Overlay | 566](#)

IPv6 Underlay for VXLAN Overlays

IN THIS CHAPTER

- [EVPN-VXLAN with an IPv6 Underlay | 643](#)
- [Example: Configure an IPv6 Underlay for Layer 2 VXLAN Gateway Leaf Devices | 649](#)

EVPN-VXLAN with an IPv6 Underlay

SUMMARY

This topic describes how to set up an IPv6 underlay for the VXLAN overlay tunneling in an EVPN-VXLAN fabric.

IN THIS SECTION

- [IPv6 Underlay Support in EVPN-VXLAN Fabrics | 643](#)
- [Configure an IPv6 Underlay with EVPN-VXLAN | 648](#)

IPv6 Underlay Support in EVPN-VXLAN Fabrics

IN THIS SECTION

- [Benefits of Using an IPv6 Underlay with a VXLAN Overlay | 644](#)
- [Overview | 644](#)
- [Underlay Routing Protocols with an IPv6 Underlay | 646](#)
- [EVPN-VXLAN Features Supported with an IPv6 Underlay | 646](#)
- [Limitations in IPv6 Underlay Support | 647](#)

Ethernet VPNs (EVPNs) connect devices with Layer 2 virtual bridges. Virtual Extensible LANs (VXLANs) establish overlay tunnels that stretch the Layer 2 connections over a Layer 3 network. In EVPN-VXLAN network configurations, a leaf or spine device can function as a VXLAN gateway at Layer 2, Layer 3, or both layers. The underlay network for the VXLAN overlay can be an IPv4 or an IPv6 network. This topic describes using an IPv6 underlay instead of an IPv4 underlay.

Benefits of Using an IPv6 Underlay with a VXLAN Overlay

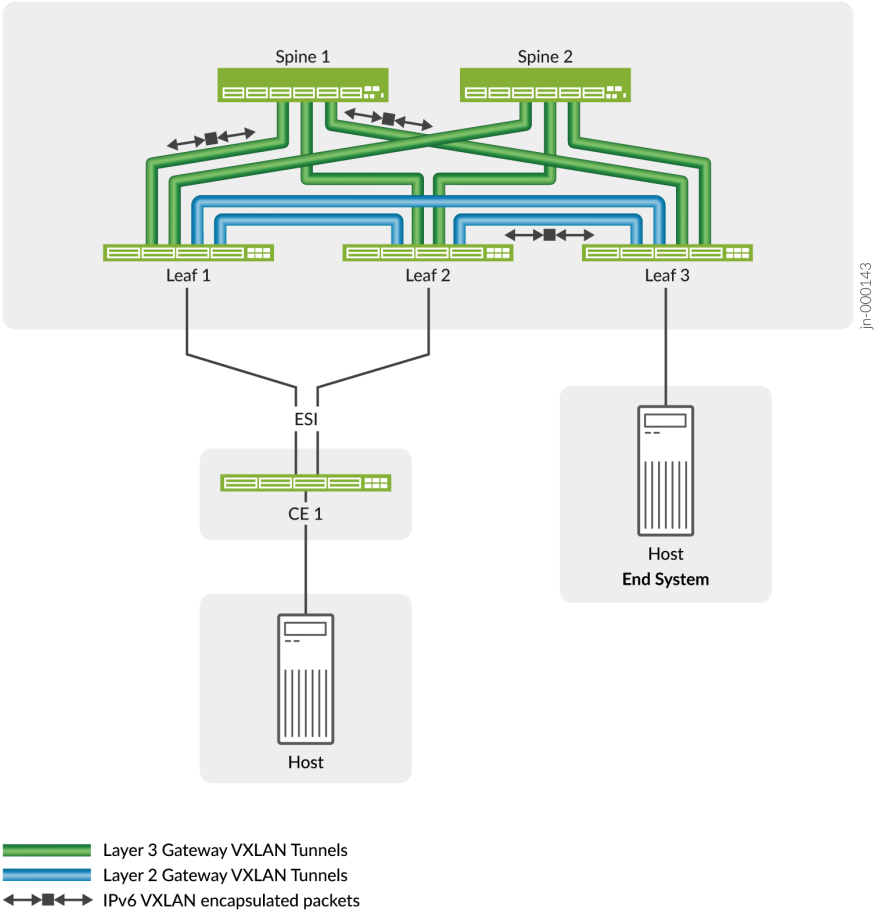
- With an IPv6 underlay VXLAN tunnel configuration, you can take advantage of the expanded addressing capabilities and efficient packet processing that the IPv6 protocol offers.

Overview

In EVPN-VXLAN installations, you configure a VXLAN overlay on Layer 2 or Layer 3 VXLAN gateway devices called virtual tunnel endpoints (VTEPs). The VXLAN overlay extends virtual tunnels between

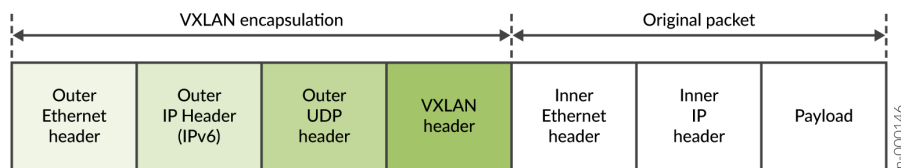
VTEPs over the underlying IP fabric. On supporting platforms, you can configure the IP underlay with IPv6 addressing to support the VXLAN overlay tunnels. For example:

Figure 57: EVPN-VXLAN Fabric with an IPv6 Underlay



When you use an IPv6 underlay, the VTEPs encapsulate VXLAN packets with an IPv6 outer header and tunnel the packets through an IPv6 underlay network.

Figure 58: IPv6 Underlay VXLAN Packet Encapsulation



You can configure an IPv6 underlay for the VXLAN overlays in an EVPN-VXLAN fabric starting in:

- Junos OS Release 17.3R1 on MX Series routers.
- Junos OS Release 21.2R2 and 21.4R1 on QFX Series switches (QFX5120 switches and the QFX10000 line of switches).

IPv6 underlay configurations are similar to IPv4 underlay configurations, except you set the VTEP source addresses as IPv6 addresses. You also assign IPv6 addresses in the underlay and establish reachability using the IPv6 protocol.

Underlay Routing Protocols with an IPv6 Underlay

We've qualified an IPv6 underlay with the following routing protocols in the underlay configuration:

- BGP—Internal BGP (iBGP) and external BGP (eBGP)
- OSPFv3—Open Shortest Path First (OSPF) routing protocol for IPv6

EVPN-VXLAN Features Supported with an IPv6 Underlay

We support the following EVPN-VXLAN features with an IPv6 underlay:

- (All devices) EVPN Type 1, Type 2, Type 3, and Type 4 routes.

(QFX Series switches only) EVPN Type 5 routes.

See ["EVPN Type 5 Route with VXLAN encapsulation for EVPN-VXLAN" on page 21](#) for more about these EVPN route types.

- (QFX Series switches only) Shared VTEP tunnels.

When you use multiple MAC-VRF instances on an EVPN-VXLAN device, to avoid VTEP scaling issues, we require that you enable the shared tunnels feature. On QFX series switches, we support an IPv6 underlay only with MAC-VRF routing instances. As a result, you must enable shared tunnels when you configure an IPv6 underlay on QFX series switches that don't have shared tunnels enabled by default. See ["MAC-VRF Routing Instance Type Overview" on page 459](#) for more about MAC-VRF instances.

- (All devices) VLAN-aware bundle, VLAN-based, and VLAN bundle Ethernet service types.

(MX Series routers, additionally) Port-based service.

See ["Understanding VLAN-Aware Bundle and VLAN-Based Service for EVPN" on page 497](#) for more about these service types.

- All-active multihoming.

See ["EVPN Multihoming Overview" on page 96](#).

- EVPN core isolation.

See ["Understanding When to Disable EVPN-VXLAN Core Isolation" on page 329](#).

- Bridged overlays.

See [Bridged Overlay Design and Implementation](#).

- Layer 3 gateway functions in ERB and CRB overlays with IPv4 or IPv6 traffic.

- Underlay and overlay load balancing.

See ["Load Balancing VXLAN Traffic" on page 421](#) and ["Dynamic Load Balancing in an EVPN-VXLAN Network" on page 455](#).

- Layer 3 protocols over IRB interfaces—BFD, BGP, OSPF.

See ["Supported Protocols on an IRB Interface in EVPN-VXLAN " on page 527](#).

- DCI—Over-the-top (OTT) full mesh DCI only.

See [Over-the-Top Data Center Interconnect in an EVPN Network](#) for details on the OTT DCI method.

- EVPN proxy ARP and ARP suppression, as well as EVPN proxy NDP and NDP suppression.

See ["EVPN Proxy ARP and ARP Suppression, and Proxy NDP and NDP Suppression" on page 361](#) for more information on these features.

Limitations in IPv6 Underlay Support

Note the following limitations in IPv6 underlay support:

- You can't mix IPv4 and IPv6 underlay configurations for the VXLAN overlays across the EVPN instances in the same fabric.
- We don't support the Open vSwitch database (OVSDb) management protocol for IPv6 underlays.
- (MX Series routers and EX9200 switches) We don't support EVPN Type 5 routes with an IPv6 underlay.
- (QFX10002-60C switches) You can only use enterprise style interface configuration; we don't support service provider style interface configuration and Q-in-Q tunneling with IPv4 or IPv6 underlays on these switches.
- (QFX Series switches) You must use MAC-VRF routing instances with EVPN protocol and VXLAN encapsulation. We don't support IPv6 underlays with other instance types such as evpn, evpn-vpws, virtual-switch or the default switching instance.
- We don't support IPv6 underlays with:
 - EVPN multicast and multicast optimization features.
 - DCI for EVPN-VXLAN in the data center to EVPN-VXLAN in a WAN using the gateway interconnection model.

Configure an IPv6 Underlay with EVPN-VXLAN

This section describes the key steps to configure the IP underlay for the VXLAN tunnels in an EVPN-VXLAN fabric to use the IPv6 protocol (instead of an IPv4 underlay). You can use an IPv6 underlay in many different EVPN-VXLAN configurations and use cases. See ["Example: Configure an IPv6 Underlay for Layer 2 VXLAN Gateway Leaf Devices" on page 649](#) for a simple example that uses OSPFv3 in the underlay and iBGP for the overlay connectivity.

Keep the following configuration options and requirements in mind:

- You can configure your EVPN-VXLAN fabric with any of the underlay routing protocols that support IPv6 underlays. See ["Underlay Routing Protocols with an IPv6 Underlay" on page 646](#).
- QFX Series switches support IPv6 underlays only with MAC-VRF routing instances in EVPN-VXLAN fabrics. You must configure your EVPN instances with VXLAN encapsulation in all MAC-VRF routing instances. The routing instances mentioned in the steps here are always EVPN-VXLAN MAC-VRF instances. See ["MAC-VRF Routing Instance Type Overview" on page 459](#) for more about MAC-VRF instances.

To enable an IPv6 underlay for VXLAN tunneling, include these items in your EVPN-VXLAN fabric configuration:

1. Assign IPv4 and IPv6 addresses to the loopback interface on the devices that serve as Layer 2 or Layer 3 VXLAN gateway VTEPs.

2. Configure the underlay and EVPN instance device connectivity using any of the supported routing protocols with IPv6 addressing. (See ["Underlay Routing Protocols with an IPv6 Underlay" on page 646.](#))
3. In the EVPN routing instance, configure the underlay VTEP source interface as the device's loopback IPv6 address. On a QFX Series switch, for example:

```
set routing-instances instance-name vtep-source-interface lo0.0 inet6
```

4. The underlay uses the IPv6 address family. However, for BGP handshaking to work in the overlay, you must configure the router ID as the loopback IPv4 address:

```
set routing-options router-id ipv4-address
```

5. (QFX Series Broadcom-based switches in Junos OS Release 21.2R2 only) Enable the Broadcom VXLAN flexible flow feature. You don't need this step starting in Junos OS Release 21.4R1, where the default configuration enables this option for you on any affected platforms. After you set this option, you must reboot the device for the change to take effect.

```
set forwarding-options vxlan-flexflow
```

SEE ALSO

[Understanding EVPN with VXLAN Data Plane Encapsulation | 398](#)

[Understanding VXLANs | 414](#)

[Example: Configure an IPv6 Underlay for Layer 2 VXLAN Gateway Leaf Devices | 649](#)

Example: Configure an IPv6 Underlay for Layer 2 VXLAN Gateway Leaf Devices

IN THIS SECTION

- [Overview | 650](#)
- [Requirements | 652](#)

- [Topology | 653](#)
- [Configure Leaf 1 | 654](#)
- [Configure Leaf 3 | 658](#)
- [Verify the IPv6 Underlay on Leaf 3 | 662](#)

Overview

Ethernet VPNs (EVPNs) enable you to connect customer sites using Layer 2 virtual bridges. Virtual Extensible LANs (VXLANs) establish overlay tunnels that stretch the Layer 2 connection over an intervening Layer 3 network. Like VLANs, VXLANs help provide network segmentation, but without the scaling limitation of traditional VLANs. EVPN with VXLAN encapsulation enables Layer 2 connectivity at scale.

The physical underlay network in EVPN-VXLAN installations is often a two-layer IP fabric that includes spine and leaf devices. The spine devices—for example, switches in the QFX10000 line—provide connectivity between the leaf devices. The leaf devices—for example, QFX5120 switches—provide connectivity to attached hosts. In different overlay network configurations, the leaf or spine devices might function as either or both of the following:

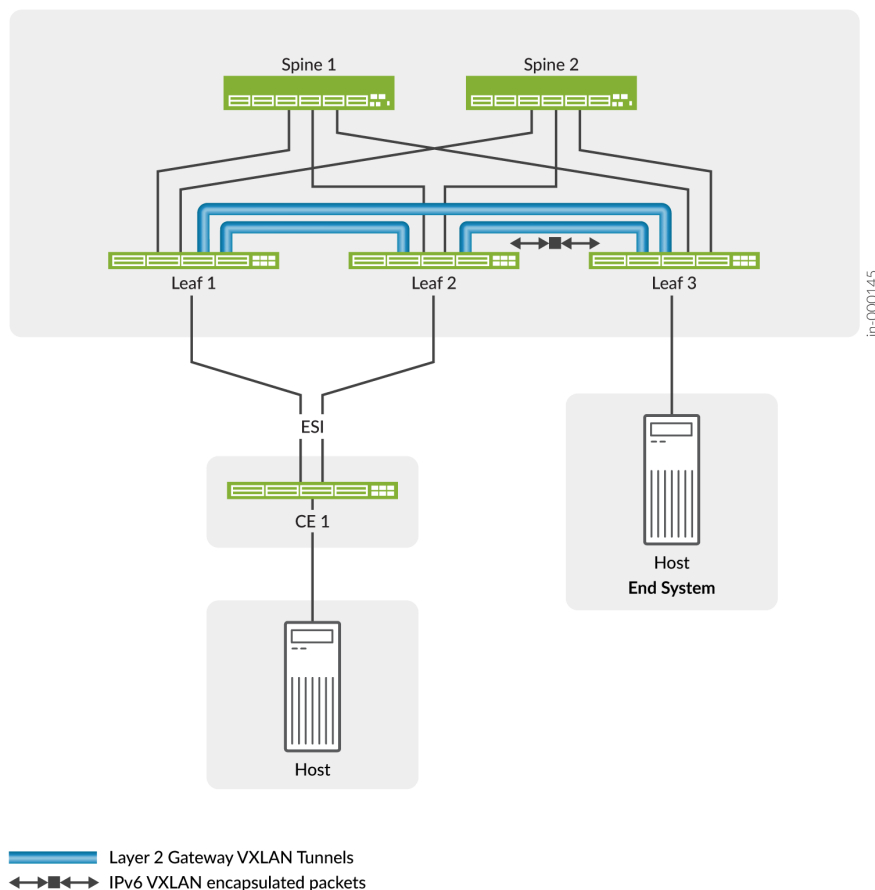
- Layer 2 gateways that handle traffic within a VXLAN.
- Layer 3 gateways that handle traffic between VXLANs using integrated routing and bridging (IRB) interfaces.

On supported platforms, in either case, the underlay network for the VXLAN overlay can use the IPv6 protocol to take advantage of the extended addressing and other capabilities of IPv6.

This example shows a use case to configure an IPv6 underlay for the Layer 2 VXLAN gateway leaf devices in a simple EVPN-VXLAN fabric. In this use case, the EVPN-VXLAN fabric supports a bridged overlay with VXLAN tunnels between the leaf devices. The leaf devices connect to end systems that

might be single homed or include EVPN multihoming for redundancy. The following figure shows a high-level view of the topology in this example:

Figure 59: EVPN-VXLAN Fabric with an IPv6 Underlay for Layer 2 VXLAN Gateway Devices



The following list describes the main differences in how you set up an IPv6 underlay compared to setting up an IPv4 underlay:

- You assign an IPv6 address in addition to an IPv4 address to the loopback interface on the devices that serve as the Layer 2 or Layer 3 VXLAN gateway VTEPs.
- QFX Series switches support an IPv6 VXLAN underlay only with MAC-VRF routing instances. (See ["MAC-VRF Routing Instance Type Overview" on page 459](#) for more information about using MAC-VRF routing instances.) As a result, you configure the EVPN instance as a MAC-VRF instance.
- You set the VTEP source interface as an IPv6 address. However, you configure the router ID with an IPv4 address, which the overlay requires for BGP handshaking to work.

- You can't mix IPv4 and IPv6 underlays in the same fabric, so you must configure an IPv6 underlay across all EVPN instances in the fabric.

Requirements

This example consists of a full mesh two-layer spine-and-leaf EVPN-VXLAN fabric with two spine devices and three leaf devices. You can configure the IPv6 underlay in this example using:

- QFX Series switches that support this feature.
- Junos OS Release 21.4R1 or later.

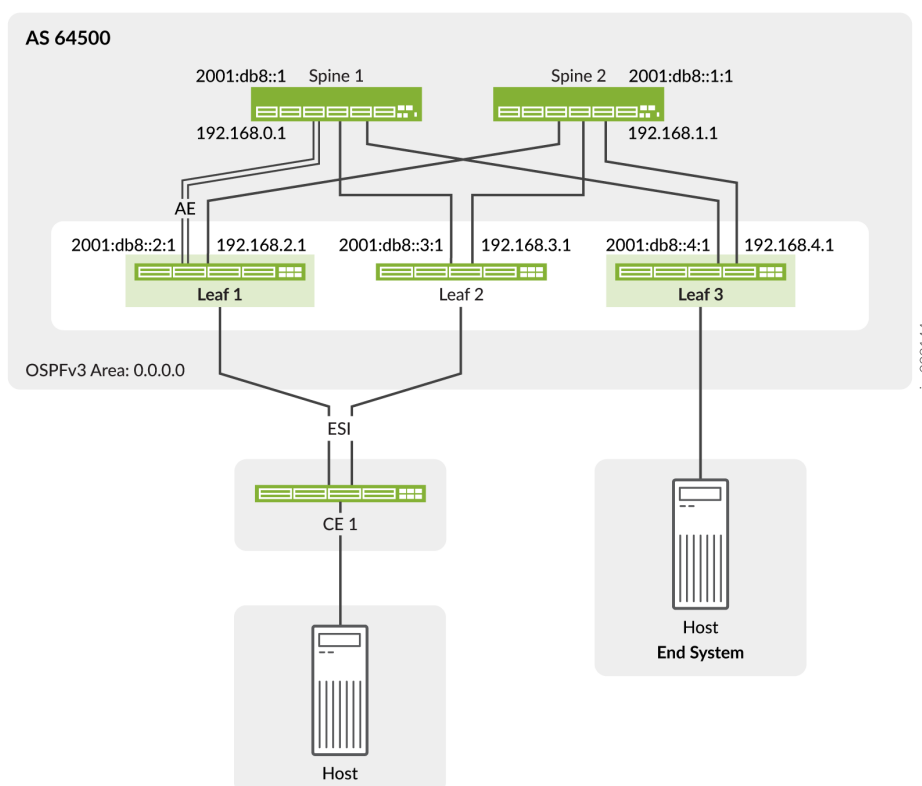
NOTE: We also support this feature in Junos OS Release 21.2R2.

The leaf devices can host multihomed or single homed end devices on the access side. This example illustrates configuring an Ethernet segment for EVPN multihoming on one leaf and a single homed end system interface on another leaf. However, the elements you configure for the IPv6 underlay are independent of the access-side configuration.

Topology

This example shows how to configure an IPv6 underlay on Leaf 1 and Leaf 3 for VXLAN overlay tunnels like those in [Figure 59 on page 651](#). The configuration uses OSPFv3 for IPv6 connectivity and iBGP with IPv6 neighbor addressing in a single autonomous system in the following topology:

Figure 60: Example Topology



Leaf 1 serves a customer edge switch that is multihomed to Leaf 1 and Leaf 2, so you would use a similar configuration on Leaf 2 to reach devices on that Ethernet segment.

In the example topology, Leaf 1 includes an aggregated Ethernet interface bundle for the connection to Spine 1. You configure the remaining spine and leaf connections on Leaf 1 and Leaf 3 as single interfaces. Leaf 3 includes an access-side interface configuration to a single-homed end system.

This example includes show commands you can run to verify IPv6 underlay operation. For simplicity, we show these verification commands and output only for Leaf 3. You see similar results from the same commands on the other leaf devices.

Configure Leaf 1

IN THIS SECTION

- [CLI Quick Configuration on Leaf 1 | 654](#)
- [Step-by-Step Procedure on Leaf 1 | 655](#)

CLI Quick Configuration on Leaf 1

To quickly configure Leaf 1 with an IPv6 underlay according to [Figure 60 on page 653](#), copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

```
set interfaces xe-0/0/18:0 description "LINK-0 to SPINE-1"
set interfaces xe-0/0/18:0 ether-options 802.3ad ae0
set interfaces xe-0/0/18:1 description "LINK-1 to SPINE-1"
set interfaces xe-0/0/18:1 ether-options 802.3ad ae0
set interfaces ae0 aggregated-ether-options link-speed mixed
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 unit 0 family inet address 10.0.2.2/30
set interfaces ae0 unit 0 family inet6 address 2001:db8::02:1:2/112
set interfaces xe-0/0/34:0 description "LINK to SPINE-2"
set interfaces xe-0/0/34:0 unit 0 family inet address 10.0.12.2/30
set interfaces xe-0/0/34:0 unit 0 family inet6 address 2001:db8::12:1:2/112
set interfaces xe-0/0/0:0 description "NETWORK LINK"
set interfaces xe-0/0/0:0 flexible-vlan-tagging
set interfaces xe-0/0/0:0 encapsulation extended-vlan-bridge
set interfaces xe-0/0/0:0 unit 100 vlan-id 100
set interfaces xe-0/0/0:0 unit 110 vlan-id 110
set interfaces lo0 unit 0 family inet address 192.168.2.1/32 primary
set interfaces lo0 unit 0 family inet6 address 2001:db8::2:1/128 primary
set forwarding-options evpn-vxlan shared-tunnels
set routing-instances USER_MVS1 instance-type mac-vrf
set routing-instances USER_MVS1 protocols evpn encapsulation vxlan
set routing-instances USER_MVS1 vtep-source-interface lo0.0 inet6
set routing-instances USER_MVS1 service-type vlan-aware
set routing-instances USER_MVS1 route-distinguisher 64500:11000002
set routing-instances USER_MVS1 vrf-target target:64500:1110
```

```

set routing-instances USER_MVS1 vrf-target auto
set routing-instances USER_MVS1 vlans V100 interface xe-0/0/0:0.100
set routing-instances USER_MVS1 vlans V100 vxlan vni 1100
set routing-instances USER_MVS1 vlans V110 interface xe-0/0/0:0.110
set routing-instances USER_MVS1 vlans V110 interface ae10.110
set routing-instances USER_MVS1 vlans V110 vxlan vni 1110
set protocols ospf3 area 0.0.0.0 interface xe-0/0/34:0.0
set protocols ospf3 area 0.0.0.0 interface ae0.0
set protocols ospf3 area 0.0.0.0 interface lo0.0 passive
set routing-options router-id 192.168.2.1
set routing-options autonomous-system 64500
set protocols bgp group vteps type internal
set protocols bgp group vteps local-address 2001:db8::2:1
set protocols bgp group vteps family evpn signaling
set protocols bgp group vteps neighbor 2001:db8::1
set protocols bgp group vteps neighbor 2001:db8::1:1
set protocols bgp group vteps neighbor 2001:db8::3:1
set protocols bgp group vteps neighbor 2001:db8::4:1
set interfaces xe-0/0/26:0 description "TO CE-1"
set interfaces xe-0/0/26:0 ether-options 802.3ad ae10
set interfaces ae10 flexible-vlan-tagging
set interfaces ae10 encapsulation extended-vlan-bridge
set interfaces ae10 esi 00:10:11:12:13:14:15:16:17:01
set interfaces ae10 esi all-active
set interfaces ae10 aggregated-ether-options link-speed mixed
set interfaces ae10 aggregated-ether-options lacp active
set interfaces ae10 aggregated-ether-options lacp periodic fast
set interfaces ae10 aggregated-ether-options lacp system-id 00:11:11:03:04:01
set interfaces ae10 unit 110 vlan-id 110

```

Step-by-Step Procedure on Leaf 1

1. Configure the interfaces for the EVPN fabric device connections. For illustrative purposes, in this example Leaf 1 connects to Spine 1 with an aggregated Ethernet (AE) interface bundle and to Spine 2 with a single interface.

```

set interfaces xe-0/0/18:0 description "LINK-0 to SPINE-1"
set interfaces xe-0/0/18:0 ether-options 802.3ad ae0
set interfaces xe-0/0/18:1 description "LINK-1 to SPINE-1"
set interfaces xe-0/0/18:1 ether-options 802.3ad ae0
set interfaces ae0 aggregated-ether-options link-speed mixed

```

```

set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 unit 0 family inet address 10.0.2.2/30
set interfaces ae0 unit 0 family inet6 address 2001:db8::02:1:2/112

set interfaces xe-0/0/34:0 description "LINK to SPINE-2"
set interfaces xe-0/0/34:0 unit 0 family inet address 10.0.12.2/30
set interfaces xe-0/0/34:0 unit 0 family inet6 address 2001:db8::12:1:2/112

```

2. Configure an interface for network traffic and the associated VLANs. This example uses a service provider style interface configuration.

```

set interfaces xe-0/0/0:0 description "NETWORK LINK"
set interfaces xe-0/0/0:0 flexible-vlan-tagging
set interfaces xe-0/0/0:0 encapsulation extended-vlan-bridge
set interfaces xe-0/0/0:0 unit 100 vlan-id 100
set interfaces xe-0/0/0:0 unit 110 vlan-id 110

```

3. Assign both an IPv4 address and an IPv6 address to the loopback interface on this device. You use both addresses in the configuration in a later step.

```

set interfaces lo0 unit 0 family inet address 192.168.2.1/32 primary
set interfaces lo0 unit 0 family inet6 address 2001:db8::2:1/128 primary

```

4. A device might have problems with VTEP scaling when the configuration uses multiple MAC-VRF instances. As a result, to avoid this problem, we require that you enable the shared tunnels feature on QFX5120 switches when setting up an IPv6 underlay . When you configure the shared-tunnels option, the device minimizes the number of next-hop entries to reach remote VTEPs. This statement is optional on the QFX10000 line of switches, which can handle higher VTEP scaling.

Include the following statement to globally enable shared VXLAN tunnels on the device:

```

set forwarding-options evpn-vxlan shared-tunnels

```

5. Create an EVPN-VXLAN MAC-VRF instance. To use an IPv6 underlay, you configure the device loopback interface as an IPv6 VTEP source interface. You configure the IPv6 underlay in a later step.

In this step you also configure the following elements in the MAC-VRF instance, which are the same whether you use an IPv4 underlay or an IPv6 underlay:

- Set the VLAN-aware Ethernet service type so you can associate multiple VLANs with the instance.

- Assign a route distinguisher for the instance.
- Assign the route target.

We also set the auto route target option here, which uses one target for both import and export and helps to simplify the configuration.

```
set routing-instances USER_MVS1 instance-type mac-vrf
set routing-instances USER_MVS1 protocols evpn encapsulation vxlan
set routing-instances USER_MVS1 vtep-source-interface lo0.0 inet6
set routing-instances USER_MVS1 service-type vlan-aware
set routing-instances USER_MVS1 route-distinguisher 64500:11000002
set routing-instances USER_MVS1 vrf-target target:64500:1110
set routing-instances USER_MVS1 vrf-target auto
```

6. Configure the VLANs associated with the MAC-VRF instance and VLAN to VNI mappings—in this example, VLAN 100 (VNI 1100) and VLAN 110 (VNI 1110). This step is the same for an IPv4 underlay or an IPv6 underlay. This step includes the access-side ESI interface in the instance as well (ae10, which you configure in the last step).

```
set routing-instances USER_MVS1 vlans V100 interface xe-0/0/0:0.100
set routing-instances USER_MVS1 vlans V100 vxlan vni 1100
set routing-instances USER_MVS1 vlans V110 interface xe-0/0/0:0.110
set routing-instances USER_MVS1 vlans V110 interface ae10.110
set routing-instances USER_MVS1 vlans V110 vxlan vni 1110
```

7. Set up the IPv6 underlay. This example uses OSPFv3 for the IPv6 underlay connectivity.

NOTE: You might alternatively use BGP (for example, external BGP [eBGP]) as the IPv6 underlay routing protocol.

```
set protocols ospf3 area 0.0.0.0 interface xe-0/0/34:0.0
set protocols ospf3 area 0.0.0.0 interface ae0.0
set protocols ospf3 area 0.0.0.0 interface lo0.0 passive
```

8. Set up the IPv6 overlay. This example uses using internal BGP (iBGP) as the overlay routing protocol for EVPN with VXLAN tunneling.

NOTE: Even though we use the IPv6 address family, for the BGP handshaking to work, you must configure the router ID as the loopback IPv4 address (192.168.2.1 in this case for Leaf 1). However, you use the IPv6 address for the VTEP local address.

```
set routing-options router-id 192.168.2.1
set routing-options autonomous-system 64500
set protocols bgp group vteps type internal
set protocols bgp group vteps local-address 2001:db8::2:1
set protocols bgp group vteps family evpn signaling
set protocols bgp group vteps neighbor 2001:db8::1
set protocols bgp group vteps neighbor 2001:db8::1:1
set protocols bgp group vteps neighbor 2001:db8::3:1
set protocols bgp group vteps neighbor 2001:db8::4:1
```

9. Set up an Ethernet segment (ESI) from Leaf 1 to CE 1, which is multihomed to Leaf 1 and Leaf 2. You would configure the ESI on Leaf 2 similarly. For simplicity, this example doesn't show the Leaf 2 configuration.

```
set interfaces xe-0/0/26:0 description "TO CE-1"
set interfaces xe-0/0/26:0 ether-options 802.3ad ae10
set interfaces ae10 flexible-vlan-tagging
set interfaces ae10 encapsulation extended-vlan-bridge
set interfaces ae10 esi 00:10:11:12:13:14:15:16:17:01
set interfaces ae10 esi all-active
set interfaces ae10 aggregated-ether-options link-speed mixed
set interfaces ae10 aggregated-ether-options lacp active
set interfaces ae10 aggregated-ether-options lacp periodic fast
set interfaces ae10 aggregated-ether-options lacp system-id 00:11:11:03:04:01
set interfaces ae10 unit 110 vlan-id 110
```

Configure Leaf 3

IN THIS SECTION

- [CLI Quick Configuration on Leaf 3 | 659](#)
- [Step-by-Step Procedure on Leaf 3 | 660](#)

CLI Quick Configuration on Leaf 3

To quickly configure Leaf 3 with an IPv6 underlay according to [Figure 60 on page 653](#), copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

```

set interfaces xe-0/0/18:0 description "LINK TO SPINE-1"
set interfaces xe-0/0/18:0 unit 0 family inet address 10.0.4.2/30
set interfaces xe-0/0/18:0 unit 0 family inet6 address 2001:db8::04:1:2/112
set interfaces xe-0/0/70:0 description "LINK TO SPINE-2"
set interfaces xe-0/0/70:0 unit 0 family inet address 10.0.14.2/30
set interfaces xe-0/0/70:0 unit 0 family inet6 address 2001:db8::14:1:2/112
set interfaces xe-0/0/35:3 description "NETWORK LINK"
set interfaces xe-0/0/35:3 flexible-vlan-tagging
set interfaces xe-0/0/35:3 native-vlan-id 110
set interfaces xe-0/0/35:3 encapsulation extended-vlan-bridge
set interfaces xe-0/0/35:3 unit 100 vlan-id 100
set interfaces xe-0/0/35:3 unit 110 vlan-id 110
set interfaces lo0 unit 0 family inet address 192.168.4.1/32 primary
set interfaces lo0 unit 0 family inet6 address 2001:db8::4:1/128 primary
set forwarding-options evpn-vxlan shared-tunnels
set routing-instances USER_MVS1 instance-type mac-vrf
set routing-instances USER_MVS1 protocols evpn encapsulation vxlan
set routing-instances USER_MVS1 vtep-source-interface lo0.0 inet6
set routing-instances USER_MVS1 service-type vlan-aware
set routing-instances USER_MVS1 route-distinguisher 64500:11000004
set routing-instances USER_MVS1 vrf-target target:64500:1110
set routing-instances USER_MVS1 vrf-target auto
set routing-instances USER_MVS1 vlans V100 interface xe-0/0/35:3.100
set routing-instances USER_MVS1 vlans V100 vxlan vni 1100
set routing-instances USER_MVS1 vlans V110 interface xe-0/0/35:3.110
set routing-instances USER_MVS1 vlans V110 vxlan vni 1110
set protocols ospf3 area 0.0.0.0 interface xe-0/0/18:0.0
set protocols ospf3 area 0.0.0.0 interface xe-0/0/70:0.0
set protocols ospf3 area 0.0.0.0 interface lo0.0 passive
set routing-options router-id 192.168.4.1
set routing-options autonomous-system 64500
set protocols bgp group vteps type internal
set protocols bgp group vteps local-address 2001:db8::4:1
set protocols bgp group vteps family evpn signaling
set protocols bgp group vteps neighbor 2001:db8::1

```

```
set protocols bgp group vteps neighbor 2001:db8::1:1
set protocols bgp group vteps neighbor 2001:db8::2:1
set protocols bgp group vteps neighbor 2001:db8::3:1
```

Step-by-Step Procedure on Leaf 3

1. Configure the interfaces for the EVPN fabric device connections from Leaf 3 to Spine 1 and Spine 2.

```
set interfaces xe-0/0/18:0 description "LINK TO SPINE-1"
set interfaces xe-0/0/18:0 unit 0 family inet address 10.0.4.2/30
set interfaces xe-0/0/18:0 unit 0 family inet6 address 2001:db8::04:1:2/112
set interfaces xe-0/0/70:0 description "LINK TO SPINE-2"
set interfaces xe-0/0/70:0 unit 0 family inet address 10.0.14.2/30
set interfaces xe-0/0/70:0 unit 0 family inet6 address 2001:db8::14:1:2/112
```

2. Configure an interface for network traffic and the associated VLANs. This example uses a service provider style interface configuration.

```
set interfaces xe-0/0/35:3 description "NETWORK LINK"
set interfaces xe-0/0/35:3 flexible-vlan-tagging
set interfaces xe-0/0/35:3 native-vlan-id 110
set interfaces xe-0/0/35:3 encapsulation extended-vlan-bridge
set interfaces xe-0/0/35:3 unit 100 vlan-id 100
set interfaces xe-0/0/35:3 unit 110 vlan-id 110
```

3. Assign both an IPv4 address and an IPv6 address to the loopback interface on this device. You use both addresses in the configuration in a later step.

```
set interfaces lo0 unit 0 family inet address 192.168.4.1/32 primary
set interfaces lo0 unit 0 family inet6 address 2001:db8::4:1/128 primary
```

4. A device might have problems with VTEP scaling when the configuration uses multiple MAC-VRF instances. As a result, to avoid this problem, we require that you enable the shared tunnels feature on QFX5120 switches when setting up an IPv6 underlay. When you configure the shared-tunnels option, the device minimizes the number of next-hop entries to reach remote VTEPs. This statement is optional on the QFX10000 line of switches, which can handle higher VTEP scaling.

Include the following statement to globally enable shared VXLAN tunnels on the device:

```
set forwarding-options evpn-vxlan shared-tunnels
```

5. Create an EVPN-VXLAN MAC-VRF instance. To use an IPv6 underlay, you configure the device loopback interface as an IPv6 VTEP source interface. You configure the IPv6 underlay in a later step.

In this step you also configure the following elements in the MAC-VRF instance, which are the same whether you use an IPv4 underlay or an IPv6 underlay:

- Set the VLAN-aware Ethernet service type so you can associate multiple VLANs with the instance.
- Assign a route distinguisher for the instance.
- Assign the route target.

We also set the auto route target option here, which uses one target for both import and export and helps to simplify the configuration.

```
set routing-instances USER_MVS1 instance-type mac-vrf
set routing-instances USER_MVS1 protocols evpn encapsulation vxlan
set routing-instances USER_MVS1 vtep-source-interface lo0.0 inet6
set routing-instances USER_MVS1 service-type vlan-aware
set routing-instances USER_MVS1 route-distinguisher 64500:11000004
set routing-instances USER_MVS1 vrf-target target:64500:1110
set routing-instances USER_MVS1 vrf-target auto
```

6. Configure the VLANs associated with the MAC-VRF instance and VLAN to VNI mappings—in this example, VLAN 100 (VNI 1100) and VLAN 110 (VNI 1110). This step is the same for an IPv4 underlay or an IPv6 underlay.

```
set routing-instances USER_MVS1 vlans V100 interface xe-0/0/35:3.100
set routing-instances USER_MVS1 vlans V100 vxlan vni 1100
set routing-instances USER_MVS1 vlans V110 interface xe-0/0/35:3.110
set routing-instances USER_MVS1 vlans V110 vxlan vni 1110
```

7. Set up the IPv6 underlay. This example uses OSPFv3 for the IPv6 underlay connectivity.

NOTE: You might alternatively use BGP (for example, external BGP [eBGP]) as the IPv6 underlay routing protocol.

```
set protocols ospf3 area 0.0.0.0 interface xe-0/0/18:0.0
set protocols ospf3 area 0.0.0.0 interface xe-0/0/70:0.0
set protocols ospf3 area 0.0.0.0 interface lo0.0 passive
```

8. Set up the IPv6 overlay. This example uses using internal BGP (iBGP) as the overlay routing protocol for EVPN with VXLAN tunneling.

NOTE: Even though we use the IPv6 address family, for the BGP handshaking to work, you must configure the router ID as the loopback IPv4 address (192.168.4.1 in this case for Leaf 3). However, you use the IPv6 address for the VTEP local address.

```
set routing-options router-id 192.168.4.1
set routing-options autonomous-system 64500
set protocols bgp group vteps type internal
set protocols bgp group vteps local-address 2001:db8::4:1
set protocols bgp group vteps family evpn signaling
set protocols bgp group vteps neighbor 2001:db8::1
set protocols bgp group vteps neighbor 2001:db8::1:1
set protocols bgp group vteps neighbor 2001:db8::2:1
set protocols bgp group vteps neighbor 2001:db8::3:1
```

Verify the IPv6 Underlay on Leaf 3

IN THIS SECTION

- [Verify Peer Device Connectivity | 663](#)
- [Verify VTEP Source Parameters | 664](#)
- [Verify Remote VTEPs | 665](#)
- [Verify MAC-VRF EVPN Instance Forwarding | 666](#)

Use the CLI commands in this section to verify the IPV6 underlay configuration is operational on the leaf devices in this example. This section shows the results from running these commands on Leaf 3.

This example includes `show mac-vrf forwarding command-name` commands that display information for MAC-VRF instance configurations. Most `show mac-vrf forwarding` commands are aliases for the same command in the following command hierarchies that you might use for the default switching instance or other instance types:

- QFX Series switches—`show ethernet-switching command-name`
- MX Series routers and EX9200 line of switches—`show l2-learning command-name` or `show bridge command-name`

See "[MAC-VRF Routing Instance Type Overview](#)" on page 459 for a full list of the MAC-VRF instance show commands and their mappings to the commands that display equivalent results for other instances.

On devices with multiple MAC-VRF EVPN instances, to avoid VTEP scaling issues, we might require or recommend that you enable the shared tunnels feature. On some platforms, shared tunnels are enabled by default. In this example, we enable shared tunnels on the leaf devices using the `set forwarding-options evpn-vxlan shared-tunnels` configuration statement. MAC-VRF show commands display shared tunnel VTEP interfaces as `vtep-index.shared-tunnel-unit`, where:

- *index* is the index associated with the MAC-VRF routing instance.
- *shared-tunnel-unit* is the unit number associated with the shared tunnel remote VTEP logical interface.

For example:

```
vtep-7.32772
```

Verify Peer Device Connectivity

Purpose

Check that the leaf device established BGP IPv6 connectivity to its peer spine and leaf devices in the fabric.

Action

Run the `show bgp summary` command on the leaf device:

```
user@leaf-3> show bgp summary

Threading mode: BGP I/O
Default eBGP mode: advertise - accept, receive - accept
Groups: 1 Peers: 5 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State   Pending
bgp.evpn.0
                33         33          0          0          0          0
Peer           AS      InPkt    OutPkt    OutQ    Flaps Last Up/Dwn State|#Active/Received/
Accepted/Damped...
2001:db8::1    64500    1428     1406      0        0    10:38:00 Establ
  bgp.evpn.0: 10/10/10/0
  USER_MVS1.evpn.0: 10/10/10/0
  __default_evpn__.evpn.0: 0/0/0/0
2001:db8::1:1  64500    1435     1406      0        0    10:38:01 Establ
  bgp.evpn.0: 10/10/10/0
  USER_MVS1.evpn.0: 10/10/10/0
  __default_evpn__.evpn.0: 0/0/0/0
2001:db8::2:1  64500    1578     1415      0        0    10:37:58 Establ
  bgp.evpn.0: 5/5/5/0
  USER_MVS1.evpn.0: 5/5/5/0
  __default_evpn__.evpn.0: 0/0/0/0
2001:db8::3:1  64500    1483     1406      0        0    10:37:55 Establ
  bgp.evpn.0: 3/3/3/0
  USER_MVS1.evpn.0: 3/3/3/0
  __default_evpn__.evpn.0: 0/0/0/0
```

Meaning

Leaf 3 (IPv6 address 2001:db8::3:1 in [Figure 60 on page 653](#)) sees its eBGP peer devices Spine 1 (2001:db8::1), Spine 2(2001:db8::1:1), Leaf 1(2001:db8::2:1) and Leaf 2 (2001:db8::3:1).

Verify VTEP Source Parameters

Purpose

View the configured IPv6 VTEP source interface(s).

Action

Run the `show mac-vrf forwarding vxlan-tunnel-end-point source` command:

```
user@leaf-3> show mac-vrf forwarding vxlan-tunnel-end-point source
Logical System Name      Id  SVTEP-IP      IFL  L3-Idx  SVTEP-Mode  ELP-SVTEP-IP
<default>                0   2001:db8::4:1  lo0.0 0
      L2-RTT              Bridge Domain      VNID   Translation-VNID  MC-Group-IP
      USER_MVS1          V110              1110      ::
```

Meaning

The output shows you configured Leaf 3 with IPv6 VTEP source address 2001:db8::4:1 on the loopback port in MAC-VRF instance USER-MVS1 for VLAN V110, which you mapped to VNI 1110.

Verify Remote VTEPs

Purpose

Verify the device has forwarding information for the remote VTEPs.

Action

Run the `show mac-vrf forwarding vxlan-tunnel-end-point remote` command:

```
user@leaf-3> show mac-vrf forwarding vxlan-tunnel-end-point remote
Logical System Name      Id  SVTEP-IP      IFL  L3-Idx  SVTEP-Mode  ELP-SVTEP-IP
<default>                0   2001:db8::4:1  lo0.0 0
RVTEP-IP      IFL-Idx  Interface  NH-Id  RVTEP-Mode  ELP-IP      Flags
2001:db8::1    612      vtep.32771 1758   RNVE
2001:db8::1:1  611      vtep.32770 1753   RNVE
2001:db8::2:1  613      vtep.32772 1759   RNVE
2001:db8::3:1  614      vtep.32773 1761   RNVE
RVTEP-IP      L2-RTT              IFL-Idx  Interface  NH-Id  RVTEP-Mode  ELP-IP
Flags
2001:db8::1    USER_MVS1          671145987 vtep-7.32771 1758   RNVE
      VNID          MC-Group-IP
      1110          ::
RVTEP-IP      L2-RTT              IFL-Idx  Interface  NH-Id  RVTEP-Mode  ELP-IP
```

```

Flags
2001:db8::1:1 USER_MVS1          671145986 vtep-7.32770 1753    RNVE
  VNID      MC-Group-IP
  1110      ::
RVTEP-IP    L2-RTT                IFL-Idx   Interface   NH-Id     RVTEP-Mode  ELP-IP
Flags
2001:db8::2:1 USER_MVS1          671145988 vtep-7.32772 1759    RNVE
  VNID      MC-Group-IP
  1110      ::
RVTEP-IP    L2-RTT                IFL-Idx   Interface   NH-Id     RVTEP-Mode  ELP-IP
Flags
2001:db8::3:1 USER_MVS1          671145989 vtep-7.32773 1761    RNVE
  VNID      MC-Group-IP
  1110      ::

```

Meaning

The output shows that Leaf 3 has forwarding information for remote IPv6 VTEPs on Leaf 1 (2001:db8::2:1) and Leaf 2 (2001:db8::3:1).

Verify MAC-VRF EVPN Instance Forwarding

Purpose

View the forwarding table for the configured MAC-VRF instance to see the interfaces for the remote VTEPs associated with the instance.

Action

Run the `show mac-vrf forwarding mac-table instance name` command for the MAC-VRF instance in this example, `USER_MVS1`:

```

user@leaf-3> show mac-vrf forwarding mac-table instance USER_MVS1

MAC flags (S - static MAC, D - dynamic MAC, L - locally learned, P - Persistent static
          SE - statistics enabled, NM - non configured MAC, R - remote PE MAC, O - ovsdb MAC)

Ethernet switching table : 3 entries, 3 learned
Routing instance : USER_MVS1

```

```

Ethernet switching table : 3 entries, 3 learned
Routing instance : USER_MVS1
  Vlan      MAC      MAC      Logical      SVLBNH/      Active
  name      address    flags    interface    VENH Index   source
  V110      00:01:01:10:00:fe  DRP      esi.1754
05:00:00:02:9a:00:00:04:56:00
  V110      54:4b:8c:d3:40:32  DRP      vtep-7.32771
2001:db8::1
  V110      84:03:28:50:3c:e0  DRP      vtep-7.32770
2001:db8::1:1

```

Meaning

The output for this command shows the MAC addresses that were populated in the MAC table.

Multicast Features with EVPN-VXLAN

IN THIS CHAPTER

- Multicast Support in EVPN-VXLAN Overlay Networks | 668
- Overview of Multicast Forwarding with IGMP Snooping or MLD Snooping in an EVPN-VXLAN Environment | 675
- Example: Preserving Bandwidth with IGMP Snooping in an EVPN-VXLAN Environment | 694
- Overview of Selective Multicast Forwarding | 747
- Configuring the number of SMET Nexthops | 750
- Assisted Replication Multicast Optimization in EVPN Networks | 752
- Optimized Inter-Subnet Multicast in EVPN Networks | 773

Multicast Support in EVPN-VXLAN Overlay Networks

IN THIS SECTION

- IPv4 Inter-VLAN Multicast Forwarding Modes for EVPN-VXLAN Overlay Networks | 668

This topic describes the following multicast feature, which is supported in an EVPN-VXLAN overlay network:

IPv4 Inter-VLAN Multicast Forwarding Modes for EVPN-VXLAN Overlay Networks

IN THIS SECTION

- Benefits of IPv4 Inter-VLAN Multicast Forwarding Modes | 669
- Supported EVPN-VXLAN Architectures and IPv4 Inter-VLAN Multicast Forwarding Modes | 669

- [Understanding Multicast Traffic Flows in a Centrally-Routed Bridging Overlay | 670](#)
- [Understanding Multicast Traffic Flows in an Edge-Routed Bridging Overlay | 672](#)
- [Differences Between IPv4 Inter-VLAN Multicast Forwarding Modes | 674](#)

Starting with Junos OS Release 17.3R3, the QFX10000 line of switches support two modes of IPv4 inter-VLAN multicast forwarding—centrally-routed mode and edge-routed mode—in an EVPN-VXLAN overlay network. The IP fabric architecture of your EVPN-VXLAN overlay network determines the mode that you must use.

Starting with Junos OS Release 20.4R1, QFX5110, QFX5120, and the QFX10000 line of switches also support centrally-routed forwarding mode for IPv6 intra-VLAN and inter-VLAN multicast traffic in an EVPN-VXLAN overlay network.

Starting with Junos OS Release 21.2R1, QFX5110, QFX5120, and QFX10002 switches also support the optimized inter-subnet multicast (OISM) routing and forwarding model for edge-routed bridging overlay EVPN-VXLAN fabrics. You can configure an edge-routed bridging overlay fabric to handle multicast traffic in local-only mode (edge-routed mode) or in OISM mode.

For more information about these EVPN-VXLAN architectures, the multicast mode required for each architecture, and how the multicast forwarding modes work, see the following sections:

Benefits of IPv4 Inter-VLAN Multicast Forwarding Modes

- The configuration statement enables you to choose the inter-VLAN multicast forwarding mode that suits the architecture of your EVPN-VXLAN overlay network.

Supported EVPN-VXLAN Architectures and IPv4 Inter-VLAN Multicast Forwarding Modes

We support the following modes of IPv4 inter-VLAN multicast forwarding:

- EVPN-VXLAN centrally routed bridging overlays (EVPN-VXLAN networks with a two-layer IP fabric) support central routing and forwarding of multicast traffic at the spine layer using a local-remote model. Centrally routed overlay networks have a layer of Layer 3 spine devices and another layer of Layer 2 leaf devices. You configure all spine devices with IRB interfaces that route the multicast packets from one VLAN to another. One spine device is elected to handle the inter-VLAN routing for the network.

In this mode, you configure the devices in the fabric with the `irb local-remote` option at the `[edit forwarding-options multicast-replication evpn]` hierarchy. (See ["multicast-replication" on page 1625](#).)

- EVPN-VXLAN edge-routed bridging overlays (EVPN-VXLAN networks with a collapsed IP fabric) support routing and forwarding of multicast traffic at the leaf layer using either a local-only model or the OISM model. Edge-routed overlay networks have leaf devices with IRB interfaces that handle multicast routing and forwarding at Layer 3 and Layer 2. (The leaf devices essentially act as spine-leaf devices.) All of the leaf devices handle routing multicast packets from one VLAN to another.

You configure leaf devices in the fabric with either the `irb local-only` option or `irb oism` option at the `[edit forwarding-options multicast-replication evpn]` hierarchy. (See ["multicast-replication" on page 1625](#).)

NOTE: This topic describes the local-only model.

See ["Optimized Inter-Subnet Multicast in EVPN Networks" on page 773](#) for complete details on leaf device configuration and operation with the OISM model.

For centrally-routed bridging overlays, you can simply retain the default setting, `irb local-remote`, at the `[edit forwarding-options multicast-replication evpn]` hierarchy level on all spine devices. For edge-routed bridging overlays, you must explicitly specify the `irb local-only` or `irb oism` option on all leaf devices.

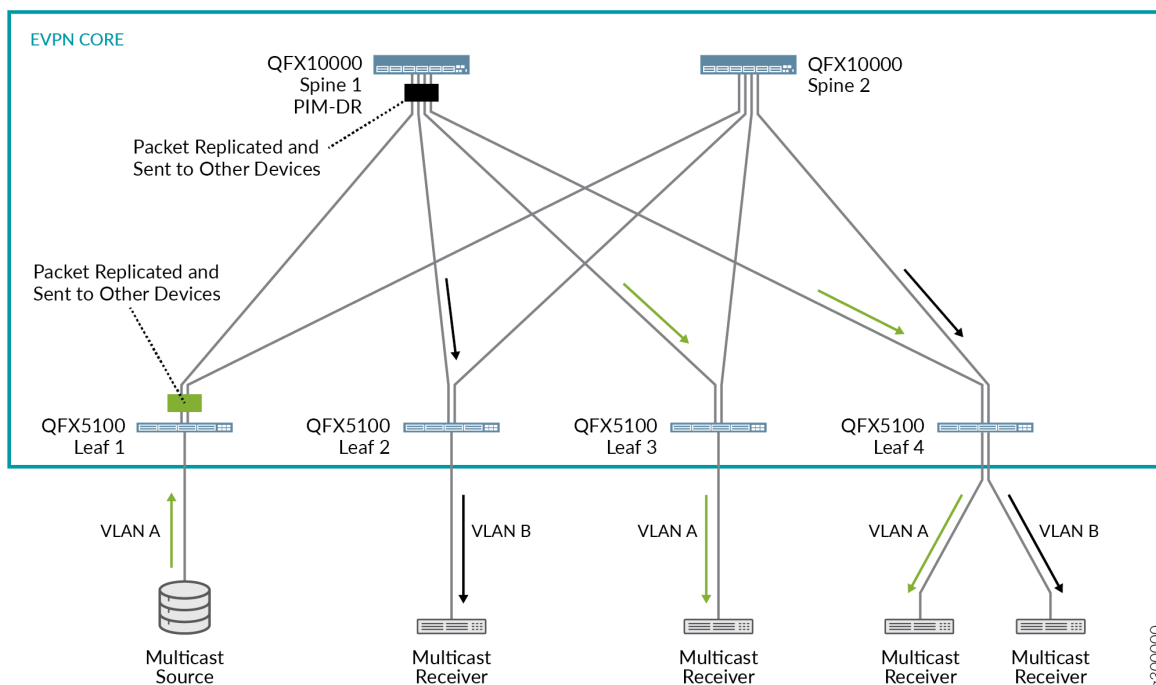
NOTE: We do not recommend specifying the `local-remote` option on some QFX10000 switches and the `local-only` option on the other QFX10000 switches in either of the overlay networks. Doing so might cause the QFX10000 switches to forward the inter-VLAN multicast traffic inconsistently.

Understanding Multicast Traffic Flows in a Centrally-Routed Bridging Overlay

This section describes the multicast traffic flows in a centrally-routed bridging overlay.

The network shown in [Figure 61 on page 671](#) includes the following devices.

Figure 61: Centrally-Routed Bridging Overlay



- Two QFX10000 switches that function as Layer 3 spine devices, on which the following key features are configured:
 - Centrally-routed (local-remote) multicast forwarding mode.
 - Protocol Independent Multicast (PIM). By virtue of PIM hello messages, Spine 1 is elected as the PIM designated router (PIM DR).
 - VLANs A and B.

NOTE: For inter-VLAN multicast forwarding to work properly in this scenario, you must configure all VLANs on each spine device.

- IRB interfaces associated with VLANs A and B.
- Four QFX5100 switches that function as Layer 2 leaf devices, on which VLANs A and B are configured are follows:
 - Leafs 1 and 3 are configured with VLAN A only.

- Leaf 2 is configured with VLAN B only.
- Leaf 4 is configured with VLANs A and B.
- A multicast source and various receivers.

When the multicast source shown in [Figure 61 on page 671](#) sends a packet from VLAN A, the following flows occur:

- **Flow 1: Intra-VLAN traffic**—Based on the ingress replication mechanism, Leaf 1 replicates and switches the packet to all spine and other leaf devices. Leafs 3 and 4, on which VLAN A is configured, receive and forward the packet to the connected multicast receivers.
- **Flow 2: Inter-VLAN traffic**—Upon receipt of the packet from Leaf 1 as described in flow 1, Spine 1, which is the PIM DR, takes the following action:
 - Routes the packet over the IRB interface associated with VLAN B.
 - Based on the ingress replication mechanism, replicates and forwards the packet to the other spine and the leaf devices.

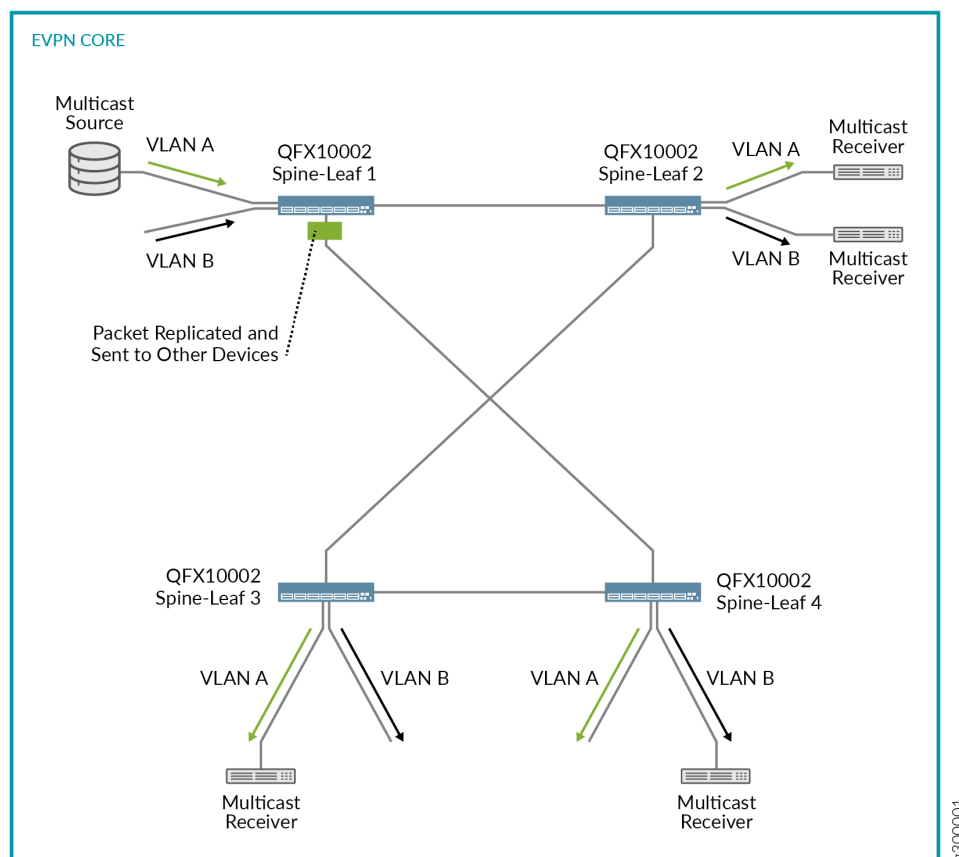
Leafs 2 and 4, on which VLAN B is configured, receive and forward the packet to the connected multicast receivers.

Understanding Multicast Traffic Flows in an Edge-Routed Bridging Overlay

This section describes the multicast traffic flow in an edge-routed bridging overlay.

The network shown in [Figure 62 on page 673](#) includes the following devices.

Figure 62: Edge-Routed Bridging Overlay



- Four QFX10002 switches that function as Layer 3 and Layer 2 spine-leaf devices, on which the following key features are configured:
 - Edge-routed (local-only) multicast forwarding mode.
 - PIM. To support the edge-routed mode of multicast forwarding, each spine-leaf device must act as the PIM DR for each VLAN. To enable a spine-leaf device to elect itself as the PIM DR, for each IRB interface, specify the `distributed-dr` configuration statement at the `[edit protocols pim interface interface-name]` hierarchy.
 - VLANs A and B.

NOTE: For inter-VLAN multicast forwarding to work properly in this scenario, you must configure all VLANs on each spine-leaf device.

- IRB interfaces associated with VLANs A and B.
- A multicast source and various receivers.

When the multicast source shown in [Figure 62 on page 673](#) sends a packet from VLAN A, the following flows occur:

- **Flow 1: Intra-VLAN traffic**—Based on the ingress replication mechanism, Spine-Leaf 1 replicates and switches the packet to the other spine-leaf devices. The spine-leaf devices forward the packet to VLAN A. Further, Spine-Leafs 2 and 3 forward the packet to VLAN A receivers.
- **Flow 2: Inter-VLAN traffic**—Upon receipt of the packet from Spine-Leaf 1 as described in flow 1, the spine-leaf devices route the packet over the IRB interface associated with VLAN B. Further, Spine-Leafs 2 and 4 forward the packet to the VLAN B receivers.

Differences Between IPv4 Inter-VLAN Multicast Forwarding Modes

There is an important difference between the centrally-routed and edge-routed modes.

With centrally-routed mode, for each VLAN, the spine device elected as the PIM-DR must replicate and send the packet to the other devices in the network. Keep in mind that the additional instances of replication consume bandwidth, and if many VLANs are configured, can potentially flood the EVPN core with multicast packets.

With edge-routed mode with the local-only model, the first spine-leaf device to receive a multicast packet replicates and sends the packet to the other spine-leaf devices. Upon receipt of the packet, the other spine-leaf devices route the packet to each VLAN and send the packet to access-side interfaces that handle traffic for the multicast receivers. In other words, the spine-leaf devices do not replicate the packet and send the packet out of the EVPN core-facing interfaces, which prevents excessive bandwidth consumption and congestion in the EVPN core.

The OISM model uses local routing, similar to the local-only model, to minimize multicast traffic flow in the EVPN core. However, with OISM, the leaf devices can also efficiently handle multicast traffic flow from sources outside the fabric to receivers within the fabric. OISM similarly enables multicast sources inside the fabric to send traffic to multicast receivers outside the fabric. See "[Optimized Inter-Subnet Multicast in EVPN Networks](#)" on page 773 for details on OISM configuration and operation.

Release History Table

Release	Description
21.2R1	Starting with Junos OS Release 21.2R1, QFX5110, QFX5120, and QFX10002 switches also support the optimized inter-subnet multicast (OISM) forwarding model for edge-routed bridging overlay EVPN-VXLAN fabrics.

20.4R1	Starting with Junos OS Release 20.4R1, QFX5110, QFX5120, and the QFX10000 line of switches also support centrally-routed forwarding mode for IPv6 intra-VLAN and inter-VLAN multicast traffic in an EVPN-VXLAN overlay network.
17.3R3	Starting with Junos OS Release 17.3R3, the QFX10000 line of switches support two modes of IPv4 inter-VLAN multicast forwarding—centrally-routed mode and edge-routed mode—in an EVPN-VXLAN overlay network. The IP fabric architecture of your EVPN-VXLAN overlay network determines the mode that you must use.

RELATED DOCUMENTATION

[Overview of Multicast Forwarding with IGMP Snooping or MLD Snooping in an EVPN-VXLAN Environment | 675](#)

[Optimized Inter-Subnet Multicast in EVPN Networks | 773](#)

Overview of Multicast Forwarding with IGMP Snooping or MLD Snooping in an EVPN-VXLAN Environment

IN THIS SECTION

- [Benefits of Multicast Forwarding with IGMP Snooping or MLD Snooping in an EVPN-VXLAN Environment | 677](#)
- [Supported IGMP or MLD Versions and Group Membership Report Modes | 678](#)
- [Summary of Multicast Traffic Forwarding and Routing Use Cases | 679](#)
- [Use Case 1: Intra-VLAN Multicast Traffic Forwarding | 680](#)
- [Use Case 2: Inter-VLAN Multicast Routing and Forwarding—IRB Interfaces with PIM | 683](#)
- [Use Case 3: Inter-VLAN Multicast Routing and Forwarding—PIM Gateway with Layer 2 Connectivity | 687](#)
- [Use Case 4: Inter-VLAN Multicast Routing and Forwarding—PIM Gateway with Layer 3 Connectivity | 689](#)
- [Use Case 5: Inter-VLAN Multicast Routing and Forwarding—External Multicast Router | 692](#)
- [EVPN Multicast Flags Extended Community | 692](#)

Internet Group Management Protocol (IGMP) snooping and Multicast Listener Discovery (MLD) snooping constrain multicast traffic in a broadcast domain to interested receivers and multicast devices.

In an environment with a significant volume of multicast traffic, using IGMP or MLD snooping preserves bandwidth because multicast traffic is forwarded only on those interfaces where there are multicast listeners. IGMP snooping optimizes IPv4 multicast traffic flow. MLD snooping optimizes IPv6 multicast traffic flow.

NOTE: Starting with Junos OS Release 21.2R1, we support optimized inter-subnet multicast (OISM) routing and forwarding with IGMP snooping in EVPN-VXLAN edge-routed bridging overlay networks. See ["Optimized Inter-Subnet Multicast in EVPN Networks" on page 773](#) for full details on OISM configuration and operation. You configure IGMP snooping on the fabric leaf devices as part of OISM configuration.

Starting with Junos OS Release 17.2R1, QFX10000 switches support IGMP snooping in an Ethernet VPN (EVPN)-Virtual Extensible LAN (VXLAN) edge-routed bridging overlay (EVPN-VXLAN topology with a collapsed IP fabric).

Starting with Junos OS Release 17.3R1, QFX10000 switches support the exchange of traffic between multicast sources and receivers in an EVPN-VXLAN edge-routed bridging overlay, which uses IGMP, and sources and receivers in an external Protocol Independent Multicast (PIM) domain. A Layer 2 multicast VLAN (MVLAN) and associated IRB interfaces enable the exchange of multicast traffic between these two domains.

IGMP snooping support in an EVPN-VXLAN network is available on the following switches in the QFX5000 line. In releases up until Junos OS Releases 18.4R2 and 19.1R2, with IGMP snooping enabled, these switches only constrain flooding for multicast traffic coming in on the VXLAN tunnel network ports; they still flood multicast traffic coming in from an access interface to all other access and network interfaces:

- Starting with Junos OS Release 18.1R1, QFX5110 switches support IGMP snooping in an EVPN-VXLAN centrally-routed bridging overlay (EVPN-VXLAN topology with a two-layer IP fabric) for forwarding multicast traffic within VLANs. You can't configure IRB interfaces on a VXLAN with IGMP snooping for forwarding multicast traffic between VLANs. (You can only configure and use IRB interfaces for unicast traffic.)
- Starting with Junos OS Release 18.4R2 (but not Junos OS Releases 19.1R1 and 19.2R1), QFX5120-48Y switches support IGMP snooping in an EVPN-VXLAN centrally-routed bridging overlay.
- Starting with Junos OS Release 19.1R1, QFX5120-32C switches support IGMP snooping in EVPN-VXLAN centrally-routed and edge-routed bridging overlays.
- Starting in Junos OS Releases 18.4R2 and 19.1R2, selective multicast forwarding is enabled by default on QFX5110 and QFX5120 switches when you configure IGMP snooping in EVPN-VXLAN networks, further constraining multicast traffic flooding. With IGMP snooping and selective multicast forwarding, these switches send the multicast traffic only to interested receivers in both the EVPN

core and on the access side for multicast traffic coming in either from an access interface or an EVPN network interface.

Starting with Junos OS Release 19.3R1, EX9200 switches, MX Series routers, and vMX virtual routers support IGMP version 2 (IGMPv2) and IGMP version 3 (IGMPv3), IGMP snooping, selective multicast forwarding, external PIM gateways, and external multicast routers with an EVPN-VXLAN centrally-routed bridging overlay.

Starting in Junos OS Releases 20.4R1, in EVPN-VXLAN centrally-routed bridging overlay fabrics, QFX5110, QFX5120 and the QFX10000 line of switches support IGMPv3 with IGMP snooping for IPv4 multicast traffic, and MLD version 1 (MLDv1) and MLD version 2 (MLDv2) with MLD snooping for IPv6 multicast traffic. You can configure these switches to process IGMPv3 and MLDv2 source-specific multicast (SSM) reports, but these devices can't process both SSM reports and any-source multicast (ASM) reports at the same time. When you configure them to operate in SSM mode, these devices drop any ASM reports. When not configured to operate in SSM mode (the default setting), these devices process any ASM reports but drop IGMPv3 and MLDv2 SSM reports.

NOTE: Unless called out explicitly, the information in this topic applies to IGMPv2, IGMPv3, MLDv1, and MLDv2 on the devices that support these protocols in the following IP fabric architectures:

- EVPN-VXLAN edge-routed bridging overlay
- EVPN-VXLAN centrally-routed bridging overlay

NOTE: On a Juniper Networks switching device, for example, a QFX10000 switch, you can configure a *VLAN*. On a Juniper Networks routing device, for example, an MX480 router, you can configure the same entity, which is called a *bridge domain*. To keep things simple, this topic uses the term *VLAN* when referring to the same entity configured on both Juniper Networks switching and routing devices.

Benefits of Multicast Forwarding with IGMP Snooping or MLD Snooping in an EVPN-VXLAN Environment

- In an environment with a significant volume of multicast traffic, using IGMP snooping or MLD snooping constrains multicast traffic in a VLAN to interested receivers and multicast devices, which conserves network bandwidth.
- Synchronizing the IGMP or MLD state among all EVPN devices for multihomed receivers ensures that all subscribed listeners receive multicast traffic, even in cases such as the following:

- IGMP or MLD membership reports for a multicast group might arrive on an EVPN device that is not the Ethernet segment's designated forwarder (DF).
- An IGMP or MLD message to leave a multicast group arrives at a different EVPN device than the EVPN device where the corresponding join message for the group was received.
- Selective multicast forwarding conserves bandwidth usage in the EVPN core and reduces the load on egress EVPN devices that do not have listeners.
- The support of external PIM gateways enables the exchange of multicast traffic between sources and listeners in an EVPN-VXLAN network and sources and listeners in an external PIM domain. Without this support, the sources and listeners in these two domains would not be able to communicate.

Supported IGMP or MLD Versions and Group Membership Report Modes

Table 30 on page 678 outlines the supported IGMP versions and the membership report modes supported for each version.

Table 30: Supported IGMP Versions and Group Membership Report Modes

IGMP Versions	Any-Source Multicast (ASM) (*,G) Only	Source-Specific Multicast (SSM) (S,G) Only	ASM (*,G) + SSM (S,G)
IGMPv2	Yes (default)	No	No
IGMPv3	Yes (default)	Yes (if configured)	No
MLDv1	Yes (default)	No	No
MLDv2	Yes (default)	Yes (if configured)	No

To explicitly configure EVPN devices to process only SSM (S,G) membership reports for IGMPv3 or MLDv2, set the `evpn-ssm-reports-only` configuration option at the `[edit protocols igmp-snooping vlan vlan-name]` hierarchy level.

You can enable SSM-only processing for one or more VLANs in an EVPN routing instance (EVI). When enabling this option for a routing instance of type `virtual switch`, the behavior applies to all VLANs in the virtual switch instance. When you enable this option, the device doesn't process ASM reports and drops them.

If you don't configure the `evpn-ssm-reports-only` option, by default, EVPN devices process IGMPv2, IGMPv3, MLDv1, or MLDv2 ASM reports and drop IGMPv3 or MLDv2 SSM reports.

Summary of Multicast Traffic Forwarding and Routing Use Cases

Table 31 on page 679 provides a summary of the multicast traffic forwarding and routing use cases that we support in EVPN-VXLAN networks and our recommendation for when you should apply a use case to your EVPN-VXLAN network.

Table 31: Supported Multicast Traffic Forwarding and Routing Use Cases and Recommended Usage

Use Case Number	Use Case Name	Summary	Recommended Usage
1	Intra-VLAN multicast traffic forwarding	Forwarding of multicast traffic to hosts within the same VLAN.	We recommend implementing this basic use case in all EVPN-VXLAN networks.
2	Inter-VLAN multicast routing and forwarding —IRB interfaces with PIM	IRB interfaces using PIM on Layer 3 EVPN devices. These interfaces route multicast traffic between source and receiver VLANs.	We recommend implementing this basic use case in all EVPN-VXLAN networks except when you prefer to use an external multicast router to handle inter-VLAN routing (see use case 5).
3	Inter-VLAN multicast routing and forwarding —PIM gateway with Layer 2 connectivity	A Layer 2 mechanism for a data center, which uses IGMP and PIM, to exchange multicast traffic with an external PIM domain.	We recommend this use case in either EVPN-VXLAN edge-routed bridging overlays or EVPN-VXLAN centrally-routed bridging overlays.
4	Inter-VLAN multicast routing and forwarding —PIM gateway with Layer 3 connectivity	A Layer 3 mechanism for a data center, which uses IGMP (or MLD) and PIM, to exchange multicast traffic with an external PIM domain.	We recommend this use case in EVPN-VXLAN centrally-routed bridging overlays only.
5	Inter-VLAN multicast routing and forwarding —external multicast router	Instead of IRB interfaces on Layer 3 EVPN devices, an external multicast router handles inter-VLAN routing.	We recommend this use case when you prefer to use an external multicast router instead of IRB interfaces on Layer 3 EVPN devices to handle inter-VLAN routing.

For example, in a typical EVPN-VXLAN edge-routed bridging overlay, you can implement use case 1 for intra-VLAN forwarding and use case 2 for inter-VLAN routing and forwarding. Or, if you want an

external multicast router to handle inter-VLAN routing in your EVPN-VXLAN network instead of EVPN devices with IRB interfaces running PIM, you can implement use case 5 instead of use case 2. If there are hosts in an existing external PIM domain that you want hosts in your EVPN-VXLAN network to communicate with, you can also implement use case 3.

When implementing any of the use cases in an EVPN-VXLAN centrally-routed bridging overlay, you can use a mix of spine devices—for example, MX Series routers, EX9200 switches, and QFX10000 switches. However, if you do this, keep in mind that the functionality of all spine devices is determined by the limitations of each spine device. For example, QFX10000 switches support a single routing instance of type `virtual-switch`. Although MX Series routers and EX9200 switches support multiple routing instances of type `evpn` or `virtual-switch`, on each of these devices, you would have to configure a single routing instance of type `virtual-switch` to interoperate with the QFX10000 switches.

Use Case 1: Intra-VLAN Multicast Traffic Forwarding

We recommend this basic use case for all EVPN-VXLAN networks.

This use case supports the forwarding of multicast traffic to hosts within the same VLAN and includes the following key features:

- Hosts that are single-homed to an EVPN device or multihomed to more than one EVPN device in all-active mode.

NOTE: EVPN-VXLAN multicast uses special IGMP and MLD group leave processing to handle multihomed sources and receivers, so we don't support the `immediate-leave` configuration option in the `[edit protocols igmp-snooping]` or `[edit protocols mld-snooping]` hierarchies in EVPN-VXLAN networks.

- Routing instances:
 - (QFX Series switches) A single routing instance of type `virtual-switch`.
 - (MX Series routers, vMX virtual routers, and EX9200 switches) Multiple routing instances of type `evpn` or `virtual-switch`.
 - EVI route target extended community attributes associated with multihomed EVIs. BGP EVPN Type 7 (Join Sync Route) and Type 8 (Leave Synch Route) routes carry these attributes to enable the simultaneous support of multiple EVPN routing instances.

For information about another supported extended community, see the “EVPN Multicast Flags Extended Community” section.

- IGMPv2, IGMPv3, MLDv1 or MLDv2. For information about the membership report modes supported for each IGMP or MLD version, see [Table 30 on page 678](#). For information about IGMP or

MLD route synchronization between multihomed EVPN devices, see ["Overview of Multicast Forwarding with IGMP or MLD Snooping in an EVPN-MPLS Environment"](#) on page 1013.

- IGMP snooping or MLD snooping. Hosts in a network send IGMP reports (for IPv4 traffic) or MLD reports (for IPv6 traffic) expressing interest in particular multicast groups from multicast sources. EVPN devices with IGMP snooping or MLD snooping enabled listen to the IGMP or MLD reports, and use the snooped information on the access side to establish multicast routes that only forward traffic for a multicast group to interested receivers.

IGMP snooping or MLD snooping supports multicast senders and receivers in the same or different sites. A site can have either receivers only, sources only, or both senders and receivers attached to it.

- Selective multicast forwarding (advertising EVPN Type 6 Selective Multicast Ethernet Tag (SMET) routes for forwarding only to interested receivers). This feature enables EVPN devices to selectively forward multicast traffic to only the devices in the EVPN core that have expressed interest in that multicast group.

NOTE: We support selective multicast forwarding to devices in the EVPN core only in EVPN-VXLAN centrally-routed bridging overlays.

When you enable IGMP snooping or MLD snooping, selective multicast forwarding is enabled by default.

- EVPN devices that do not support IGMP snooping, MLD snooping, and selective multicast forwarding.

Although you can implement this use case in an EVPN single-homed environment, this use case is particularly effective in an EVPN multihomed environment with a high volume of multicast traffic.

All multihomed interfaces must have the same configuration, and all multihomed peer EVPN devices must be in active mode (not standby or passive mode).

An EVPN device that initially receives traffic from a multicast source is known as the *ingress device*. The ingress device handles the forwarding of intra-VLAN multicast traffic as follows:

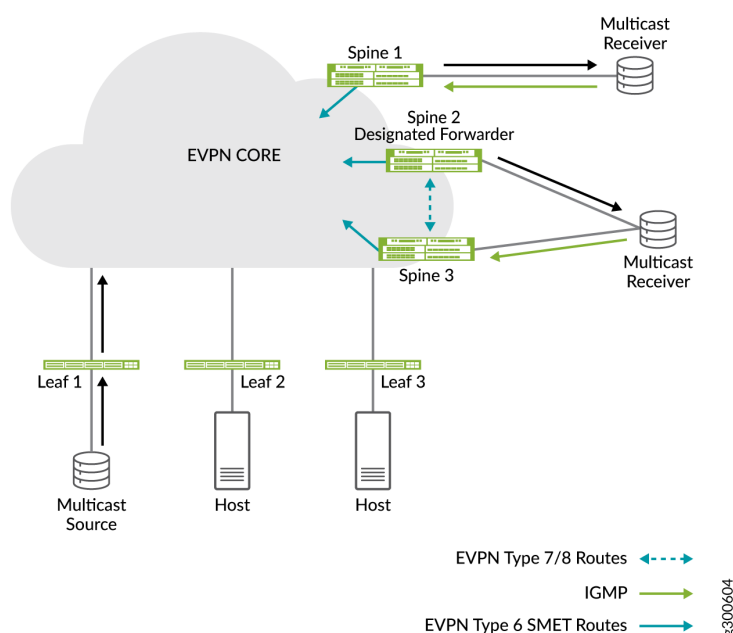
- With IGMP snooping or MLD snooping enabled (which also enable selective multicast forwarding on supporting devices):
 - As shown in [Figure 63 on page 682](#), the ingress device (leaf 1) selectively forwards the traffic to other EVPN devices with access interfaces where there are interested receivers for the same multicast group.
 - The traffic is then selectively forwarded to egress devices in the EVPN core that have advertised the EVPN Type 6 SMET routes.

- If any EVPN devices do not support IGMP snooping or MLD snooping, or the ability to originate EVPN Type 6 SMET routes, the ingress device floods multicast traffic to these devices.
- If a host is multihomed to more than one EVPN device, the EVPN devices exchange EVPN Type 7 and Type 8 routes as shown in [Figure 63 on page 682](#). This exchange synchronizes IGMP or MLD membership reports received on multihomed interfaces to coordinate status from messages that go to different EVPN devices or in case one of the EVPN devices fails.

NOTE: The EVPN Type 7 and Type 8 routes carry EVI route extended community attributes to ensure the right EVPN instance gets the IGMP state information on devices with multiple routing instances. QFX Series switches support IGMP snooping only in the default EVPN routing instance (default-switch). In Junos OS releases before 17.4R2, 17.3R3, or 18.1R1, these switches did not include EVI route extended community attributes in Type 7 and Type 8 routes, so they don't properly synchronize the IGMP state if you also have other routing instances configured. Starting in Junos OS releases 17.4R2, 17.3R3, and 18.1R1, QFX10000 switches include the EVI route extended community attributes that identify the target routing instance, and can synchronize IGMP state if IGMP snooping is enabled in the default EVPN routing instance when other routing instances are configured.

In releases that support MLD and MLD snooping in EVPN-VXLAN fabrics with multihoming, the same behavior applies to synchronizing the MLD state.

Figure 63: Intra-VLAN Multicast Traffic Flow with IGMP Snooping and Selective Multicast Forwarding



If you have configured IRB interfaces with PIM on one or more of the Layer 3 devices in your EVPN-VXLAN network (use case 2), note that the ingress device forwards the multicast traffic to the Layer 3 devices. The ingress device takes this action to register itself with the Layer 3 device that acts as the PIM rendezvous point (RP).

Use Case 2: Inter-VLAN Multicast Routing and Forwarding—IRB Interfaces with PIM

We recommend this basic use case for all EVPN-VXLAN networks except when you prefer to use an external multicast router to handle inter-VLAN routing (see Use Case 5: Inter-VLAN Multicast Routing and Forwarding—External Multicast Router).

For this use case, IRB interfaces using Protocol Independent Multicast (PIM) route multicast traffic between source and receiver VLANs. The EVPN devices on which the IRB interfaces reside then forward the routed traffic using these key features:

- Inclusive multicast forwarding with ingress replication
- IGMP snooping or MLD snooping (if supported)
- Selective multicast forwarding

The default behavior of inclusive multicast forwarding is to replicate multicast traffic and flood the traffic to all devices. For this use case, however, we support inclusive multicast forwarding coupled with IGMP snooping (or MLD snooping) and selective multicast forwarding. As a result, the multicast traffic is replicated but selectively forwarded to access interfaces and devices in the EVPN core that have interested receivers.

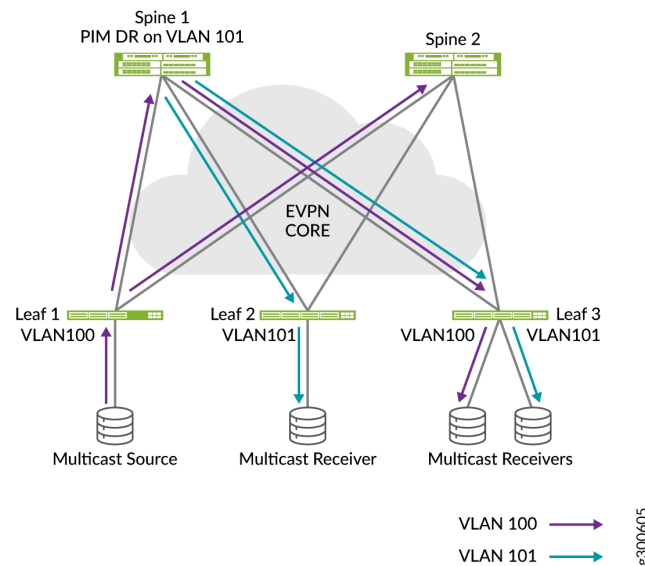
For information about the EVPN multicast flags extended community, which Juniper Networks devices that support EVPN and IGMP snooping (or MLD snooping) include in EVPN Type 3 (Inclusive Multicast Ethernet Tag) routes, see the “EVPN Multicast Flags Extended Community” section.

In an EVPN-VXLAN centrally-routed bridging overlay, you can configure the spine devices so that some of them perform inter-VLAN routing and forwarding of multicast traffic and some do not. At a minimum, we recommend that you configure two spine devices to perform inter-VLAN routing and forwarding.

When there are multiple devices that can perform the inter-VLAN routing and forwarding of multicast traffic, one device is elected as the designated router (DR) for each VLAN.

In the sample EVPN-VXLAN centrally-routed bridging overlay shown in [Figure 64 on page 684](#), assume that multicast traffic needs to be routed from source VLAN 100 to receiver VLAN 101. Receiver VLAN 101 is configured on spine 1, which is designated as the DR for that VLAN.

Figure 64: Inter-VLAN Multicast Traffic Flow with IRB Interface and PIM



After the inter-VLAN routing occurs, the EVPN device forwards the routed traffic to:

- Access interfaces that have multicast listeners (IGMP snooping or MLD snooping).
- Egress devices in the EVPN core that have sent EVPN Type 6 SMET routes for the multicast group members in receiver VLAN 2 (selective multicast forwarding).

To understand how IGMP snooping (or MLD snooping) and selective multicast forwarding reduce the impact of the replicating and flooding behavior of inclusive multicast forwarding, assume that an EVPN-VXLAN centrally-routed bridging overlay includes the following elements:

- 100 IRB interfaces using PIM starting with irb.1 and going up to irb.100
- 100 VLANs
- 20 EVPN devices

For the sample EVPN-VXLAN centrally-routed bridging overlay, m represents the number of VLANs, and n represents the number of EVPN devices. Assuming that IGMP snooping (or MLD snooping) and selective multicast forwarding are disabled, when multicast traffic arrives on irb.1, the EVPN device replicates the traffic $m * n$ times or $100 * 20$ times, which equals a rate of 20,000 packets. If the

incoming traffic rate for a particular multicast group is 100 packets per second (pps), the EVPN device would have to replicate 200,000 pps for that multicast group.

If IGMP snooping (or MLD snooping) and selective multicast forwarding are enabled in the sample EVPN-VXLAN centrally-routed bridging overlay, assume that there are interested receivers for a particular multicast group on only 4 VLANs and 3 EVPN devices. In this case, the EVPN device replicates the traffic at a rate of $100 * m * n$ times ($100 * 4 * 3$), which equals 1200 pps. Note the significant reduction in the replication rate and the amount of traffic that must be forwarded.

When implementing this use case, keep in mind that there are important differences for EVPN-VXLAN centrally-routed bridging overlays and EVPN-VXLAN edge-routed bridging overlays. [Table 32 on page 685](#) outlines these differences

Table 32: Use Case 2: Important Differences for EVPN-VXLAN Edge-routed and Centrally-routed Bridging Overlays

EVPN VXLAN IP Fabric Architectures	Support Mix of Juniper Networks Devices?	All EVPN Devices Required to Host All VLANs In EVPN-VXLAN Network?	All EVPN Devices Required to Host All VLANs that Include Multicast Listeners?	Required PIM Configuration
EVPN-VXLAN edge-routed bridging overlay	No. We support only QFX10000 switches for all EVPN devices.	Yes	Yes	Configure PIM distributed designated router (DDR) functionality on the IRB interfaces of the EVPN devices.

Table 32: Use Case 2: Important Differences for EVPN-VXLAN Edge-routed and Centrally-routed Bridging Overlays (*Continued*)

EVPN VXLAN IP Fabric Architectures	Support Mix of Juniper Networks Devices?	All EVPN Devices Required to Host All VLANs In EVPN-VXLAN Network?	All EVPN Devices Required to Host All VLANs that Include Multicast Listeners?	Required PIM Configuration
EVPN-VXLAN centrally-routed bridging overlay	<p>Yes.</p> <p>Spine devices: We support mix of MX Series routers, EX9200 switches, and QFX10000 switches.</p> <p>Leaf devices: We support mix of MX Series routers and QFX5110 switches.</p> <p>NOTE: If you deploy a mix of spine devices, keep in mind that the functionality of all spine devices is determined by the limitations of each spine device. For example, QFX10000 switches support a single routing instance of type virtual-switch. Although MX Series routers and EX9200 switches support multiple routing instances of type evpn or virtual-switch, on each of these devices, you would have to configure a single routing instance of type virtual-switch to interoperate with the QFX10000 switches.</p>	No	No. However, you must configure all VLANs that include multicast listeners on each spine device that performs inter-VLAN routing. You don't need to configure all VLANs that include multicast listeners on each leaf device.	Do not configure DDR functionality on the IRB interfaces of the spine devices. By not enabling DDR on an IRB interface, PIM remains in a default mode on the interface, which means that the interface acts the designated router for the VLANs.

In addition to the differences described in [Table 32 on page 685](#), a hair pinning issue exists with an EVPN-VXLAN centrally-routed bridging overlay. Multicast traffic typically flows from a source host to a leaf device to a spine device, which handles the inter-VLAN routing. The spine device then replicates and forwards the traffic to VLANs and EVPN devices with multicast listeners. When forwarding the

traffic in this type of EVPN-VXLAN overlay, be aware that the spine device returns the traffic to the leaf device from which the traffic originated (hair-pinning). This issue is inherent with the design of the EVPN-VXLAN centrally-routed bridging overlay. When designing your EVPN-VXLAN overlay, keep this issue in mind especially if you expect the volume of multicast traffic in your overlay to be high and the replication rate of traffic ($m * n$ times) to be large.

Use Case 3: Inter-VLAN Multicast Routing and Forwarding—PIM Gateway with Layer 2 Connectivity

We recommend the PIM gateway with Layer 2 connectivity use case for both EVPN-VXLAN edge-routed bridging overlays and EVPN-VXLAN centrally-routed bridging overlays.

For this use case, we assume the following:

- You have deployed a EVPN-VXLAN network to support a data center.
- In this network, you have already set up:
 - Intra-VLAN multicast traffic forwarding as described in use case 1.
 - Inter-VLAN multicast traffic routing and forwarding as described in use case 2.
- There are multicast sources and receivers within the data center that you want to communicate with multicast sources and receivers in an external PIM domain.

NOTE: We support this use case with both EVPN-VXLAN edge-routed bridging overlays and EVPN-VXLAN centrally-routed bridging overlays.

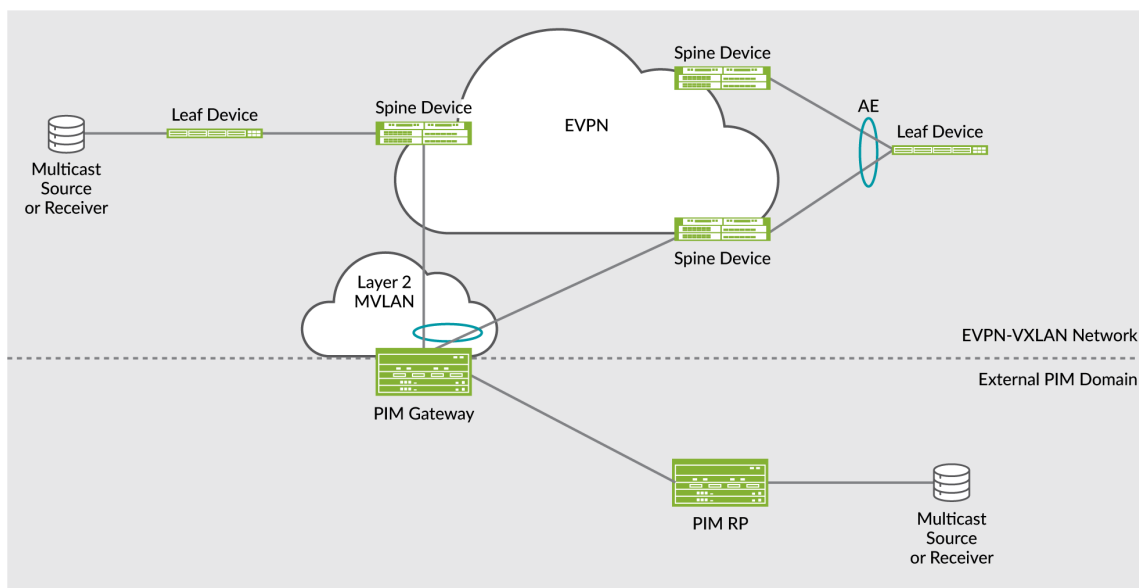
The use case provides a mechanism for the data center, which uses IGMP (or MLD) and PIM, to exchange multicast traffic with the external PIM domain. Using a Layer 2 multicast VLAN (MVLAN) and associated IRB interfaces on the EVPN devices in the data center to connect to the PIM domain, you can enable the forwarding of multicast traffic from:

- An external multicast source to internal multicast destinations
- An internal multicast source to external multicast destinations

NOTE: In this section, *external* refers to components in the PIM domain. *Internal* refers to components in your EVPN-VXLAN network that supports a data center.

Figure 65 on page 688 shows the required key components for this use case in a sample EVPN-VXLAN centrally-routed bridging overlay.

Figure 65: Use Case 3: PIM Gateway with Layer 2 Connectivity—Key Components



- Components in the PIM domain:
 - A PIM gateway that acts as an interface between an existing PIM domain and the EVPN-VXLAN network. The PIM gateway is a Juniper Networks or third-party Layer 3 device on which PIM and a routing protocol such as OSPF are configured. The PIM gateway does not run EVPN. You can connect the PIM gateway to one, some, or all EVPN devices.
 - A PIM rendezvous point (RP) is a Juniper Networks or third-party Layer 3 device on which PIM and a routing protocol such as OSPF are configured. You must also configure the PIM RP to translate PIM join or prune messages into corresponding IGMP (or MLD) report or leave messages then forward the reports and messages to the PIM gateway.
- Components in the EVPN-VXLAN network:

NOTE: These components are in addition to the components already configured for use cases 1 and 2.

- EVPN devices. For redundancy, we recommend multihoming the EVPN devices to the PIM gateway through an aggregated Ethernet interface on which you configure an Ethernet segment identifier (ESI). On each EVPN device, you must also configure the following for this use case:

- A Layer 2 multicast VLAN (MVLAN). The MVLAN is a VLAN that is used to connect the PIM gateway. In the MVLAN, PIM is enabled.
- An MVLAN IRB interface on which you configure PIM, IGMP snooping (or MLD snooping), and a routing protocol such as OSPF. To reach the PIM gateway, the EVPN device forwards multicast traffic out of this interface.
- To enable the EVPN devices to forward multicast traffic to the external PIM domain, configure:
 - PIM-to-IGMP translation:

For EVPN-VXLAN **edge-routed bridging overlays**, configure PIM-to-IGMP translation by including `pim-to-igmp-proxy upstream-interface irb-interface-name` configuration statements at the `[edit routing-options multicast]` hierarchy level. Specify the MVLAN IRB interface for the IRB interface parameter. You also must set IGMP passive mode using `igmp interface irb-interface-name passive` configuration statements at the `[edit protocols]` hierarchy level on the upstream interfaces where you set `pim-to-igmp-proxy`.

For EVPN-VXLAN **centrally-routed bridging overlays**, you do not need to include the `pim-to-igmp-proxy upstream-interface irb-interface-name` or `pim-to-ml-d-proxy upstream-interface irb-interface-name` configuration statements. In this type of overlay, the PIM protocol handles the routing of multicast traffic from the PIM domain to the EVPN-VXLAN network and vice versa.

- Multicast router interface:

Configure the multicast router interface by including the `multicast-router-interface` configuration statement at the `[edit routing-instances routing-instance-name bridge-domains bridge-domain-name protocols (igmp-snooping | mld-snooping) interface interface-name]` hierarchy level. For the interface name, specify the MVLAN IRB interface.

- PIM passive mode. For EVPN-VXLAN edge-routed bridging overlays only, you must ensure that the PIM gateway views the data center as only a Layer 2 multicast domain. To do so, include the passive configuration statement at the `[edit protocols pim]` hierarchy level.

Use Case 4: Inter-VLAN Multicast Routing and Forwarding—PIM Gateway with Layer 3 Connectivity

We recommend the PIM gateway with Layer 3 connectivity use case for EVPN-VXLAN centrally-routed bridging overlays only.

For this use case, we assume the following:

- You have deployed a EVPN-VXLAN network to support a data center.
- In this network, you have already set up:

- Intra-VLAN multicast traffic forwarding as described in use case 1.
- Inter-VLAN multicast traffic routing and forwarding as described in use case 2.
- There are multicast sources and receivers within the data center that you want to communicate with multicast sources and receivers in an external PIM domain.

NOTE: We recommend the PIM gateway with Layer 3 connectivity use case for EVPN-VXLAN centrally-routed bridging overlays only.

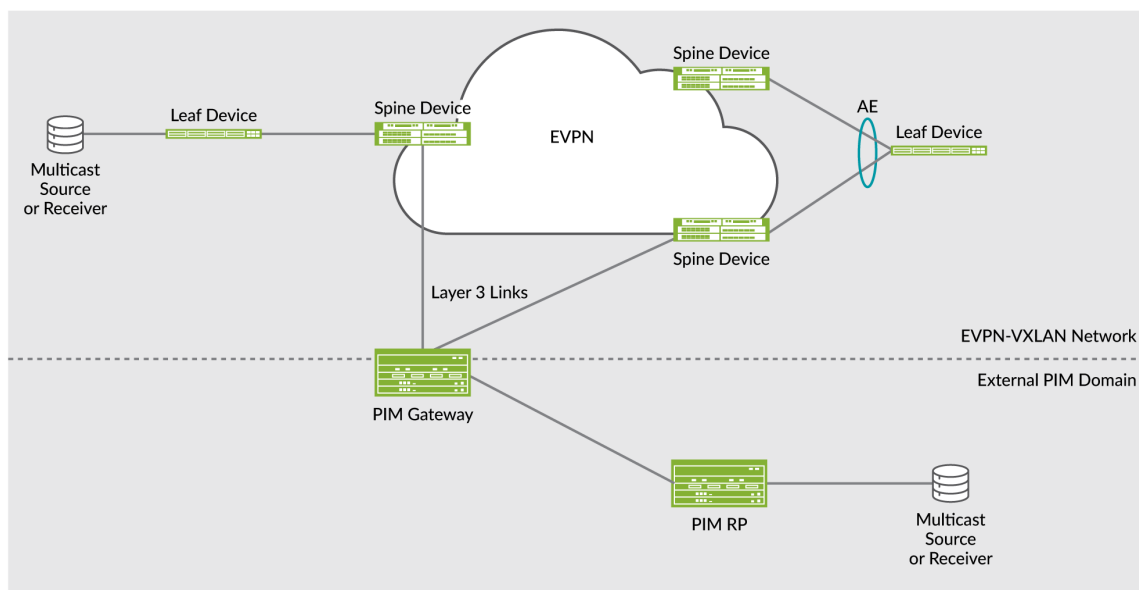
This use case provides a mechanism for the data center, which uses IGMP (or MLD) and PIM, to exchange multicast traffic with the external PIM domain. Using Layer 3 interfaces on the EVPN devices in the data center to connect to the PIM domain, you can enable the forwarding of multicast traffic from:

- An external multicast source to internal multicast destinations
- An internal multicast source to external multicast destinations

NOTE: In this section, *external* refers to components in the PIM domains. *Internal* refers to components in your EVPN-VXLAN network that supports a data center.

Figure 66 on page 691 shows the required key components for this use case in a sample EVPN-VXLAN centrally-routed bridging overlay.

Figure 66: Use Case 4: PIM Gateway with Layer 3 Connectivity—Key Components



- Components in the PIM domain:
 - A PIM gateway that acts as an interface between an existing PIM domain and the EVPN-VXLAN network. The PIM gateway is a Juniper Networks or third-party Layer 3 device on which PIM and a routing protocol such as OSPF are configured. The PIM gateway does not run EVPN. You can connect the PIM gateway to one, some, or all EVPN devices.
 - A PIM rendezvous point (RP) is a Juniper Networks or third-party Layer 3 device on which PIM and a routing protocol such as OSPF are configured. You must also configure the PIM RP to translate PIM join or prune messages into corresponding IGMP or MLD report or leave messages then forward the reports and messages to the PIM gateway.
- Components in the EVPN-VXLAN network:

NOTE: These components are in addition to the components already configured for use cases 1 and 2.

- EVPN devices. You can connect one, some, or all EVPN devices to a PIM gateway. You must make each connection through a Layer 3 interface on which PIM is configured. Other than the Layer 3 interface with PIM, this use case does not require additional configuration on the EVPN devices.

Use Case 5: Inter-VLAN Multicast Routing and Forwarding—External Multicast Router

Starting with Junos OS Release 17.3R1, you can configure an EVPN device to perform inter-VLAN forwarding of multicast traffic without having to configure IRB interfaces on the EVPN device. In such a scenario, an external multicast router is used to send IGMP or MLD queries to solicit reports and to forward VLAN traffic through a Layer 3 multicast protocol such as PIM. IRB interfaces are not supported with the use of an external multicast router.

For this use case, you must include the `igmp-snooping proxy` or `mld-snooping proxy` configuration statements at the `[edit routing-instances routing-instance-name protocols vlan vlan-name]` hierarchy level.

EVPN Multicast Flags Extended Community

Juniper Networks devices that support EVPN-VXLAN and IGMP snooping also support the EVPN multicast flags extended community. When you have enabled IGMP snooping on one of these devices, the device adds the community to EVPN Type 3 (Inclusive Multicast Ethernet Tag) routes.

The absence of this community in an EVPN Type 3 route can indicate the following about the device that advertises the route:

- The device does not support IGMP snooping.
- The device does not have IGMP snooping enabled on it.
- The device is running a Junos OS software release that doesn't support the community.
- The device does not support the advertising of EVPN Type 6 SMET routes.
- The device has IGMP snooping and a Layer 3 interface with PIM enabled on it. Although the Layer 3 interface with PIM performs snooping on the access side and selective multicast forwarding on the EVPN core, the device needs to attract all traffic to perform source registration to the PIM RP and inter-VLAN routing.

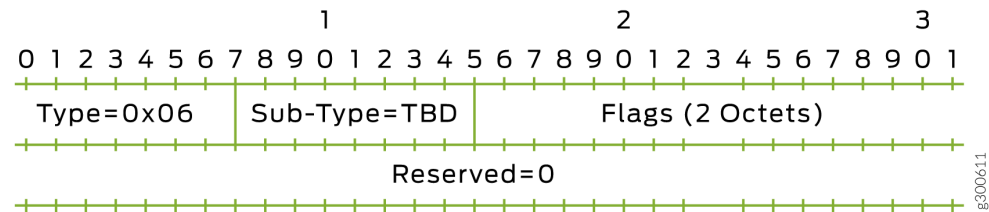
The behavior described above also applies to devices that support EVPN-VXLAN with MLD and MLD snooping.

[Figure 67 on page 693](#) shows the EVPN multicast flag extended community, which has the following characteristics:

- The community is encoded as an 8-bit value.
- The Type field has a value of 6.
- The IGMP Proxy Support flag is set to 1, which means that the device supports IGMP proxy.

The same applies to the MLD Proxy Support flag; if that flag is set to 1, the device supports MLD proxy. Either or both flags might be set.

Figure 67: EVPN Multicast Flag Extended Community



Release History Table

Release	Description
21.2R1	Starting with Junos OS Release 21.2R1, we support optimized inter-subnet multicast (OISM) routing and forwarding with IGMP snooping on QFX5110, QFX5120, and QFX10002 switches in EVPN-VXLAN edge-routed bridging overlay networks.
20.4R1	Starting in Junos OS Releases 20.4R1, in EVPN-VXLAN centrally-routed bridging overlay fabrics, QFX5110, QFX5120 and the QFX10000 line of switches support IGMPv3 with IGMP snooping for IPv4 multicast traffic, and MLD version 1 (MLDv1) and MLD version 2 (MLDv2) with MLD snooping for IPv6 multicast traffic.
19.3R1	Starting with Junos OS Release 19.3R1, EX9200 switches, MX Series routers, and vMX virtual routers support IGMP version 2 (IGMPv2) and IGMP version 3 (IGMPv3), IGMP snooping, selective multicast forwarding, external PIM gateways, and external multicast routers with an EVPN-VXLAN centrally-routed bridging overlay.
19.1R1	Starting with Junos OS Release 19.1R1, QFX5120-32C switches support IGMP snooping in EVPN-VXLAN centrally-routed and edge-routed bridging overlays.
18.4R2	Starting with Junos OS Release 18.4R2 (but not Junos OS Releases 19.1R1 and 19.2R1), QFX5120-48Y switches support IGMP snooping in an EVPN-VXLAN centrally-routed bridging overlay.
18.4R2	Starting in Junos OS Releases 18.4R2 and 19.1R2, selective multicast forwarding is enabled by default on QFX5110 and QFX5120 switches when you configure IGMP snooping in EVPN-VXLAN networks, further constraining multicast traffic flooding. With IGMP snooping and selective multicast forwarding, these switches send the multicast traffic only to interested receivers in both the EVPN core and on the access side for multicast traffic coming in either from an access interface or an EVPN network interface.

18.1R1	Starting with Junos OS Release 18.1R1, QFX5110 switches support IGMP snooping in an EVPN-VXLAN centrally-routed bridging overlay (EVPN-VXLAN topology with a two-layer IP fabric) for forwarding multicast traffic within VLANs.
17.3R1	Starting with Junos OS Release 17.3R1, QFX10000 switches support the exchange of traffic between multicast sources and receivers in an EVPN-VXLAN edge-routed bridging overlay, which uses IGMP, and sources and receivers in an external Protocol Independent Multicast (PIM) domain. A Layer 2 multicast VLAN (MVLAN) and associated IRB interfaces enable the exchange of multicast traffic between these two domains.
17.3R1	Starting with Junos OS Release 17.3R1, you can configure an EVPN device to perform inter-VLAN forwarding of multicast traffic without having to configure IRB interfaces on the EVPN device.
17.2R1	Starting with Junos OS Release 17.2R1, QFX10000 switches support IGMP snooping in an Ethernet VPN (EVPN)-Virtual Extensible LAN (VXLAN) edge-routed bridging overlay (EVPN-VXLAN topology with a collapsed IP fabric).

RELATED DOCUMENTATION

[distributed-dr](#)

[igmp-snooping](#)

[mld-snooping](#)

[multicast-router-interface](#)

[Example: Preserving Bandwidth with IGMP Snooping in an EVPN-VXLAN Environment | 694](#)

Example: Preserving Bandwidth with IGMP Snooping in an EVPN-VXLAN Environment

IN THIS SECTION

- [Requirements | 695](#)
- [Overview | 695](#)
- [Configuration | 697](#)
- [Verification | 739](#)

This example shows how to configure IGMP snooping on provider edge (PE) devices in an Ethernet VPN (EVPN)-Virtual Extensible LAN. When multicast traffic arrives at the VXLAN core, a PE device configured with EVPN forwards traffic only to the local access interfaces where there are IGMP listeners.

Requirements

This example uses the following hardware and software components:

- Two QFX10000 switches configured as multihomed PE devices that are connected to the CE, one QFX10000 device configured as a PE device connected to the multihomed PEs and a QFX5110 configured as a CE device.
- Junos OS Release 17.2R1 or later running on all devices.

Overview

IN THIS SECTION

- [Topology | 696](#)

IGMP snooping is used to constrain multicast traffic in a broadcast domain to interested receivers and multicast devices. In an environment with significant multicast traffic, IGMP snooping preserves bandwidth because multicast traffic is forwarded only on those interfaces where there are IGMP listeners. IGMP is enabled to manage multicast group membership.

When you enable IGMP snooping on each VLAN, the device advertises EVPN Type 7 and Type 8 routes (Join and Leave Sync Routes) to synchronize IGMP join and leave states among multihoming peer devices in the EVPN instance. On the access side, devices only forward multicast traffic to subscribed listeners. However, multicast traffic is still flooded in the EVPN core even when there are no remote receivers.

Configuring IGMP snooping in an EVPN-VXLAN environment requires the following:

- Multihoming peer PE devices in all-active mode.
- IGMP version 2 only. (IGMP versions 1 and 3 are not supported.)
- IGMP snooping configured in proxy mode for the PE to become the IGMP querier for the local access interfaces.

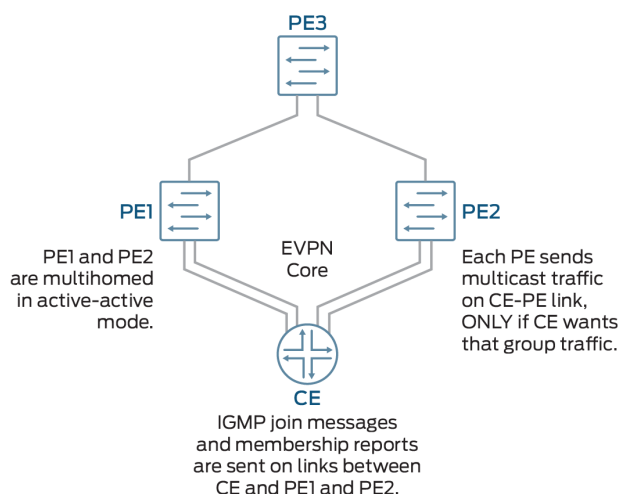
This feature supports both intra-VLAN and inter-VLAN multicast forwarding. You can configure a PE device to perform either or both. In this example, to enable inter-VLAN forwarding, each PE device is

configured as a statically defined Protocol Independent Multicast (PIM) rendezvous point (RP) to enable multicast forwarding. You also configure the `distributed-dr` statement at the `[edit protocols pim interface interface-name]` hierarchy level for each IRB interface. This mode enables PIM to forward multicast traffic more efficiently by disabling PIM features that are not required in this scenario. When you configure this statement, PIM ignores the designated router (DR) status of the interface when processing IGMP reports received on the interface. When the interface receives the IGMP report, the PE device sends PIM upstream join messages to pull the multicast stream and forward it to the interface—regardless of the DR status of the interface.

Topology

Figure 68 on page 696 illustrates an EVPN-VXLAN environment where two PE devices (PE1 and PE2) are connected to the customer edge (CE) device. These PEs are dual-homed in active-active mode to provide redundancy. A third PE device forwards traffic to the PE devices that face the CE. IGMP is enabled on integrated routing and bridging (IRB) interfaces. The CE device hosts five VLANs; IGMP snooping is enabled on all of the VLANs. Because this implementation does not support the use of a multicast router, each VLAN in the PE is enabled as an IGMP Layer 2 querier. The multihomed PE devices forward traffic towards the CE only on those interfaces where there are IGMP listeners.

Figure 68: IGMP Snooping in an EVPN-VXLAN Environment



Configuration

IN THIS SECTION

- [CLI Quick Configuration | 697](#)
- [Configuring PE1 | 704](#)
- [Configuring PE2 | 715](#)
- [Configuring CE Device | 725](#)
- [Configuring PE3 | 729](#)

To configure IGMP Snooping in an EVPN-VXLAN environment, perform these tasks:

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the [edit] hierarchy level, and then enter `commit` from configuration mode.

Device PE1

```
set chassis aggregated-devices ethernet device-count 2
set interfaces xe-0/0/0:1 description "Connected to CE"
set interfaces xe-0/0/0:2 ether-options 802.3ad ae1
set interfaces xe-0/0/1:0 enable
set interfaces xe-0/0/1:0 unit 0 description "Connected to PE3"
set interfaces xe-0/0/1:0 unit 0 family inet address 192.0.2.1/24
set interfaces ae0 enable
set interfaces ae1 enable
set interfaces lo0 unit 0 family inet address 192.168.1.1/32
set interfaces ae0 esi 00:11:11:11:11:11:11:11:11:11
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp periodic fast
set interfaces ae0 aggregated-ether-options lacp system-id 00:00:00:00:00:01
set interfaces ae0 unit 0 description "Connected to CE"
set interfaces ae1 esi 00:22:22:22:22:22:22:22:22:22
set interfaces ae1 esi all-active
set interfaces ae1 aggregated-ether-options lacp active
```

```

set interfaces ae1 aggregated-ether-options periodic fast
set interfaces ae1 aggregated-ether-options lacp system id 00:00:00:00:00:02
set interfaces ae1 unit 0 description "Connected to CE"
set interfaces ae0 family ethernet-switching interface-mode trunk
set interfaces ae0 family ethernet-switching vlan members [ VLAN1 VLAN2 VLAN3 VLAN4 VLAN5 ]
set interfaces ae1 unit 0 family ethernet-switching interface-mode trunk
set interfaces ae1 unit 0 family ethernet-switching vlan [ VLAN1 VLAN2 VLAN3 VLAN4 VLAN5 ]
set interfaces irb unit 1 family inet address 10.1.1.1/24 virtual-gateway-address 10.1.1.10
set interfaces irb unit 2 family inet address 10.2.1.1/24 virtual-gateway-address 10.2.1.10
set interfaces irb unit 3 family inet address 10.3.1.1/24 virtual-gateway-address 10.3.1.10
set interfaces irb unit 4 family inet address 10.4.1.1/24 virtual-gateway-address 10.4.1.10
set interfaces irb unit 5 family inet address 10.5.1.1/24 virtual-gateway-address 10.5.1.10
set routing-options router-id 192.168.1.1
set routing-options autonomous-system 65536
set protocols ospf area 0.0.0.0 interface xe-0/0/1:0.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols bgp group INT type internal
set protocols bgp group INT local-address 192.168.1.1
set protocols bgp group INT family evpn signaling
set protocols bgp group INT local-as 65536
set protocols bgp group INT neighbor 172.16.1.1
set vlans VLAN vlan-id 1
set vlans VLAN1 l3-interface irb.1
set vlans VLAN1 vxlan vni 1
set vlans VLAN2 vlan-id 2
set vlans VLAN2 l3-interface irb.2
set vlans VLAN2 vxlan vni2
set vlans VLAN3 vlan-id 3
set vlans VLAN3 l3-interface irb.3
set vlans VLAN3 vxlan vni 3
set vlans VLAN4 vlan-id 4
set vlans VLAN4 l3-interface irb.4
set vlans VLAN4 vxlan vni 4
set vlans VLAN5 vlan-id 5
set vlans VLAN5 l3-interface irb.5
set vlans VLAN5 vxlan vni 5
set protocols evpn encapsulation vxlan
set protocols evpn ingress-replication
set protocols evpn extended-vini-list 1-5
set policy-options policy-statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pplb then load-balance per-packet
set routing-options forwarding-table export evpn-pplb
set switch-options vtep-source-interface lo0:0

```

```

set switch-options route distinguisher 192.168.1.1:1
set switch-options vrf-target target:1:1
set protocols igmp interface irb.1
set protocols igmp interface irb.2
set protocols igmp interface irb.3
set protocols igmp interface irb.4
set protocols igmp interface irb.5
set protocols igmp-snooping vlan VLAN1 l2-querier source-address 10.1.1.1
set protocols igmp-snooping vlan VLAN1 proxy
set protocols igmp-snooping vlan VLAN2 l2-querier source-address 10.2.1.1
set protocols igmp-snooping vlan VLAN2 proxy
set protocols igmp-snooping vlan VLAN3 l2-querier source-address 10.3.1.1
set protocols igmp-snooping vlan VLAN3 proxy
set protocols igmp-snooping vlan VLAN4 l2-querier source-address 10.4.1.1
set protocols igmp-snooping vlan VLAN4 proxy
set protocols igmp-snooping vlan VLAN4 l2-querier source-address 10.5.1.1
set protocols igmp-snooping vlan VLAN5 proxy
set protocols pim rp static address 172.16.1.1
set protocols pim interface irb.1 distributed-dr
set protocols pim interface irb.2 distributed-dr
set protocols pim interface irb.3 distributed-dr
set protocols pim interface irb.4 distributed-dr
set protocols pim interface irb.5 distributed-dr

```

Device PE2

```

set chassis aggregated-devices ethernet device-count 2
set interfaces xe-0/0/0:0 ether-options 802.3ad ae1
set interfaces xe-0/0/0:1 description "Connected to CE"
set interfaces xe-0/0/0:1 ether-options 802.3ad ae0
set interfaces xe-0/0/1:0 enable
set interfaces xe-0/0/1:0 unit 0 description "Connected to PE3"
set interfaces xe-0/0/1:0 family inet address 198.51.100.1/24
set interfaces ae0 enable
set interfaces ae1 enable
set interfaces lo0 unit 0 family inet address 192.168.2.1/32
set interfaces ae0 esi 00:11:11:11:11:11:11:11:11:11
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated ether-options lacp periodic fast
set interfaces ae0 aggregated ether-options lacp system-id 00:00:00:00:00:01
set interfaces ae1 esi 00:22:22:22:22:22:22:22:22:22

```

```

set interfaces ae1 esi all-active
set interfaces ae1 aggregated-ether-options lacp active
set interfaces ae1 aggregated-ether-options lacp periodic fast
set interfaces ae1 aggregated-ether-options lacp system-id 00:00:00:00:00:02
set interfaces ae0 unit 0 description "Connected to CE"
set interfaces ae0 unit family ethernet-switching interface-mode trunk
set interfaces ae0 unit family ethernet-switching vlan members [ VLAN1 VLAN2 VLAN3 VLAN4 VLAN5 ]
set interfaces ae1 unit 0 description "Connected to CE"
set interfaces ae1 unit 0 family ethernet-switching interface-mode trunk
set interfaces ae1 unit 0 family ethernet-switching vlan members [ VLAN1 VLAN2 VLAN3 VLAN4
VLAN5 ]
set interfaces irb unit 1 family inet address 10.1.1.2/24 virtual-gateway-address 10.1.1.10
set interfaces irb unit 2 family inet address 10.2.1.2/24 virtual-gateway-address 10.2.1.10
set interfaces irb unit 3 family inet address 10.3.1.2/24 virtual-gateway-address 10.3.1.10
set interfaces irb unit 4 family inet address 10.4.1.2/24 virtual-gateway-address 10.4.1.10
set interfaces irb unit 5 family inet address 10.5.1.2/24 virtual-gateway-address 10.5.1.10
set routing-options router-id 192.168.2.1
set routing-options autonomous-system 65536
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface xe-0/0/1:0.0
set protocols bgp group INT type internal
set protocols bgp group INT local-address 192.168.2.1
set protocols bgp group INT family evpn signaling
set protocols bgp group INT local-as 65536
set protocols bgp group INT neighbor 172.16.1.1
set vlans VLAN1 vlan-id 1
set vlans VLAN1 l3-interface irb.1
set vlans VLAN1 vxlan vni 1
set vlans VLAN2 vlan-id 2
set vlans VLAN2 l3-interface irb.2
set vlans VLAN2 vxlan vni 2
set vlans VLAN3 vlan-id 3
set vlans VLAN3 l3-interface irb.3
set vlans VLAN3 vxlan vni 3
set vlans VLAN4 vlan-id 4
set vlans VLAN4 l3-interface irb.4
set vlans VLAN4 vxlan vni 4
set vlans VLAN5 vlan-id 5
set vlans VLAN5 l3-interface irb.5
set vlans VLAN5 vxlan vni 3
set protocols evpn encapsulation vxlan
set protocols evpn multicast-mode ingress-encapsulation
set protocols evpn extended-vni-list 1-5

```

```

set policy-options policy statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pplb then load-balance per-packet
set routing-options forwarding-table export evpn-pplb
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 192.168.2.1:1
set switch-options vrf-target target:1:1
set protocols igmp interface irb.1
set protocols igmp interface irb.2
set protocols igmp interface irb.3
set protocols igmp interface irb.4
set protocols igmp interface irb.5
set protocols igmp-snooping vlan VLAN1 l2-querier source-address 10.1.1.2
set protocols igmp-snooping vlan VLAN1 proxy
set protocols igmp-snooping vlan VLAN2 l2-querier source-address 10.2.1.2
set protocols igmp-snooping vlan VLAN2 proxy
set protocols igmp-snooping vlan VLAN3 l2-querier source-address 10.3.1.2
set protocols igmp-snooping vlan VLAN3 proxy
set protocols igmp-snooping vlan VLAN4 l2-querier source-address 10.4.1.2
set protocols igmp-snooping vlan VLAN4 proxy
set protocols igmp-snooping vlan VLAN5 l2-querier source-address 10.5.1.2
set protocols igmp-snooping vlan VLAN5 proxy
set protocols pim rp static address 172.16.1.1
set protocols pim interface irb.1 distributed-dr
set protocols pim interface irb.2 distributed-dr
set protocols pim interface irb.3 distributed-dr
set protocols pim interface irb.4 distributed-dr
set protocols pim interface irb.5 distributed-dr

```

CE

```

set chassis aggregated-devices ethernet device-count 2
set interfaces xe-0/2/0 ether-options 802.3ad ae0
set interfaces xe-0/2/1 ether-options 802.3ad ae0
set interfaces xe-0/2/0 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/2/0 unit 0 family ethernet-switching vlan members 1-5
set interfaces xe-0/2/1 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/2/1 unit 0 family ethernet-switching vlan members 1-5
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp periodic fast
set interfaces ae0 unit 0 family ethernet-switching interface-mode trunk
set interfaces ae0 unit 0 family ethernet-switching vlan members 1-5
set interfaces ae1 aggregated-ether-options lacp active

```



```

set interfaces ae1 aggregated-ether-options lacp periodic fast
set interfaces ae1 unit 0 family ethernet-switching interface-mode trunk
set interfaces ae1 unit 0 family ethernet-switching vlan members 1-5
set vlans BD-1 vlan-id 1
set vlans BD-2 vlan-id 2
set vlans BD-3 vlan-id 3
set vlans BD-4 vlan-id 4
set vlans BD-5 vlan-id 5

```

PE3

```

set interfaces xe-0/0/0 enable
set interfaces xe-0/0/0 unit 0 description "Connected to PE1"
set interfaces xe-0/0/0 unit 0 family inet 192.0.2.2/24
set interfaces xe-0/0/1 enable
set interfaces xe-0/0/1 unit 0 description "Connected to PE2"
set interfaces xe-0/0/1 unit 0 family inet 198.51.100.2/24
set interfaces lo0 unit 0 family inet address 172.16.1.1/32
set interfaces xe-0/0/0:1 enable
set interfaces xe-0/0/0:1 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/0:1 unit 0 family ethernet-switching vlan members [ VLAN1 VLAN2 VLAN3
VLAN4 VLAN5 ]
set interfaces xe-0/0/1:1 enable
set interfaces xe-0/0/1:1 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/1:1 unit 0 family ethernet-switching vlan members [ VLAN1 VLAN2 VLAN3
VLAN4 VLAN5 ]
set interfaces irb unit 1 family inet address 10.1.1.5/24 virtual-gateway address 10.1.1.10
set interfaces irb unit 2 family inet address 10.2.1.5/24 virtual-gateway address 10.2.1.10
set interfaces irb unit 3 family inet address 10.3.1.5/24 virtual-gateway address 10.3.1.10
set interfaces irb unit 4 family inet address 10.4.1.5/24 virtual-gateway address 10.4.1.10
set interfaces irb unit 5 family inet address 10.5.1.5/24 virtual-gateway address 10.5.1.10
set routing-options router-id 172.16.1.1
set routing-options autonomous-system 65536
set ospf area 0.0.0.0 interface all
set ospf area 0.0.0.0 fxp0.0 disable
set protocols bgp group INT type internal
set protocols bgp group INT local-address 172.16.1.1
set protocols bgp group INT family evpn signaling
set protocols bgp group INT local-as 65536
set protocols bgp group INT neighbor 192.168.1.1
set protocols bgp group INT neighbor 192.168.2.1
set vlans VLAN1 vlan-id 1

```

```

set vlans VLAN1 l3-interface irb.1
set vlans VLAN1 vxlan vni 1
set vlans VLAN2 vlan-id 2
set vlans VLAN2 l3-interface irb.2
set vlans VLAN2 vxlan vni 2
set vlans VLAN3 vlan-id 3
set vlans VLAN3 l3-interface irb.3
set vlans VLAN3 vxlan vni 3
set vlans VLAN4 vlan-id 4
set vlans VLAN4 l3-interface irb.4
set vlans VLAN4 vxlan vni 4
set vlans VLAN5 vlan-id 5
set vlans VLAN5 l3-interface irb.5
set vlans VLAN5 vxlan vni 5
set protocols evpn encapsulation vxlan
set protocols evpn multicast-mode ingress-replication
set protocols evpn extended-vni-list 1-5
set policy-options policy-statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pbllb then load-balance per packet
set routing-options forwarding-table export evpn-pplb
set switch-options vtep-source-interface lo0
set switch-options route-distinguisher 172.16.1.1:1
set switch-option vrf-target target:1:1
set protocols igmp interface irb.1
set protocols igmp interface irb.2
set protocols igmp interface irb.3
set protocols igmp interface irb.4
set protocols igmp interface irb.5
set protocols igmp-snooping vlan VLAN1 l2-querier source-address 10.1.1.5
set protocols igmp-snooping vlan VLAN1 proxy
set protocols igmp-snooping vlan VLAN2 l2-querier source-address 10.2.1.5
set protocols igmp-snooping vlan VLAN2 proxy
set protocols igmp-snooping vlan VLAN3 l2-querier source-address 10.3.1.5
set protocols igmp-snooping vlan VLAN3 proxy
set protocols igmp-snooping vlan VLAN4 l2-querier source-address 10.4.1.5
set protocols igmp-snooping vlan VLAN4 proxy
set protocols igmp-snooping vlan VLAN5 l2-querier source-address 10.5.1.5
set protocols igmp-snooping vlan VLAN5 proxy
set protocols pim rp local 172.16.1.1
set protocols pim interface irb.1 distributed-dr
set protocols pim interface irb.2 distributed-dr
set protocols pim interface irb.3 distributed-dr

```

```
set protocols pim interface irb.4 distributed-dr
set protocols pim interface irb.5 distributed-dr
```

Configuring PE1

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure device PE1:

1. Specify the number of aggregated Ethernet logical interfaces.

```
[edit chassis]
user@PE1# set aggregated-devices ethernet device-count 2
```

2. Configure the interfaces.

```
[edit interfaces]
user@PE1# set xe-0/0/0:1 description "Connected to CE"
user@PE1# set xe-0/0/0:2 ether-options 802.3ad ae1
user@PE1# set xe-0/0/1:0 enable
user@PE1# set xe-0/0/1:0 unit 0 description "Connected to PE3"
user@PE1# set xe-0/0/1:0 unit 0 family inet address 192.0.2.1/24
user@PE1# set ae0 enable
user@PE1# set ae1 enable
user@PE2# set lo0 unit 0 family inet address 192.168.1.1/32
```

3. Configure active-active multihoming and enable the Link Aggregation Control Protocol (LACP) on each aggregated Ethernet interface.

```
[edit interfaces]
user@PE1# set ae0 esi 00:11:11:11:11:11:11:11:11:11
user@PE1# set esi all-active
user@PE1# set ae0 aggregated-ether-options lacp active
user@PE1# set ae0 aggregated-ether-options lacp periodic-fast
```

```

user@PE1# set ae0 aggregated-ether-options lacp system-id 00:00:00:00:00:01
user@PE1# set ae1 esi 00:22:22:22:22:22:22:22
user@PE1# set ae1 esi all-active
user@PE1# set ae1 aggregated-ether-options lacp active
user@PE1# set ae1 aggregated-ether-options lacp periodic-fast
user@PE1# set ae0 aggregated-ether-options lacp system-id 00:00:00:00:00:02

```

4. Configure each aggregated Ethernet interface as a trunk port.

```

[edit interfaces]
user@PE1# set ae0 unit 0 description "Connected to CE"
user@PE1# set ae0 unit 0 family ethernet-switching interface-mode trunk
user@PE1# set ae0 unit 0 family ethernet-switching vlan members [ VLAN1 VLAN2 VLAN3 VLAN4
VLAN5 ]
user@PE1# set ae1 unit 0 description "Connected to CE"
user@PE1# set ae1 unit 0 family ethernet-switching interface-mode trunk
user@PE1# set ae1 unit 0 family ethernet-switching vlan members [ VLAN1 VLAN2 VLAN3 VLAN4
VLAN5 ]

```

5. Configure IRB interfaces and virtual-gateway addresses.

```

[edit interfaces]
user@PE1# set irb unit 1 family inet address 10.1.1.1/24 virtual-gateway-address 10.1.1.10
user@PE1# set irb unit 2 family inet address 10.2.1.1/24 virtual-gateway-address 10.2.1.10
user@PE1# set irb unit 3 family inet address 10.3.1.1/24 virtual-gateway-address 10.3.1.10
user@PE1# set irb unit 4 family inet address 10.4.1.1/24 virtual-gateway-address 10.4.1.10
user@PE1# set irb unit 5 family inet address 10.5.1.1/24 virtual-gateway-address 10.5.1.10

```

6. Configure the autonomous system.

```

[edit routing-options]
user@PE1# set router-id 192.168.1.1
user@PE1# set autonomous-system 65536

```

7. Configure OSPF.

```
[edit protocols ospf]
user@PE1# set area 0.0.0.0 interface xe-0/0/1:0.0
user@PE1# set area 0.0.0.0 interface lo0. passive
```

8. Configure BGP internal peering.

```
[edit protocols bgp]
user@PE1# set group INT type internal
user@PE1# set group INT local-address 192.168.1.1
user@PE1# set group INT family evpn signaling
user@PE1# set group INT local-as 65536
user@PE1# set group INT neighbor 172.16.1.1
```

9. Configure the VLANs.

```
[set vlans]
user@PE1# set VLAN1 vlan-id 1
user@PE1# set VLAN1 l3-interface irb.1
user@PE1# set VLAN1 vxlan vni 1
user@PE1# set VLAN2 vlan-id 2
user@PE1# set VLAN2 l3-interface irb.2
user@PE1# set VLAN2 vxlan vni 2
user@PE1# set VLAN3 vlan-id 3
user@PE1# set VLAN3 l3-interface irb.3
user@PE1# set VLAN3 vxlan vni 3
user@PE1# set VLAN4 vlan-id 4
user@PE1# set VLAN4 l3-interface irb.4
user@PE1# set VLAN4 vxlan vni 4
user@PE1# set VLAN5 vlan-id 5
user@PE1# set VLAN5 l3-interface irb.5
user@PE1# set VLAN2 vxlan vni 5
```

10. Enable EVPN.

```
[edit protocols evpn]
user@PE1# set encapsulation vxlan
```

```

user@PE1# set multicast-mode ingress-replication
user@PE1# set extended-vni-list 1-5

```

11. Configure an export routing policy to load balance EVPN traffic.

```

[edit policy-options]
user@PE1# set policy-statement evpn-pplb from protocol evpn
user@PE1# set policy-statement evpn-pplb then load-balance per packet

[edit routing-options]
user@PE1# set forwarding-table export evpn-pplb

```

12. Configure the source interface for the VXLAN tunnel.

```

[edit switch-options]
user@PE1# set vtep-source-interface lo0.0
user@PE1# set route-distinguisher 192.168.1.1:1
user@PE1# set vrf-target target:1:1

```

13. Enable IGMP on the IRB interfaces associated with the VLANs.

```

[edit protocols igmp]
user@PE1# set interface irb.1
user@PE1# set interface irb.2
user@PE1# set interface irb.3
user@PE1# set interface irb.4
user@PE1# set interface irb.5

```

14. Enable IGMP snooping on the VLANs.

```

[edit protocols igmp-snooping vlan]
user@PE1# set VLAN1 l2-querier source-address 10.1.1.1
user@PE1# set VLAN1 proxy
user@PE1# set VLAN2 l2-querier source-address 10.2.1.1
user@PE1# set VLAN2 proxy
user@PE1# set VLAN3 l2-querier source address 10.3.1.1
user@PE1# set VLAN3 proxy
user@PE1# set VLAN4 l2-querier source-address 10.4.1.1
user@PE1# set VLAN4 proxy

```

```
user@PE1# set VLAN5 l2-querier source-address 10.5.1.1
user@PE1# set VLAN5 proxy
```

15. Configure PIM by defining a static rendezvous point and enabling on the IRB interfaces associated with the VLANs..

NOTE: This step is required only if you want to configure inter-VLAN forwarding. If your PE device is performing only intra-VLAN forwarding, omit this step.

```
[edit protocols pim]
user@PE1# set rp static address 172.16.1.1
user@PE1# set interface irb.1 distributed-dr
user@PE1# set interface irb.2 distributed-dr
user@PE1# set interface irb.3 distributed-dr
user@PE1# set interface irb.4 distributed-dr
user@PE1# set interface irb.5 distributed-dr
```

Results

From configuration mode, confirm your configuration by entering the `show chassis`, `show interfaces`, `show routing-options`, `show protocols`, `show vlans`, `show policy-options`, and `show switch-options` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show chassis
aggregated-devices {
  ethernet {
    device-count 2;
  }
}
```

```
user@PE1# show
interfaces {
  xe-0/0/0:1 {
    description "Connected to CE";
    ether-options {
      802.3ad ae0;
```

```

    }
}
xe-0/0/0:2 {
    ether-options {
        802.3ad ae1;
    }
}
xe-0/0/1:0 {
    enable;
    unit 0 {
        description "Connected to PE3";
        family inet {
            address 192.0.2.1/24;
        }
    }
}
xe-0/0/1:1 {
    enable;
}
ae0 {
    enable;
    esi {
        00:11:11:11:11:11:11:11:11:11;
        all-active;
    }
    aggregated-ether-options {
        lacp {
            active;
            periodic fast;
            system-id 00:00:00:00:00:01;
        }
    }
    unit 0 {
        description "Connected to CE";
        family ethernet-switching {
            interface-mode trunk;
            vlan {
                members [ VLAN1 VLAN2 VLAN3 VLAN4 VLAN5 ];
            }
        }
    }
}
ae1 {

```



```

enable;
esi {
    00:22:22:22:22:22:22:22:22:22;
    all-active;
}
aggregated-ether-options {
    lacp {
        active;
        periodic fast;
        system-id 00:00:00:00:00:02;
    }
}
unit 0 {
    description "Connected to CE";
    family ethernet-switching {
        interface-mode trunk;
        vlan {
            members [ VLAN1 VLAN2 VLAN3 VLAN4 VLAN5 ];
        }
    }
}
}
irb {
    unit 1 {
        family inet {
            address 10.1.1.1/24 {
                virtual-gateway-address 10.1.1.10;
            }
        }
    }
    unit 2 {
        family inet {
            address 10.2.1.1./24 {
                virtual-gateway-address 10.2.1.10;
            }
        }
    }
    unit 3 {
        family inet {
            address 10.3.1.1./24 {
                virtual-gateway-address 10.3.1.10;
            }
        }
    }
}

```

```

    }
    unit 4 {
        family inet {
            address 10.4.1.1./24 {
                virtual-gateway-address 10.4.1.10;
            }
        }
    }
    unit 5 {
        family inet {
            address 10.5.1.1./24 {
                virtual-gateway-address 10.5.1.10;
            }
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.1.1/32;
        }
    }
}
routing-options {
    router-id 192.168.1.1;
    autonomous-system 65536;
    forwarding-table {
        export evpn-pplb;
    }
}
protocols {
    igmp {
        interface irb.1;
        interface irb.2;
        interface irb.3;
        interface irb.4;
        interface irb.5;
    }
    bgp {
        group INT {
            type internal;
            local-address 192.168.1.1;

```

```

        family evpn {
            signaling;
        }
        local-as 65536;
        neighbor 172.16.1.1;
    }
}
ospf {
    area 0.0.0.0 {
        interface xe-0/0/1:0.0;
        interface lo0.0 {
            passive;
        }
    }
}
pim {
    rp {
        static {
            address 172.16.1.1;
        }
    }
    interface irb.1 {
        distributed-dr;
    }
    interface irb.2 {
        distributed-dr;
    }
    interface irb.3 {
        distributed-dr;
    }
    interface irb.4 {
        distributed-dr;
    }
    interface irb.5 {
        distributed-dr;
    }
}
evpn {
    encapsulation vxlan;
    multicast-mode ingress-replication;
    extended-vni-list 1-5;
}
igmp-snooping {

```

```

    vlan VLAN1 {
        l2-querier {
            source-address 10.1.1.1;
        }
        proxy;
    }
    vlan VLAN2 {
        l2-querier {
            source-address 10.2.1.1;
        }
        proxy;
    }
    vlan VLAN3 {
        l2-querier {
            source-address 10.3.1.1;
        }
        proxy;
    }
    vlan VLAN4 {
        l2-querier {
            source-address 10.4.1.1;
        }
        proxy;
    }
    vlan VLAN5 {
        l2-querier {
            source-address 10.5.1.1;
        }
        proxy;
    }
}

policy-options {
    policy-statement evpn-pplb {
        from protocol evpn;
        then {
            load-balance per-packet;
        }
    }
}

switch-options {
    vtep-source-interface lo0.0;
    route-distinguisher 192.168.1.1:1;
}

```

```
    vrf-target target:1:1;
}
vlans {
    VLAN1 {
        vlan-id 1;
        l3-interface irb.1;
        vxlan {
            vni 1;
        }
    }
    VLAN2 {
        vlan-id 2;
        l3-interface irb.2;
        vxlan {
            vni 2;
        }
    }
    VLAN3 {
        vlan-id 3;
        l3-interface irb.3;
        vxlan {
            vni 3;
        }
    }
    VLAN4 {
        vlan-id 4;
        l3-interface irb.4;
        vxlan {
            vni 4;
        }
    }
    VLAN5 {
        vlan-id 5;
        l3-interface irb.5;
        vxlan {
            vni 5;
        }
    }
}
```

Configuring PE2

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure device PE2:

1. Specify the number of aggregated Ethernet logical interfaces.

```
[edit chassis]
user@PE2# set aggregated-devices ethernet device-count 2
```

2. Configure the interfaces.

```
[edit interfaces]
user@PE2# set xe-0/0/0:0 ether-options 802.3ad ae1
user@PE2# set xe-0/0/0:1 description "Connected to CE"
user@PE2# set xe-0/0/1:0 enable
user@PE2# set xe-0/0/1:0 unit 0 description "Connected to PE3"
user@PE2# set xe-0/0/1:0 unit 0 family inet address 198.51.100.1/24
user@PE2# set ae0 enable
user@PE2# set ae1 enable
user@PE2# set lo0 unit 0 family inet address 192.168.2.1/32
```

3. Configure active-active multihoming and enable the Link Aggregation Control Protocol (LACP) on each aggregated Ethernet interface.

```
[edit interfaces]
user@PE2# set ae0 esi 00:11:11:11:11:11:11:11:11
user@PE2# set esi all-active
user@PE2# set ae0 aggregated-ether-options lacp active
user@PE2# set ae0 aggregated-ether-options lacp periodic-fast
user@PE2# set ae0 aggregated-ether-options lacp system-id 00:00:00:00:00:01
user@PE2# set ae1 esi 00:22:22:22:22:22:22:22:22
user@PE2# set ae1 esi all-active
user@PE2# set ae1 aggregated-ether-options lacp active
```

```

user@PE2# set ae1 aggregated-ether-options lacp periodic-fast
user@PE2# set ae0 aggregated-ether-options lacp system-id 00:00:00:00:00:02

```

4. Configure each aggregated Ethernet interface as a trunk port.

```

[edit interfaces]
user@PE2# set ae0 unit 0 description "Connected to CE"
user@PE2# set ae0 unit 0 family ethernet-switching interface-mode trunk
user@PE2# set ae0 unit 0 family ethernet-switching vlan members [ VLAN1 VLAN2 VLAN3 VLAN4
VLAN5 ]
user@PE2# set ae1 unit 0 description "Connected to CE"
user@PE2# set ae1 unit 0 family ethernet-switching interface-mode trunk
user@PE2# set ae1 unit 0 family ethernet-switching vlan members [ VLAN1 VLAN2 VLAN3 VLAN4
VLAN5 ]

```

5. Configure IRB interfaces and virtual-gateway addresses.

```

[edit interfaces]
user@PE2# set irb unit 1 family inet address 10.1.1.2/24 virtual-gateway-address 10.1.1.10
user@PE2# set irb unit 2 family inet address 10.2.1.2/24 virtual-gateway-address 10.2.1.10
user@PE2# set irb unit 3 family inet address 10.3.1.2/24 virtual-gateway-address 10.3.1.10
user@PE2# set irb unit 4 family inet address 10.4.1.2/24 virtual-gateway-address 10.4.1.10
user@PE2# set irb unit 5 family inet address 10.5.1.2/24 virtual-gateway-address 10.5.1.10

```

6. Configure the autonomous system.

```

[edit routing-options]
user@PE2# set router-id 192.168.2.1
user@PE2# set autonomous-system 65536

```

7. Configure OSPF.

```

[edit protocols ospf]
user@PE2# set area 0.0.0.0 interface xe-0/0/1:0.0
user@PE2# set area 0.0.0.0 interface lo0. passive

```

8. Configure BGP internal peering.

```
[edit protocols bgp]
user@PE2# set group INT type internal
user@PE2# set group INT local-address 192.168.2.1
user@PE2# set group INT family evpn signaling
user@PE2# set group INT local-as 65536
user@PE2# set group INT neighbor 172.16.1.1
```

9. Configure the VLANs.

```
[edit vlans]
user@PE2# set VLAN1 vlan-id 1
user@PE2# set VLAN1 l3-interface irb.1
user@PE2# set VLAN1 vxlan vni 1
user@PE2# set VLAN2 vlan-id 2
user@PE2# set VLAN2 l3-interface irb.2
user@PE2# set VLAN2 vxlan vni 2
user@PE2# set VLAN3 vlan-id 3
user@PE2# set VLAN3 l3-interface irb.3
user@PE2# set VLAN3 vxlan vni 3
user@PE2# set VLAN4 vlan-id 4
user@PE2# set VLAN4 l3-interface irb.4
user@PE2# set VLAN4 vxlan vni 4
user@PE2# set VLAN5 vlan-id 5
user@PE2# set VLAN5 l3-interface irb.5
user@PE2# set VLAN2 vxlan vni 5
```

10. Enable EVPN.

```
[edit protocols evpn]
user@PE2# set encapsulation vxlan
user@PE2# set multicast-mode ingress-replication
user@PE2# set extended-vni-list 1-5
```


11. Configure an export routing policy to load balance EVPN traffic and apply it to the forwarding-table.

```
[edit policy-options]
user@PE2# set policy-statement evpn-pplb from protocol evpn
user@PE2# set policy-statement evpn-pplb then load-balance per packet

[edit routing-options]
user@PE2# set forwarding-table export evpn-pplb
```

12. Configure the source interface for the VXLAN tunnel.

```
[edit switch-options]
user@PE2# set vtep-source-interface lo0.0
user@PE2# set route-distinguisher 192.168.2.1:1
user@PE2# set vrf-target target:1:1
```

13. Enable IGMP on the IRB interfaces.

```
[edit protocols igmp]
user@PE2# set interface irb.1
user@PE2# set interface irb.2
user@PE2# set interface irb.3
user@PE2# set interface irb.4
user@PE2# set interface irb.5
```

14. Enable IGMP snooping on the IRB interfaces.

```
[edit protocols igmp-snooping vlan]
user@PE2# set VLAN1 12-querier source-address 10.1.1.2
user@PE2# set VLAN1 proxy
user@PE2# set VLAN2 12-querier source-address 10.2.1.2
user@PE2# set VLAN2 proxy
user@PE2# set VLAN3 12-querier source address 10.3.1.2
user@PE2# set VLAN3 proxy
user@PE2# set VLAN4 12-querier source-address 10.4.1.2
user@PE2# set VLAN4 proxy
```

```

user@PE2# set VLAN5 l2-querier source-address 10.5.1.2
user@PE2# set VLAN5 proxy

```

15. Configure PIM by defining a static rendezvous point and enabling on the IRB interfaces.

NOTE: This step is required only if you want to configure inter-VLAN forwarding. If your PE device is performing only intra-VLAN forwarding, omit this step.

```

[edit protocols pim]
user@PE2# set rp static address 172.16.1.1
user@PE2# set interface irb.1 distributed-dr
user@PE2# set interface irb.2 distributed-dr
user@PE2# set interface irb.3 distributed-dr
user@PE2# set interface irb.4 distributed-dr
user@PE2# set interface irb.5 distributed-dr

```

Results

From configuration mode, confirm your configuration by entering the `show chassis`, `show interfaces`, `show routing-options`, `show protocols`, `show vlans`, `show policy-options`, and `show switch-options` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```

user@PE2# show chassis
aggregated-devices {
  ethernet {
    device-count 2;
  }
}

```

```

user@PE2# show
interfaces {
  xe-0/0/0:0 {
    ether-options {
      802.3ad ae1;
    }
  }
}

```

```

xe-0/0/0:1 {
    description "Connected to CE";
    ether-options {
        802.3ad ae0;
    }
}
xe-0/0/1:0 {
    enable;
    unit 0 {
        description "Connected to PE3";
        family inet {
            address 198.51.100.1/24;
        }
    }
}
ae0 {
    esi {
        00:11:11:11:11:11:11:11:11:11;
        all-active;
    }
    aggregated-ether-options {
        lacp {
            active;
            periodic fast;
            system-id 00:00:00:00:00:01;
        }
    }
    unit 0 {
        description "Connected to CE";
        family ethernet-switching {
            interface-mode trunk;
            vlan {
                members [ VLAN1 VLAN2 VLAN3 VLAN4 VLAN5 ];
            }
        }
    }
}
ae1 {
    enable;
    esi {
        00:22:22:22:22:22:22:22:22:22;
        all-active;
    }
}

```

```

aggregated-ether-options {
    lacp {
        active;
        periodic fast;
        system-id 00:00:00:00:00:02;
    }
}
unit 0 {
    description "CONNECTED TO CE";
    family ethernet-switching {
        interface-mode trunk;
        vlan {
            members [ VLAN1 VLAN2 VLAN3 VLAN4 VLAN5 ];
        }
    }
}
}
irb {
    unit 1 {
        family inet {
            address 10.1.1.2/24 {
                virtual-gateway-address 10.1.1.10;
            }
        }
    }
    unit 2 {
        family inet {
            address 10.2.1.2/24 {
                virtual-gateway-address 10.2.1.10;
            }
        }
    }
    unit 3 {
        family inet {
            address 10.3.1.2/24 {
                virtual-gateway-address 10.3.1.10;
            }
        }
    }
    unit 4 {
        family inet {
            address 10.4.1.2/24 {
                virtual-gateway-address 10.4.1.10;
            }
        }
    }
}

```

```

        }
    }
}
unit 5 {
    family inet {
        address 10.5.1.2/24 {
            virtual-gateway-address 10.5.1.10;
        }
    }
}
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.2.1/32;
        }
    }
}
}
routing-options {
    router-id 192.168.2.1;
    autonomous-system 65536;
    forwarding-table {
        export evpn-pplb;
    }
}
protocols {
    igmp {
        interface irb.1;
        interface irb.2;
        interface irb.3;
        interface irb.4;
        interface irb.5;
    }
    bgp {
        group INT {
            type internal;
            local-address 192.168.2.1;
            family evpn {
                signaling;
            }
            local-as 65536;
            neighbor 172.16.1.1;

```

```

    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.0 {
            passive;
        }
        interface xe-0/0/1:0.0;
    }
}
pim {
    rp {
        static {
            address 172.16.1.1;
        }
    }
    interface irb.1 {
        distributed-dr;
    }
    interface irb.2 {
        distributed-dr;
    }
    interface irb.3 {
        distributed-dr;
    }
    interface irb.4 {
        distributed-dr;
    }
    interface irb.5 {
        distributed-dr;
    }
}
evpn {
    encapsulation vxlan;
    multicast-mode ingress-replication;
    extended-vni-list 1-5;
}
igmp-snooping {
    vlan VLAN1 {
        l2-querier {
            source-address 10.1.1.2;
        }
    }
    proxy;
}

```

```

    }
    vlan VLAN2 {
        l2-querier {
            source-address 10.2.1.2;
        }
        proxy;
    }
    vlan VLAN3 {
        l2-querier {
            source-address 10.3.1.2;
        }
        proxy;
    }
    vlan VLAN4 {
        l2-querier {
            source-address 10.4.1.2;
        }
        proxy;
    }
    vlan VLAN5 {
        l2-querier {
            source-address 10.5.1.2;
        }
        proxy;
    }
}

policy-options {
    policy-statement evpn-pplb {
        from protocol evpn;
        then {
            load-balance per-packet;
        }
    }
}

switch-options {
    vtep-source-interface lo0.0;
    route-distinguisher 192.168.2.1:1;
    vrf-target target:1:1;
}

vlans {
    VLAN1 {
        vlan-id 2;
    }
}

```

```

        l3-interface irb.2;
        vxlan {
            vni 2;
        }
    }
    VLAN2 {
        vlan-id 1;
        l3-interface irb.1;
        vxlan {
            vni 1;
        }
    }
    VLAN3 {
        vlan-id 3;
        l3-interface irb.3;
        vxlan {
            vni 3;
        }
    }
    VLAN4 {
        vlan-id 4;
        l3-interface irb.4;
        vxlan {
            vni 4;
        }
    }
    VLAN5 {
        vlan-id 5;
        l3-interface irb.5;
        vxlan {
            vni 5;
        }
    }
}

```

Configuring CE Device

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure device CE:

1. Specify the number of aggregated Ethernet logical interfaces.

```
[edit chassis]
user@CE# set aggregated-devices ethernet device-count 2
```

2. Configure the interfaces and enable LACP on the aggregated Ethernet interfaces.

```
[edit interfaces]
user@CE# set xe-0/2/0 ether-options 802.3ad ae0
user@CE# set xe-0/2/1 ether-options 802.3ad ae0
user@CE# set ae0 aggregated ether-options lacp active
user@CE# set ae0 aggregated ether-options lacp periodic fast
user@CE# set ae1 aggregated ether-options lacp active
user@CE# set ae1 aggregated ether-options lacp periodic fast
```

3. Create the Layer 2 customer bridge domains and the VLANs associated with the domains.

```
[edit]
user@CE# set vlans BD-1 vlan-id 1
user@CE# set vlans BD-2 vlan-id 2
user@CE# set vlans BD-3 vlan-id 3
user@CE# set vlans BD-4 vlan-id 4
user@CE# set vlans BD-5 vlan-id 5
```

4. Configure each interface to include in CE domain as a trunk port for accepting packets tagged with the specified VLAN identifiers.

```
[edit interfaces]
user@CE# set xe-0/2/0 unit 0 family ethernet-switching interface-mode trunk
user@CE# set xe-0/2/0 unit 0 family ethernet-switching vlan members 1-5
user@CE# set xe-0/2/1 unit 0 family ethernet-switching interface-mode trunk
user@CE# set xe-0/2/1 unit 0 family ethernet-switching vlan members 1-5
user@CE# set ae0 unit 0 family ethernet-switching interface-mode trunk
user@CE# set ae0 unit 0 family ethernet-switching vlan members 1-5
user@CE# set ae1 unit 0 family ethernet-switching interface-mode trunk
user@CE# set ae1 unit 0 family ethernet-switching vlan members 1-5
```

Results

From configuration mode, confirm your configuration by entering the `show chassis` and `show interfaces` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@CE# show chassis
aggregated-devices {
  ethernet {
    device-count 2;
  }
}
```

```
user@CE# show interfaces
xe-0/2/0 {
  ether-options {
    802.3ad ae0;
  }
  unit 0 {
    family ethernet-switching {
      interface-mode trunk;
      vlan {
        members 1-5;
      }
    }
  }
}
xe-0/2/1 {
  ether-options {
    802.3ad ae0;
  }
  unit 0 {
    family ethernet-switching {
      interface-mode trunk;
      vlan {
        members 1-5;
      }
    }
  }
}
ae0 {
```

```

    aggregated-ether-options {
        lacp {
            active;
            periodic fast;
        }
    }
    unit 0 {
        family ethernet-switching {
            interface-mode trunk;
            vlan {
                members 1-5;
            }
        }
    }
}
ae1 {
    aggregated-ether-options {
        lacp {
            active;
            periodic fast;
        }
    }
    unit 0 {
        family ethernet-switching {
            interface-mode trunk;
            vlan {
                members 1-5;
            }
        }
    }
}
}

```

```

user@CE# show vlans
BD-1 {
    vlan-id 1;
    domain-type bridge;
}
BD-2 {
    vlan-id 2;
    domain-type bridge;
}

```

```

BD-3 {
    vlan-id 3;
    domain-type bridge;
}
BD-3 {
    vlan-id 3;
    domain-type bridge;
}
BD-4 {
    vlan-id 4;
    domain-type bridge;
}
BD-5 {
    vlan-id 5;
    domain-type bridge;
}

```

Configuring PE3

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure device PE3:

1. Configure the interfaces.

```

[edit interfaces]
user@PE3# set xe-0/0/0 enable
user@PE3# set xe-0/0/0 unit 0 description "Connected to PE1"
user@PE3# set xe-0/0/0 unit 0 family inet 192.0.2.2/24
user@PE3# set xe-0/0/1 unit 0 description "Connected to PE2"
user@PE3# set xe-0/0/1 198.51.100.2/24
user@PE3# set lo0 unit 0 family inet address 172.16.1.1/32

```

2. Configure each logical Ethernet interface as a trunk port for accepting packets tagged with the specified VLAN identifiers.

```
[edit interfaces]
user@PE3# set xe-0/0/0:1 enable
user@PE3# set xe-0/0/0:1 unit 0 family ethernet-switching interface-mode trunk
user@PE3# set xe-0/0/0:1 unit 0 family ethernet-switching vlan members [ VLAN1 VLAN2 VLAN3
VLAN4 VLAN5 ]
user@PE3# set xe-0/0/1:1 enable
user@PE3# set xe-0/0/1:1 unit 0 family ethernet-switching interface-mode trunk
user@PE3# set xe-0/0/1:1 unit 0 family ethernet-switching vlan members [ VLAN1 VLAN2 VLAN3
VLAN4 VLAN5 ]
```

3. Configure IRB interfaces and virtual-gateway addresses.

```
[edit interfaces]
user@PE3# set irb unit 1 family inet address 10.1.1.5/24 virtual-gateway-address 10.1.1.10
user@PE3# set irb unit 2 family inet address 10.2.1.5/24 virtual-gateway-address 10.2.1.10
user@PE3# set irb unit 3 family inet address 10.3.1.5/24 virtual-gateway-address 10.3.1.10
user@PE3# set irb unit 4 family inet address 10.4.1.5/24 virtual-gateway-address 10.4.1.10
user@PE3# set irb unit 5 family inet address 10.5.1.5/24 virtual-gateway-address 10.5.1.10
```

4. Configure the autonomous system.

```
[edit routing-options]
user@PE3# set router-id 172.16.1.1
user@PE3# set autonomous-system 65536
```

5. Configure OSPF

```
[edit protocols ospf]
user@PE3# set area 0.0.0.0 interface all
user@PE3# set ospf area 0.0.0.0 fxp0.0 disable
```

6. Configure BGP internal peering with PE1 and PE2.

```
[edit protocols bgp]
user@PE3# set group INT type internal
```

```

user@PE3# set group INT local-address 172.16.1.1
user@PE3# set group INT family evpn signaling
user@PE3# set group INT local-as 65536
user@PE3# set group INT neighbor 192.168.1.1
user@PE3# set group INT neighbor 192.168.2.1

```

7. Configure the VLANs.

```

[edit vlans]
user@PE3# set VLAN1 vlan-id 1
user@PE3# set VLAN1 l3-interface irb.1
user@PE3# set VLAN1 vxlan vni 1
user@PE3# set VLAN2 vlan-id 2
user@PE3# set VLAN2 l3-interface irb.2
user@PE3# set VLAN2 vxlan vni 2
user@PE3# set VLAN3 vlan-id 3
user@PE3# set VLAN3 l3-interface irb.3
user@PE3# set VLAN3 vxlan vni 3
user@PE3# set VLAN4 vlan-id 4
user@PE3# set VLAN4 l3-interface irb.4
user@PE3# set VLAN4 vxlan vni 4
user@PE3# set VLAN5 vlan-id 5
user@PE3# set VLAN5 l3-interface irb.5
user@PE3# set VLAN2 vxlan vni 5

```

8. Enable EVPN.

```

[edit protocols evpn]
user@PE3# set encapsulation vxlan
user@PE3# set multicast-mode ingress-replication
user@PE3# set extended-vni-list 1-5

```

9. Configure an export routing policy to load balance EVPN traffic.

```

[edit policy-options]
user@PE3# set policy-statement evpn-pplb from protocol evpn
user@PE3# set policy-statement evpn-pplb then load-balance per packet

```

```
[edit routing-options]
user@PE3# set forwarding-table export evpn-pplb
```

10. Configure the source interface for the VXLAN tunnel.

```
[edit switch-options]
user@PE3# set vtep-source-interface lo0.0
user@PE3# set route-distinguisher 172.16.1.1:1
user@PE3# set vrf-target target:1:1
```

11. Enable IGMP on the IRB interfaces.

```
[edit protocols igmp]
user@PE1# set interface irb.1
user@PE1# set interface irb.2
user@PE1# set interface irb.3
user@PE1# set interface irb.4
user@PE1# set interface irb.5
```

12. Enable IGMP snooping on the IRB interfaces.

```
[edit protocols igmp-snooping vlan]
user@PE1# set VLAN1 l2-querier source-address 10.1.1.5
user@PE1# set VLAN1 proxy
user@PE1# set VLAN2 l2-querier source-address 10.2.1.5
user@PE1# set VLAN2 proxy
user@PE1# set VLAN3 l2-querier source address 10.3.1.5
user@PE1# set VLAN3 proxy
user@PE1# set VLAN4 l2-querier source-address 10.4.1.5
user@PE1# set VLAN4 proxy
user@PE1# set VLAN5 l2-querier source-address 10.5.1.5
user@PE1# set VLAN5 proxy
```

13. Configure PIM by defining the local rendezvous point and enabling on the IRB interfaces.

NOTE: This step is required only if you want to configure inter-VLAN forwarding. If your PE device is performing only intra-VLAN forwarding, omit this step.

```
[edit protocols pim]
user@PE1# set rp local address 172.16.1:1
user@PE1# set interface irb.1 distributed-dr
user@PE1# set interface irb.2 distributed-dr
user@PE1# set interface irb.3 distributed-dr
user@PE1# set interface irb.4 distributed-dr
user@PE1# set interface irb.5 distributed-dr
```

Results

From configuration mode, confirm your configuration by entering the `show chassis`, `show interfaces`, `show routing-options`, `show protocols`, `show vlans`, `show policy-options`, and `show switch-options` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE3# show
interfaces {
  xe-0/0/0 {
    enable;
    unit 0 {
      description "Connected to PE1";
      family inet {
        address 192.0.2.2/24;
      }
    }
  }
  xe-0/0/1 {
    enable;
    unit 0 {
      description "Connected to PE2";
      family inet {
        address 198.51.100.2/24
      }
    }
  }
}
```



```

xe-0/0/0:1 {
    enable;
    unit 0 {
        family ethernet-switching {
            interface-mode trunk;
            vlan {
                members [ VLAN1 VLAN2 VLAN3 VLAN4 VLAN5 ];
            }
        }
    }
}
xe-0/0/1:1 enable;
unit 0 {
    family ethernet-switching {
        interface-mode trunk;
        vlan {
            members [ VLAN1 VLAN2 VLAN3 VLAN4 VLAN5 ];
        }
    }
}
}
irb {
    unit 1 {
        family inet {
            address 10.1.1.5/24 {
                virtual-gateway-address 10.1.1.10;
            }
        }
    }
    unit 2 {
        family inet {
            address 10.2.1.5/24 {
                virtual-gateway-address 10.2.1.10;
            }
        }
    }
    unit 3 {
        family inet {
            address 10.3.1.5/24 {
                virtual-gateway-address 10.3.1.10;
            }
        }
    }
}

```

```

    unit 4 {
        family inet {
            address 10.4.1.5/24 {
                virtual-gateway-address 10.4.1.10;
            }
        }
    }
    unit 5 {
        family inet {
            address 10.5.1.5/24 {
                virtual-gateway-address 10.5.1.10;
            }
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 172.16.1.1/32;
        }
    }
}
routing-options {
    router-id 172.16.1.1;
    autonomous-system 65536
    forwarding-table {
        export evpn-pplb;
    }
}
protocols {
    igmp {
        interface irb.1;
        interface irb.2;
        interface irb.3;
        interface irb.4;
        interface irb.5;
    }
    bgp {
        group INT {
            type internal;
            local-address 172.16.1.1;
            family evpn {

```

```

        signaling;
    }
    local-as 65546;
    neighbor 192.168.1.1;
    neighbor 192.168.2.1;
}
}
ospf {
    area 0.0.0.0 {
        interface all ;
        interface fxp0.0;
        disable;
    }
}
}
pim {
    rp {
        local {
            address 172.16.1.1;
        }
    }
    interface irb.1 {
        distributed-dr;
    }
    interface irb.2 {
        distributed-dr;
    }
    interface irb.3 {
        distributed-dr;
    }
    interface irb.4 {
        distributed-dr;
    }
    interface irb.5 {
        distributed-dr;
    }
}
evpn {
    encapsulation vxlan;
    multicast-mode ingress-replication;
    extended-vni-list 1-5;
}
igmp-snooping {

```

```

    vlan VLAN1 {
        l2-querier {
            source-address 10.1.1.5;
        }
        proxy;
    }
    vlan VLAN2 {
        l2-querier {
            source-address 10.2.1.5;
        }
        proxy;
    }
    vlan VLAN3 {
        l2-querier {
            source-address 10.3.1.5;
        }
        proxy;
    }
    vlan VLAN4 {
        l2-querier {
            source-address 10.4.1.5;
        }
        proxy;
    }
    vlan VLAN5 {
        l2-querier {
            source-address 10.5.1.5;
        }
        proxy;
    }
}

policy-options {
    policy-statement evpn-pplb {
        from protocol evpn;
        then {
            load-balance per-packet;
        }
    }
}

switch-options {
    vtep-source-interface lo0.0;
    route-distinguisher 172.16.1.1:1;
}

```

```
    vrf-target target:1:1;
}
vlans {
    VLAN1 {
        vlan-id 1;
        l3-interface irb.1;
        vxlan {
            vni 1;
        }
    }
    VLAN2 {
        vlan-id 2;
        l3-interface irb.2;
        vxlan {
            vni 2;
        }
    }
    VLAN3 {
        vlan-id 3;
        l3-interface irb.3;
        vxlan {
            vni 3;
        }
    }
    VLAN4 {
        vlan-id 4;
        l3-interface irb.4;
        vxlan {
            vni 4;
        }
    }
    VLAN5 {
        vlan-id 5;
        l3-interface irb.5;
        vxlan {
            vni 5;
        }
    }
}
```

Verification

IN THIS SECTION

- [Verifying IGMP Messages are Synced | 739](#)
- [Verifying Source Addresses Are Learned and Multicast Traffic Is Being Forwarded | 741](#)

Confirm that the configuration is working properly.

Verifying IGMP Messages are Synced

Purpose

Verify on each PE that IGMP join and leave messages are synced.

Action

From operational mode, run the `show evpn instance extensive` command.

```
user@PE1> show evpn instance extensive
Instance: default-switch
Route Distinguisher: 192.168.1.1:1
Encapsulation type: VXLAN
MAC database status
MAC advertisements:          Local  Remote
MAC+IP advertisements:      25      40
Default gateway MAC advertisements: 10      15
                             10      10
Number of local interfaces: 3 (3 up)
Interface name  ESI                               Mode        Status    AC-Role
ae0.0           00:11:11:11:11:11:11:11:11:11:11:11  all-active  Up        Root
ae1.0           00:22:22:22:22:22:22:22:22:22:22:22  all-active  Up        Root
xe-0/0/1:1.0    00:00:00:00:00:00:00:00:00:00:00:00  single-homed Up        Root
Number of IRB interfaces: 5 (5 up)
Interface name  VLAN  VNI  Status  L3 context
irb.1           1     Up   master
irb.2           2     Up   master
irb.3           3     Up   master
irb.4           4     Up   master
```

```

    irb.5          5      Up      master
Number of bridge domains: 5
  VLAN  Domain ID  Intfs / up  IRB intf  Mode          MAC sync  IM route label  SG
sync  IM core nexthop
    1      1          3    3    irb.1      Extended      Enabled    1
Enabled  2097159
    2      2          3    3    irb.2      Extended      Enabled    2
Enabled  2097161
    3      3          3    3    irb.3      Extended      Enabled    3
Enabled  2097162
    4      4          3    3    irb.4      Extended      Enabled    4
Enabled  2097163
    5      5          3    3    irb.5      Extended      Enabled    5
Enabled  2097164
Number of neighbors: 2
  Address          MAC      MAC+IP      AD      IM      ES Leaf-label
192.168.2.1        25       10         9       5       0
172.16.1.1         15       10         1       5       0
Number of ethernet segments: 7
ESI: 00:11:11:11:11:11:11:11:11:11
  Status: Resolved by IFL ae0.0
  Local interface: ae0.0, Status: Up/Forwarding
  Number of remote PEs connected: 1
    Remote PE      MAC label  Aliasing label  Mode
    192.168.2.1    5          0              all-active
  DF Election Algorithm: MOD based
  Designated forwarder: 192.168.2.1
  Backup forwarder: 192.168.1.1
  Last designated forwarder update: Jul 13 01:22:45
ESI: 00:22:22:22:22:22:22:22:22:22
  Status: Resolved by IFL ae1.0
  Local interface: ae1.0, Status: Up/Forwarding
  Number of remote PEs connected: 1
    Remote PE      MAC label  Aliasing label  Mode
    192.168.2.1    5          0              all-active
  DF Election Algorithm: MOD based
  Designated forwarder: 192.168.2.1
  Backup forwarder: 192.168.1.1
  Last designated forwarder update: Jul 13 21:02:47
ESI: 05:00:00:00:64:00:00:00:01:00
  Local interface: irb.1, Status: Up/Forwarding
  Number of remote PEs connected: 2
    Remote PE      MAC label  Aliasing label  Mode

```

```

192.168.2.1      1      0      all-active
172.16.1.1      1      0      single-homed
ESI: 05:00:00:00:64:00:00:00:02:00
Local interface: irb.2, Status: Up/Forwarding
Number of remote PEs connected: 2
  Remote PE      MAC label  Aliasing label  Mode
  192.168.2.1    2          0          all-active
  172.16.1.1     2          0          single-homed
ESI: 05:00:00:00:64:00:00:00:03:00
Local interface: irb.3, Status: Up/Forwarding
Number of remote PEs connected: 2
  Remote PE      MAC label  Aliasing label  Mode
  192.168.2.1    3          0          all-active
  172.16.1.1     3          0          single-homed
ESI: 05:00:00:00:64:00:00:00:04:00
Local interface: irb.4, Status: Up/Forwarding
Number of remote PEs connected: 2
  Remote PE      MAC label  Aliasing label  Mode
  192.168.2.1    4          0          all-active
  172.16.1.1     4          0          single-homed
ESI: 05:00:00:00:64:00:00:00:05:00
Local interface: irb.5, Status: Up/Forwarding
Number of remote PEs connected: 2
  Remote PE      MAC label  Aliasing label  Mode
  172.16.1.1     5          0          all-active
  192.168.2.1    5          0          all-active
Router-ID: 192.168.1.1

```

Meaning

The SG Sync is Enabled and the IM Core next-hop field displays a valid route.

Verifying Source Addresses Are Learned and Multicast Traffic Is Being Forwarded

Purpose

Verify on each PE that multicast receivers have learned the source interface for the VXLAN tunnel.

Action

From operational mode, enter the `show evpn igmp-snooping database extensive l2-domain-id 1` command and the `show igmp snooping evpn database vlan VLAN1` commands.

These commands displays output for VLAN1. You can use them to display output for each configured VLAN

From operational mode, enter the `show evpn multicast-snooping next-hops` to verify that downstream interface has been learned.

```
user@PE1> show evpn igmp-snooping database extensive
l2-domain-id 1
Instance: default-switch
  VN Identifier: 1
    Group IP: 225.1.1.1
      Access OIF Count: 2
        Interface      ESI      Local  Remote
        ae1.0          00:22:22:22:22:22:22:22:22 1      0
        ae0.0          00:11:11:11:11:11:11:11:11 0      1
      Remote OIF Count: 2, Core NH: 2097159
        Interface      Nbr
        vtep.32770      192.168.2.1
        vtep.32769      172.16.1.1
    Group IP: 225.1.1.2
      Access OIF Count: 2
        Interface      ESI      Local  Remote
        ae1.0          00:22:22:22:22:22:22:22:22 1      0
        ae0.0          00:11:11:11:11:11:11:11:11 0      1
      Remote OIF Count: 2, Core NH: 2097159
        Interface      Nbr
        vtep.32770      192.168.2.1
        vtep.32769      172.16.1.1
    Group IP: 225.1.1.3
      Access OIF Count: 2
        Interface      ESI      Local  Remote
        ae0.0          00:11:11:11:11:11:11:11:11 1      0
        ae1.0          00:22:22:22:22:22:22:22:22 1      0
      Remote OIF Count: 2, Core NH: 2097159
        Interface      Nbr
        vtep.32770      192.168.2.1
        vtep.32769      172.16.1.1
    Group IP: 225.1.1.4
```

Access OIF Count: 2

Interface	ESI	Local	Remote
ae0.0	00:11:11:11:11:11:11:11:11	1	0
ae1.0	00:22:22:22:22:22:22:22:22	0	1

Remote OIF Count: 2, Core NH: 2097159

Interface	Nbr
vtep.32770	192.168.2.1
vtep.32769	172.16.1.1

Group IP: 225.1.1.5

Access OIF Count: 2

Interface	ESI	Local	Remote
ae0.0	00:11:11:11:11:11:11:11:11	1	0
ae1.0	00:22:22:22:22:22:22:22:22	1	0

Remote OIF Count: 2, Core NH: 2097159

Interface	Nbr
vtep.32770	192.168.2.1
vtep.32769	172.16.1.1

Group IP: 225.1.1.6

Access OIF Count: 2

Interface	ESI	Local	Remote
ae0.0	00:11:11:11:11:11:11:11:11	0	1
ae1.0	00:22:22:22:22:22:22:22:22	1	0

Remote OIF Count: 2, Core NH: 2097159

Interface	Nbr
vtep.32770	192.168.2.1
vtep.32769	172.16.1.1

Group IP: 225.1.1.7

Access OIF Count: 2

Interface	ESI	Local	Remote
ae0.0	00:11:11:11:11:11:11:11:11	0	1
ae1.0	00:22:22:22:22:22:22:22:22	1	0

Remote OIF Count: 2, Core NH: 2097159

Interface	Nbr
vtep.32770	192.168.2.1
vtep.32769	172.16.1.1

Group IP: 225.1.1.8

Access OIF Count: 2

Interface	ESI	Local	Remote
ae0.0	00:11:11:11:11:11:11:11:11	1	0
ae1.0	00:22:22:22:22:22:22:22:22	0	1

Remote OIF Count: 2, Core NH: 2097159

Interface	Nbr
vtep.32770	192.168.2.1

```

vtep.32769 172.16.1.1
Group IP: 225.1.1.9
Access OIF Count: 2
  Interface      ESI      Local  Remote
  ae0.0          00:11:11:11:11:11:11:11:11 1      0
  ae1.0          00:22:22:22:22:22:22:22:22 0      1
Remote OIF Count: 2, Core NH: 2097159
  Interface      Nbr
  vtep.32770     192.168.2.1
  vtep.32769     172.16.1.1
Group IP: 225.1.1.10
Access OIF Count: 2
  Interface      ESI      Local  Remote
  ae0.0          00:11:11:11:11:11:11:11:11 0      1
  ae1.0          00:22:22:22:22:22:22:22:22 0      1
Remote OIF Count: 2, Core NH: 2097159
  Interface      Nbr
  vtep.32770     192.168.2.1
  vtep.32769     172.16.1.1

```

```

user@PE1> show igmp snooping evpn database vlan
VLAN1
Instance: default-switch
Bridge-Domain: VLAN1, VN Identifier: 1
Group IP: 225.1.1.1
Core NH: 2097159
Access OIF Count: 3
  Interface      Local  Remote
  ae0.0          0      1
  xe-0/0/1:1.0   1      0
  ae1.0          1      0
Group IP: 225.1.1.2
Core NH: 2097159
Access OIF Count: 3
  Interface      Local  Remote
  ae0.0          0      1
  xe-0/0/1:1.0   1      0
  ae1.0          1      0
Group IP: 225.1.1.3
Core NH: 2097159
Access OIF Count: 3

```

Interface	Local	Remote
xe-0/0/1:1.0	1	0
ae1.0	1	0
ae0.0	1	0

Group IP: 225.1.1.4

Core NH: 2097159

Access OIF Count: 3

Interface	Local	Remote
ae1.0	0	1
xe-0/0/1:1.0	1	0
ae0.0	1	0

Group IP: 225.1.1.5

Core NH: 2097159

Access OIF Count: 3

Interface	Local	Remote
ae1.0	1	0
xe-0/0/1:1.0	1	0
ae0.0	1	0

Group IP: 225.1.1.6

Core NH: 2097159

Access OIF Count: 3

Interface	Local	Remote
ae0.0	0	1
ae1.0	1	0
xe-0/0/1:1.0	1	0

Group IP: 225.1.1.7

Core NH: 2097159

Access OIF Count: 3

Interface	Local	Remote
ae0.0	0	1
ae1.0	1	0
xe-0/0/1:1.0	1	0

Group IP: 225.1.1.8

Core NH: 2097159

Access OIF Count: 3

Interface	Local	Remote
ae1.0	0	1
xe-0/0/1:1.0	1	0
ae0.0	1	0

Group IP: 225.1.1.9

Core NH: 2097159

Access OIF Count: 3

Interface	Local	Remote
-----------	-------	--------

```

        ae1.0          0      1
        xe-0/0/1:1.0   1      0
        ae0.0          1      0
Group IP: 225.1.1.10
Core NH: 2097159
Access OIF Count: 3
      Interface  Local  Remote
        ae1.0      0      1
        ae0.0      0      1
        xe-0/0/1:1.0  1      0

```

```

user@PE1> show evpn multicast-snooping next-hops
Family: INET
ID          Refcount KRefCount Downstream interface Addr
2097159      3         1 vtep.32769
                vtep.32770
2097161      3         1 vtep.32769
                vtep.32770
2097162      3         1 vtep.32769
                vtep.32770
2097163      3         1 vtep.32769
                vtep.32770
2097164      3         1 vtep.32769
                vtep.32770

```

RELATED DOCUMENTATION

[Overview of Multicast Forwarding with IGMP Snooping or MLD Snooping in an EVPN-VXLAN Environment | 675](#)

[igmp-snooping](#)

Overview of Selective Multicast Forwarding

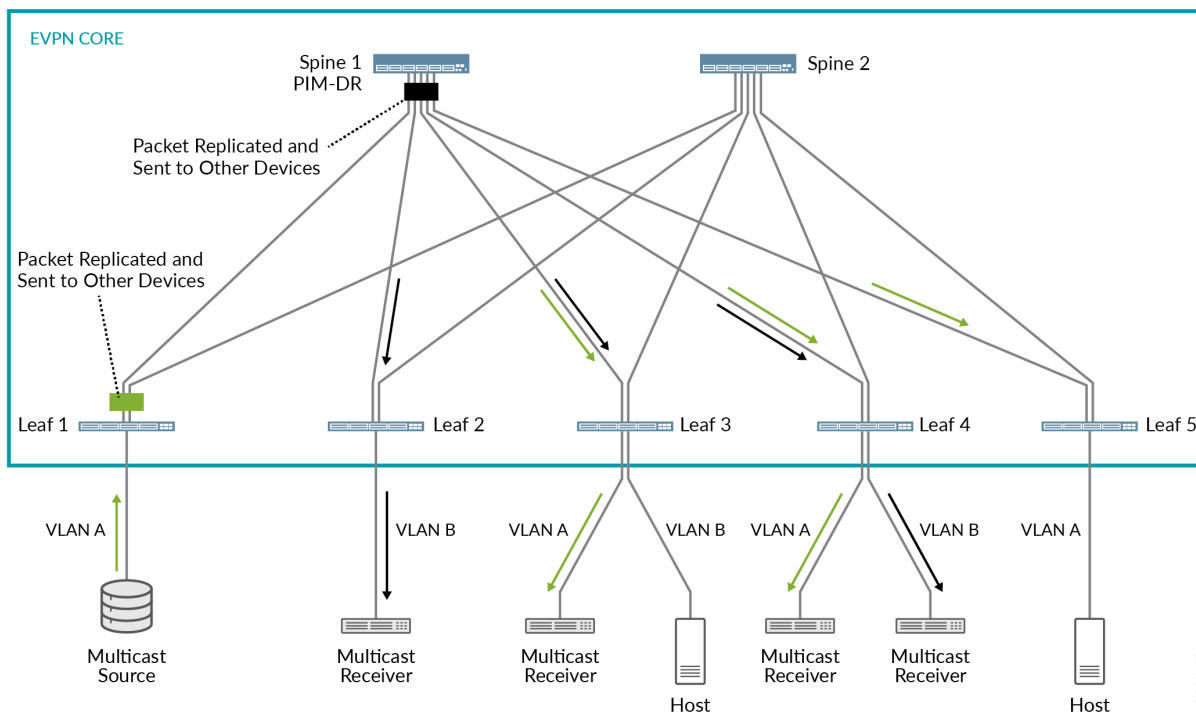
IN THIS SECTION

- [Benefits of Selective Multicast Forwarding | 750](#)
- [Limitations of Selective Multicast Forwarding | 750](#)

Prior to Junos OS Release 18.4R1, PE devices with IGMP snooping enabled only constrained the local multicast traffic going to its access interface. For intra-VLAN multicast traffic in a leaf-spine topology, when a leaf device receives multicast traffic from another device or from a multicast sender on one of its access interface, it replicates and forwards the multicast traffic to its own interfaces with interested receivers. The leaf device will also replicate and send the multicast traffic to all the leaf and spine devices in the EVPN core. For inter-VLAN multicast traffic, the spine device in a centrally routed EVPN-VXLAN network would route the multicast traffic between the VLANs through the IRB interface.

Figure 1 shows how inclusive multicast traffic flows in a centrally-routed EVPN network.

Figure 69: Inclusive Multicast Forwarding in a Centrally-Routed EVPN-VXLAN Network



When the multicast source sends a packet from VLAN A, the following flows occur:

- **Flow 1: Intra-VLAN traffic**—Based on the ingress replication mechanism, Leaf 1 replicates and switches the packet to all spine and other leaf devices. Leafs 3 and 4, on which VLAN A is configured, receive and forward the packet to the connected multicast receivers. Leaf 5, which is also on VLAN A network still receives the multicast packet even though there are no receivers.
- **Flow 2: Inter-VLAN traffic**—Upon receipt of the packet from Leaf 1 as described in flow 1, Spine 1, which is the PIM DR, takes the following action:
 - Routes the packet over the IRB interface associated with VLAN B.
 - Based on the ingress replication mechanism, Spine 1 replicates and forwards the packet to the other spine and leafs.

Leafs 2 and 4, on which VLAN B is configured, receive and forward the packet to the connected multicast receivers. Leaf 3 receives the multicast packet, but does not forward the packet since it does not have a receiver.

Starting in Junos OS Release 18.4R1, devices with IGMP snooping enabled use selective multicast forwarding by default in a centrally routed EVPN-VXLAN network to replicate and forward multicast traffic.

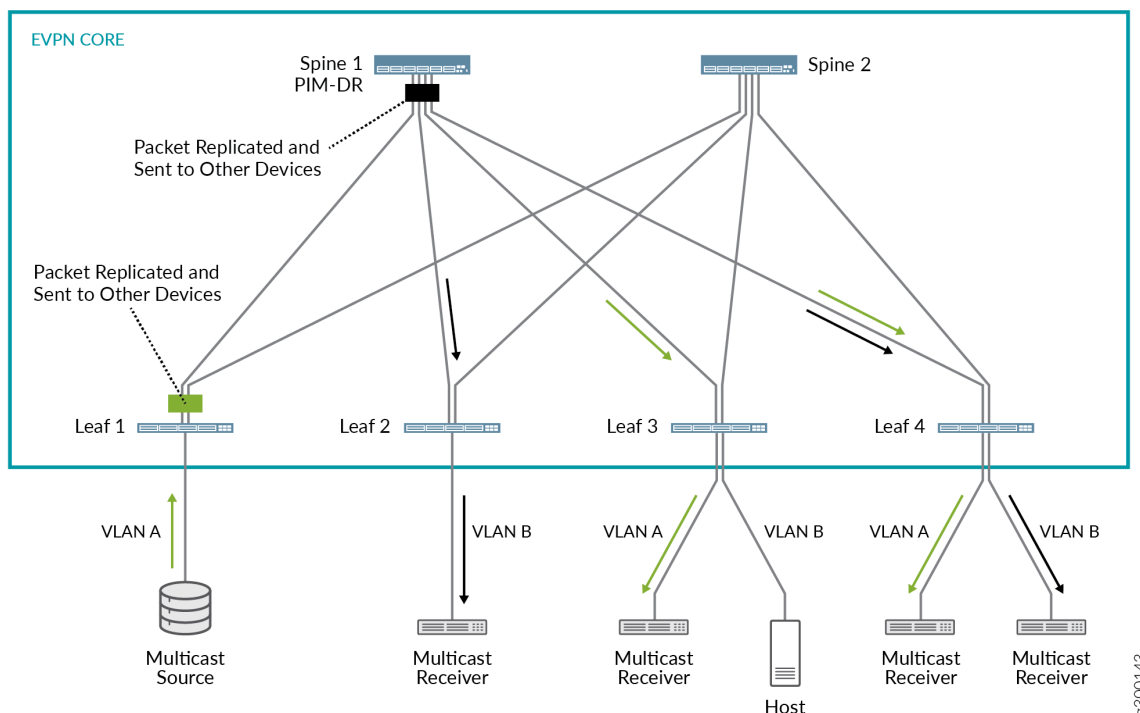
Similarly, starting in Junos OS Release 20.4R1, QFX Series switches with MLD snooping enabled in a centrally routed EVPN-VXLAN network also enable selective multicast forwarding by default.

This is how selective multicast forwarding works with IGMP or MLD snooping on the leaf devices:

- IGMP or MLD snooping allows leaf devices to send multicast traffic only to the access interface with an interested receiver.
- When you enable IGMP or MLD snooping, a leaf device selectively sends multicast traffic to only the other leaf devices across the EVPN core that have expressed an interest in that multicast group.
- With selective multicast forwarding, leaf devices also always send multicast traffic to a spine device so that the spine device can route inter-VLAN multicast traffic through its IRB interfaces.

Figure 2 shows how selective multicast traffic flows in a centrally-routed EVPN network.

Figure 70: Selective Multicast Forwarding in a Centrally-Routed EVPN-VXLAN Network



When the multicast source sends a packet from VLAN A, the following flows occur:

- **Flow 1: Intra-VLAN traffic**—Based on the ingress replication mechanism, Leaf 1 replicates and switches the packet to all spine and other leaf devices with interested receivers. In this case, leafs 3 and 4, on which VLAN A is configured and forward the packet to the connected interested multicast receivers.
- **Flow 2: Inter-VLAN traffic**—Upon receipt of the packet from Leaf 1 as described in flow 1, Spine 1, which is the PIM DR, takes the following action:
 - Routes the packet over the IRB interface associated with VLAN B.
 - Based on the ingress replication mechanism, Spine 1 replicates and forwards the packet to the other spine and interested leaf devices.

Leafs 2 and 4, on which VLAN B is configured, receive and forward the packet to the connected multicast receivers. Leaf 3 does not have receivers and does not get a multicast packet.

Benefits of Selective Multicast Forwarding

Selective multicast forwarding provides greater network efficiency and reduces traffic in the EVPN network. Selective multicast forwarding conserves bandwidth usage in the core and reduces the load on egress PE devices that do not have listeners. The benefits of selective multicast forwarding increases when there are listeners in multiple VLANs.

Limitations of Selective Multicast Forwarding

- Supported in a centrally-routed leaf-spine topology.
- Support for EVPN-VXLAN encapsulation only.
- Support for EVPN-ELAN services only.
- Supported with IGMPv2, MLDv1, and MLDv2 traffic on particular QFX Series platforms and Junos OS releases.

RELATED DOCUMENTATION

[Overview of Multicast Forwarding with IGMP Snooping or MLD Snooping in an EVPN-VXLAN Environment | 675](#)

[igmp-snooping](#)

[mld-snooping](#)

Configuring the number of SMET Nexthops

Junos OS uses EVPN route type 6, selective multicast Ethernet (SMET) route message to support the following:

- External multicast sender (IGMP proxy)—EVPN route type 6 messages are used within the EVPN network. The ingress PE device translates the IGMP message to an EVPN route type 6 message and the egress PE device translates the EVPN route type 6 message back to an IGMP message.
- Inter-VLAN multicast—EVPN type route 6 messages are used to create a PIM states on a PE device with an IRB interface. This allows multicast traffic to be sent across VLANs.
- Selective multicast forwarding—The EVPN route type 6 messages are used to distribute the routing information indicating a PE device's interest for a multicast group.

The information in the EVPN route type 6 message are used to build a list of SMET next hops, which can be used to selectively replicate and forward multicast packets. SMET next hop is a list of outgoing interfaces (OIFs) identifying the interested PEs. Multicast groups are mapped to SMET next hops. If multicast groups that have the same set of interested PEs, they can share a SMET next hop. The number of SMET nexthops defaults to 10,000 and can be increased by configuring the `smet-nexthop-limit` option. When a device reaches the SMET nexthop limit, the device will start using inclusive multicast forwarding for multicast traffic.

Devices in the network that do not support snooping or cannot send EVPN route type 6 messages are always included in the SMET next hop. This ensures those devices that do not support EVPN type 6 messages will be able to receive multicast traffic.

To configure the number of SMET nexthops, you can use the following statement:

```
User@PE1# set forwarding-options multicast-replication evpn smet-nexthop-limit nexthop range.
```

In some cases, you may want to bypass selective multicast forwarding and send multicast traffic to all devices. If you wish to send multicast traffic to a group, you can list the multicast group address with the following statement

```
User@PE1# set multicast-snooping-options flood-groups [ip-addresses]
```

NOTE: OSPF messages, which use multicast addresses for communication, are automatically included in the multicast snooping forwarding table.

RELATED DOCUMENTATION

[multicast-replication](#)

[multicast-snooping-options](#)

[show evpn igmp-snooping proxy](#) | 1831

[show evpn instance](#) | 1834

[show multicast snooping route](#)

[show route table](#) | 2003

Assisted Replication Multicast Optimization in EVPN Networks

SUMMARY

Assisted replication helps to optimize multicast traffic flow in EVPN networks by offloading traffic replication to devices that can more efficiently handle the task.

IN THIS SECTION

- [Assisted Replication in EVPN Networks | 752](#)
- [Configure Assisted Replication | 766](#)
- [Verify Assisted Replication Setup and Operation | 768](#)

Assisted Replication in EVPN Networks

IN THIS SECTION

- [Benefits of Assisted Replication | 753](#)
- [AR Device Roles | 753](#)
- [How AR Works | 754](#)
- [Multicast Forwarding Use Cases in an EVPN Network with AR Devices | 760](#)

Assisted replication (AR) is a Layer 2 multicast traffic optimization feature supported in EVPN networks. With AR enabled, an ingress device replicates a multicast stream to another device in the EVPN network that can more efficiently handle replicating and forwarding the stream to the other devices in the network. For backward compatibility, ingress devices that don't support AR operate transparently with other devices that have AR enabled, and use ingress replication to distribute the traffic themselves to other devices in the EVPN network.

AR provides an overlay multicast optimization Layer 2 solution, and does not require PIM to be enabled in the underlay.

NOTE: This documentation describes AR functions in terms of multicast traffic replication and forwarding, but applies to any broadcast, unknown unicast, and multicast (BUM) traffic in general.

Benefits of Assisted Replication

- In an EVPN network with a significant volume of broadcast, unknown unicast, and multicast (BUM) traffic, helps to optimize BUM traffic flow by passing replication and forwarding tasks to other devices that have more capacity to better handle the load.
- In an EVPN-VXLAN environment, reduces the number of hardware next hops to multiple remote virtual tunnel endpoints (VTEPs) over traditional multicast ingress replication.
- Can operate with other multicast optimizations such as IGMP snooping, MLD snooping, and selective multicast Ethernet tag (SMET) forwarding.
- Operates transparently with devices that do not support AR for backward compatibility in existing EVPN networks.

AR Device Roles

Starting in Junos OS Releases 18.4R2 and 19.4R1, you can enable AR on QFX Series switches that support EVPN-VXLAN, which includes the QFX10000 line of switches and QFX5110 and QFX5120 switches.

To enable AR, you configure devices in the EVPN network into AR replicator and AR leaf roles. For backward compatibility, your EVPN network can also include devices that don't support AR, which operate in a regular network virtualization edge (NVE) device role.

[Table 33 on page 753](#) summarizes these roles:

Table 33: AR Device Roles

Role	Description
AR leaf device	Device in an EVPN network that offloads multicast ingress replication tasks to another device in the EVPN network to handle the replication and forwarding load.
AR replicator device	Device in an EVPN network that helps perform multicast ingress replication and forwarding for traffic received from AR leaf devices on an AR overlay tunnel to other ingress replication overlay tunnels.
Regular NVE device	Device that does not support AR or is not configured into an AR role in an EVPN network, and replicates and forwards multicast traffic using the usual EVPN network ingress replication.

If you enable other supported multicast optimization features in your EVPN network such as IGMP snooping, MLD snooping, and SMET forwarding, AR replicator and AR leaf devices forward traffic in the EVPN core or on the access side only toward interested receivers.

NOTE: With QFX Series AR devices in EVPN-VXLAN networks, AR replicator and AR leaf devices must operate in extended AR mode. In this mode, AR leaf devices that have multihomed Ethernet segments share some of the replication load with the AR replicator devices, which enables them to use split horizon and local bias rules to prevent traffic loops and duplicate forwarding. See ["Extended AR Mode for Multihomed Ethernet Segments" on page 758](#). For details on how devices in an EVPN-VXLAN environment forward traffic to multihomed Ethernet segments using local bias and split horizon filtering, see ["EVPN-over-VXLAN Supported Functionality" on page 408](#).

How AR Works

In general, devices in the EVPN network use ingress replication to distribute BUM traffic. The ingress device (the device where the source traffic enters the network) replicates and sends the traffic on overlay tunnels to all other devices in the network. For EVPN topologies with EVPN multihoming (ESI-LAGs), network devices employ local bias or designated forwarder (DF) rules to avoid duplicating forwarded traffic to receivers on multihomed Ethernet segments. With multicast optimizations like IGMP or MLD snooping and SMET forwarding enabled, forwarding devices avoid sending unnecessary traffic by replicating the traffic only to other devices that have active listeners.

AR operates within the existing EVPN network overlay and multicast forwarding mechanisms, but defines special AR overlay tunnels over which AR leaf devices pass traffic from multicast sources to AR replicator devices. AR replicator devices treat incoming source traffic on AR overlay tunnels as requests to replicate the traffic toward other devices in the EVPN network on behalf of the sending AR leaf device.

AR basically works like this:

- Each AR replicator device advertises its replicator capabilities and AR IP address to the EVPN network using EVPN Type 3 (inclusive multicast Ethernet tag [IMET]) routes. You configure a secondary IP address on the lo0 loopback interface as an AR IP address when you assign the replicator role to the device. AR leaf devices use that IP address for the AR overlay tunnel.
- An AR leaf device receives these advertisements to learn about available AR replicator devices and their capabilities.
- AR leaf devices advertise EVPN Type 3 routes for ingress replication that include the AR leaf device ingress replication tunnel IP address.

- The AR leaf device forwards multicast source traffic to a selected AR replicator on its AR overlay tunnel.

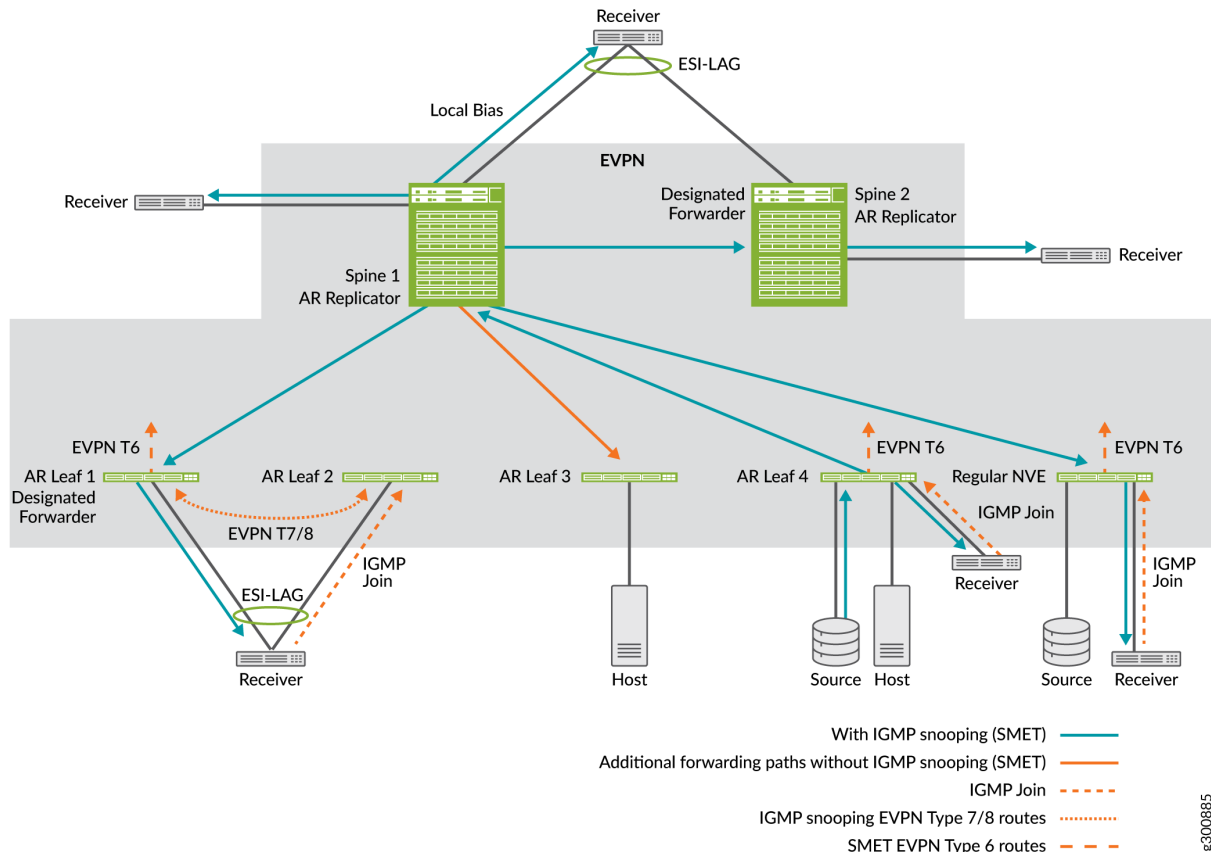
When the network has multiple AR replicator devices, AR leaf devices automatically load-balance among them. (See ["AR Leaf Device Load Balancing With Multiple Replicators" on page 759.](#))

- The AR replicator device receives multicast source traffic from an AR leaf device on an AR overlay tunnel and replicates it to the other devices in the network using the established ingress replication overlay tunnels. It also forwards the traffic toward local receivers and external gateways, including those on multihomed Ethernet segments, using the same local bias or DF rules it would use for traffic received from directly connected sources or regular NVE devices.

NOTE: When AR operates in extended AR mode, an AR leaf device with a multihomed source handles the replication to its EVPN multihoming peers, and the AR replicator device receiving source traffic from that AR leaf device skips forwarding to the multihomed peers. (See ["Extended AR Mode for Multihomed Ethernet Segments" on page 758](#); [Figure 75 on page 765](#) illustrates this use case.)

Figure 71 on page 756 shows an example of multicast traffic flow in an EVPN network with two AR replicators, four AR leaf devices, and a regular NVE device.

Figure 71: Assisted Replication Traffic Flow



In Figure 71 on page 756, AR Leaf 4 forwards multicast source traffic on an AR overlay tunnel to Spine 1, one of the available AR replicators. Spine 1 receives the traffic on the AR tunnel and replicates it on the usual ingress replication overlay tunnels to all other devices in the EVPN network, including AR leaf devices, other AR replicators, and regular NVE devices. In this example, Spine 1 also forwards the traffic to its local receivers, and, applying local bias rules, forwards the traffic to its multihomed receiver regardless of whether or not it is the DF for that multihomed Ethernet segment. In addition, according to split horizon rules, AR replicator Spine 1 does not forward the traffic to AR Leaf 4. Instead, AR Leaf 4, the ingress device, does local bias forwarding to its attached receiver.

When you enable IGMP snooping or MLD snooping, SMET forwarding is also enabled by default, and the AR replicator skips sending the traffic on ingress replication overlay tunnels to any devices that don't have active listeners.

Figure 71 on page 756 shows a simplified view of the control messages and traffic flow with IGMP snooping enabled:

- Receivers attached to AR Leaf 1 (multihomed to AR Leaf 2), AR Leaf 4, and Regular NVE send IGMP join messages to express interest in receiving the multicast stream.
- For IGMP snooping to work with EVPN multihoming and avoid duplicate traffic, multihomed peers AR Leaf 1 and AR Leaf 2 synchronize the IGMP state using EVPN Type 7 and 8 (Join Sync and Leave Sync) routes. (Spine 1 and Spine 2 do the same for their multihomed receiver, although the figure doesn't show that specifically.)
- The EVPN devices hosting receivers (and only the DF for multihomed receivers) advertise EVPN Type 6 routes in the EVPN core (see ["Overview of Selective Multicast Forwarding" on page 747](#)), so forwarding devices only send the traffic to the other EVPN devices with interested receivers.

For IPv6 multicast traffic with MLD and MLD snooping enabled, you'll see the same behavior as illustrated for IGMP snooping.

Regular NVE devices and other AR replicator devices don't send source traffic on AR overlay tunnels to an AR replicator device, and AR replicators don't perform AR tasks when receiving multicast source traffic on regular ingress replication overlay tunnels. AR replicators forward traffic they did not receive on AR tunnels the way they normally would using EVPN network ingress replication.

Similarly, if an AR leaf device doesn't see any available AR replicators, the AR leaf device defaults to using the usual ingress replication forwarding behavior (the same as a regular NVE device). In that case, AR show commands display the AR operational mode as No replicators/Ingress Replication. See the [show evpn multicast-snooping assisted-replication replicators](#) command for more details.

AR Route Advertisements

Devices in an EVPN network advertise EVPN Type 3 (IMET) routes for regular ingress replication and assisted replication.

AR replicator devices advertise EVPN Type 3 routes for regular ingress replication that include:

- The AR replicator device ingress replication tunnel IP address
- Tunnel type—IR
- AR device role—AR-REPLICATOR

AR replicators also advertise EVPN Type 3 routes for the special AR overlay tunnels that include:

- An AR IP address you configure on a loopback interface on the AR replicator device (see the ["replicator" on page 1649](#) configuration statement)
- Tunnel type—AR

- AR device role—AR-REPLICATOR
- EVPN multicast flags extended community with the Extended-MH-AR flag when operating in extended AR mode (see ["Extended AR Mode for Multihomed Ethernet Segments" on page 758](#))

AR leaf devices advertise EVPN Type 3 routes for regular ingress replication that include:

- The AR leaf device ingress replication tunnel IP address
- Tunnel type—IR
- AR device role—AR-LEAF

Regular NVE devices, which don't support AR or are not configured as an AR leaf device, ignore AR route advertisements and forward multicast traffic using the usual EVPN network ingress replication rules.

Extended AR Mode for Multihomed Ethernet Segments

EVPN networks employ split horizon and local bias rules to enable multicast optimizations when the architecture includes multihomed Ethernet segments (see ["EVPN-over-VXLAN Supported Functionality" on page 408](#)). These methods include information about the ingress AR leaf device in forwarded packets to ensure that the traffic isn't unnecessarily forwarded or looped back to the source by way of the ingress device's multihomed peers.

Some devices serving in the AR replicator role can't retain the source IP address or Ethernet segment identifier (ESI) label on behalf of the ingress AR leaf device when forwarding the traffic to other overlay tunnels. AR replicator devices with this limitation operate in *extended AR mode*, where the AR replicator forwards traffic toward other devices in the EVPN network with the AR replicator's ingress replication IP address as the source IP address.

NOTE: QFX Series AR devices in an EVPN-VXLAN environment use extended AR mode. We currently only support AR in VXLAN overlay architectures, so the only supported AR mode is extended AR mode.

AR replicators include the EVPN multicast flags extended community in the EVPN Type 3 AR route advertisement. The Extended-MH-AR flag indicates the AR operating mode.

When AR devices operate in extended AR mode:

- Besides forwarding the traffic to the AR replicator device, an AR leaf device with multihomed sources *also* handles replicating the traffic to its multihoming peers.
- An AR replicator device receiving source traffic from an AR leaf device with multihomed sources skips forwarding to the AR leaf device's multihoming peers.

See ["Source Behind an AR Leaf Device \(Multihomed Ethernet Segment\) with Extended AR Mode" on page 765](#), which shows the traffic flow in extended AR mode for an EVPN network with a multihomed source.

Extended AR mode behavior applies only when AR leaf devices have multihomed Ethernet segments with other AR leaf devices, not with AR replicators from which the AR leaf requests replication assistance. When AR replicator and AR leaf devices share multihomed Ethernet segments in environments that require extended AR mode, AR can't work correctly, so ingress AR leaf devices don't use the AR tunnels and default to using only ingress replication. In this case, AR show commands display the AR operational mode as Misconfiguration/Ingress Replication.

AR replicators that can retain the source IP address or ESI label of the ingress AR leaf device operate in *regular AR mode*.

See the `show evpn multicast-snooping assisted-replication replicators` command for more information on AR operational modes.

AR Leaf Device Load Balancing With Multiple Replicators

When the EVPN network has more than one advertised AR replicator device, the AR leaf devices automatically load-balance among the available AR replicator devices.

For QFX Series switches in an EVPN-VXLAN network:

- AR leaf devices in the QFX5000 line of switches designate a particular AR replicator device for a VLAN or VXLAN network identifier (VNI) to load-balance among the available AR replicators.
- AR leaf devices in the QFX10000 line of switches actively load-balance among the available AR replicators based on traffic flow levels within a VNI.

AR Limitations With IGMP or MLD Snooping

AR replicator devices with IGMP or MLD snooping enabled will silently drop multicast traffic toward AR Leaf devices that don't support IGMP or MLD snooping in an EVPN-VXLAN environment.

If you want to include IGMP or MLD snooping multicast traffic optimization with AR, you can work around this limitation by disabling AR on the EVPN devices that don't support IGMP or MLD snooping so they don't function as AR leaf devices. Those devices then behave like regular network virtualization edge (NVE) devices and can receive the multicast traffic by way of the usual EVPN network ingress replication, although without the benefits of AR and IGMP or MLD snooping optimizations. You can configure devices that do support IGMP or MLD snooping in this environment as AR leaf devices with IGMP or MLD snooping.

See [Table 33 on page 753](#) for more about the behavior differences between AR leaf devices and regular NVE devices.

BEST PRACTICE: For best results in EVPN-VXLAN networks with a combination of leaf devices that don't support IGMP or MLD snooping (such as QFX5100 switches) and leaf devices that do support IGMP or MLD snooping (such as QFX5110 switches), follow these recommendations based on the VTEP scaling you need:

- VTEP scaling with 100 or more VTEPs: Enable AR for all supporting devices in the network, and don't use IGMP or MLD snooping.
- VTEP scaling with less than 100 VTEPs: Enable AR and IGMP snooping on the AR replicators and other AR leaf devices that support IGMP or MLD snooping, but disable AR on leaf devices that don't support IGMP or MLD snooping (so those act as regular NVE devices and not as AR leaf devices).

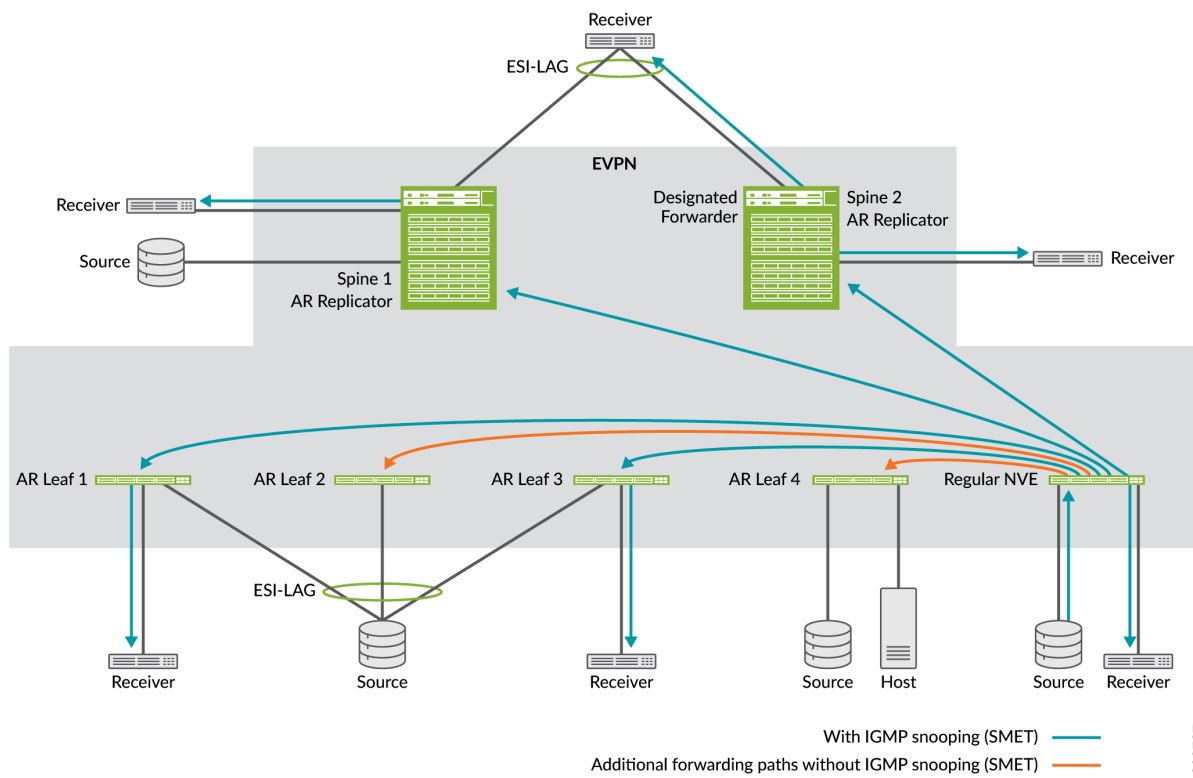
Multicast Forwarding Use Cases in an EVPN Network with AR Devices

This section shows multicast traffic flow in several common use cases where the source is located behind different AR devices or regular NVE devices in an EVPN network.

Source Behind a Regular NVE Device

[Figure 72 on page 761](#) shows an EVPN network with multicast traffic from a source attached to Regular NVE, a device that doesn't support AR.

Figure 72: Multicast Source Traffic Ingress at a Regular NVE Device



In this case, forwarding behavior is the same whether or not AR is enabled because the ingress device, Regular NVE, is not an AR leaf device sending traffic for replication to an AR replicator. Regular NVE employs the usual ingress replication forwarding rules as follows:

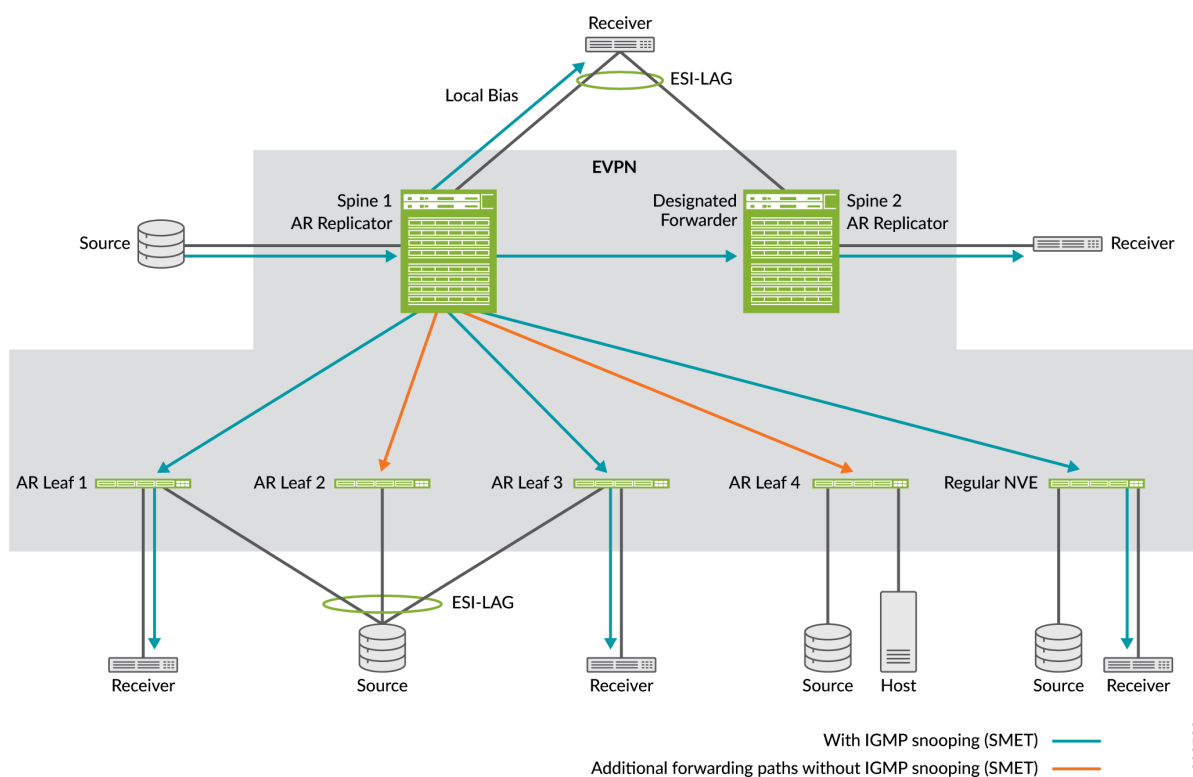
- Without IGMP or MLD snooping and SMET forwarding enabled, Regular NVE floods the traffic to all the other devices in the EVPN network.
- With IGMP or MLD snooping and SMET forwarding enabled, Regular NVE only forwards the traffic to other devices in the EVPN network with active listeners. In this case, Regular NVE forwards to all the other devices except AR Leaf 2 and AR Leaf 4.

Spine 1 and Spine 2 replicate and forward the traffic toward their local single-homed or multihomed receivers or external gateways using the usual EVPN network ingress replication local bias or DF behaviors. In this case, [Figure 72 on page 761](#) shows that Spine 2 is the elected DF and forwards the traffic to a multihomed receiver on an Ethernet segment that the two spine devices share.

Source Behind an AR Replicator Device

Figure 73 on page 762 shows an EVPN network with multicast traffic from a source attached to an AR replicator device called Spine 1.

Figure 73: Multicast Source Traffic Ingress at an AR Replicator Device



In this case, forwarding behavior is the same whether or not AR is enabled because the ingress device, Spine 1, is not an AR leaf device sending traffic for replication to an AR replicator. Spine 1, although it's configured as an AR replicator device, does not act as an AR replicator. Instead, it employs the usual ingress replication forwarding rules as follows:

- Without IGMP or MLD snooping and SMET forwarding enabled, Spine 1, the ingress device, floods the traffic to all the other devices in the EVPN network.
- With IGMP or MLD snooping and SMET forwarding enabled, Spine 1 only forwards the traffic to other devices in the EVPN network with active listeners. In this case, Spine 1 forwards to the other devices except AR Leaf 2 and AR Leaf 4, and to Spine 2, which has a local interested receiver.
- Spine 1 also replicates the traffic toward local single-homed or multihomed receivers or external gateways using the usual EVPN network ingress replication local bias or DF behaviors. In this case,

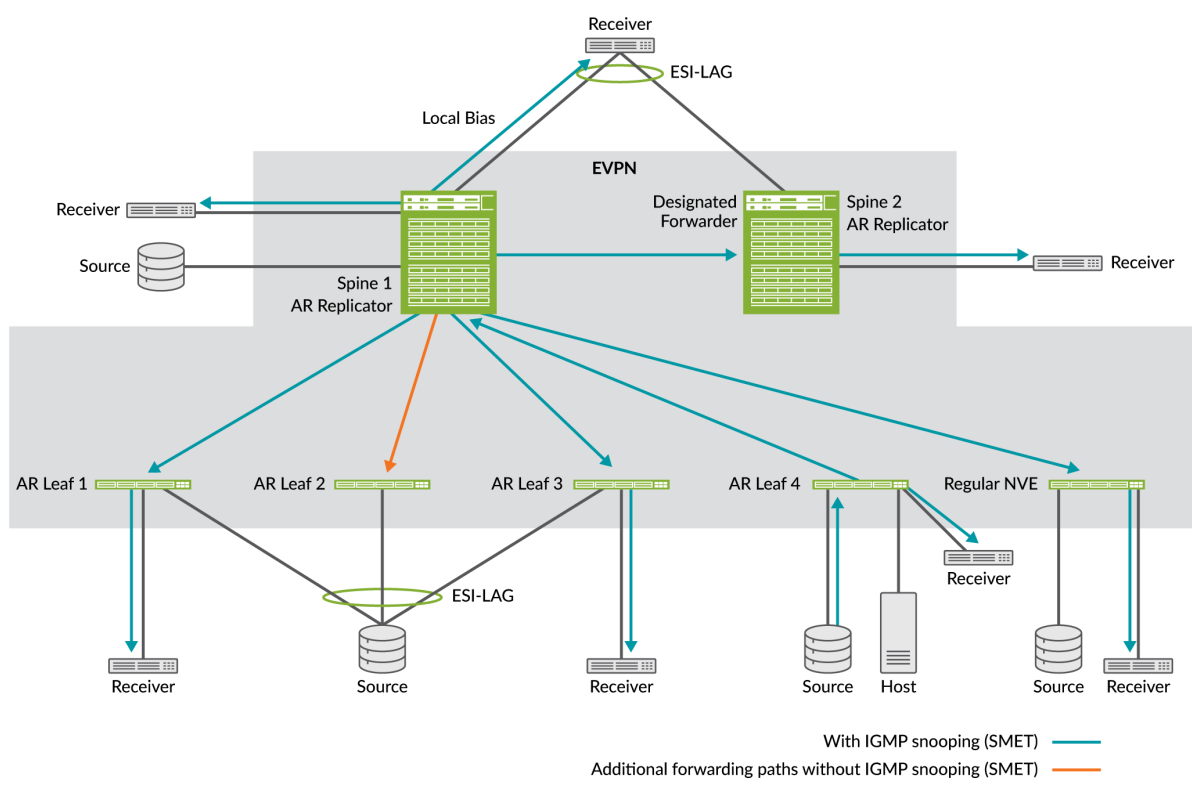
Spine 1 uses local bias and forwards the traffic to a multihomed receiver due to local bias rules (whether or not it is the DF on that Ethernet segment).

Spine 2 forwards the traffic received from Spine 1 to its local receiver, does a local bias check for its multihomed receiver, and skips forwarding to the multihomed receiver even though it is the DF for that Ethernet segment.

Source Behind an AR Leaf Device (Single-Homed Ethernet Segment)

Figure 74 on page 763 shows an EVPN network with multicast traffic from a source attached to the AR leaf device called AR Leaf 4. Traffic flow is the same whether the ingress AR leaf device and AR replicator are operating in extended AR mode or not, because the source isn't multihomed to any other leaf devices.

Figure 74: Source Traffic Ingress at an AR Leaf Device (Single-Homed Source)



Without IGMP or MLD snooping and SMET forwarding enabled:

- AR Leaf 4 forwards one copy of the multicast traffic to an advertised AR replicator device, in this case Spine 1, using the AR overlay tunnel secondary IP address that you configured on the loopback interface lo0 for Spine 1.

- AR Leaf 4 also applies local bias forwarding rules and replicates the multicast traffic to its locally attached receiver.
- Spine 1 receives the multicast traffic on the AR overlay tunnel and replicates and forwards it on behalf of Leaf 4 using the usual ingress replication overlay tunnels to all the other devices in the EVPN network, including Spine 2. Spine 1 skips forwarding the traffic back to the ingress device AR Leaf 4 (according to split horizon rules).

With IGMP or MLD snooping and SMET forwarding enabled:

- AR replicator Spine 1 further optimizes replication by only forwarding the traffic towards other devices in the EVPN network with active listeners. In this case, Spine 1 skips forwarding to AR Leaf 2.

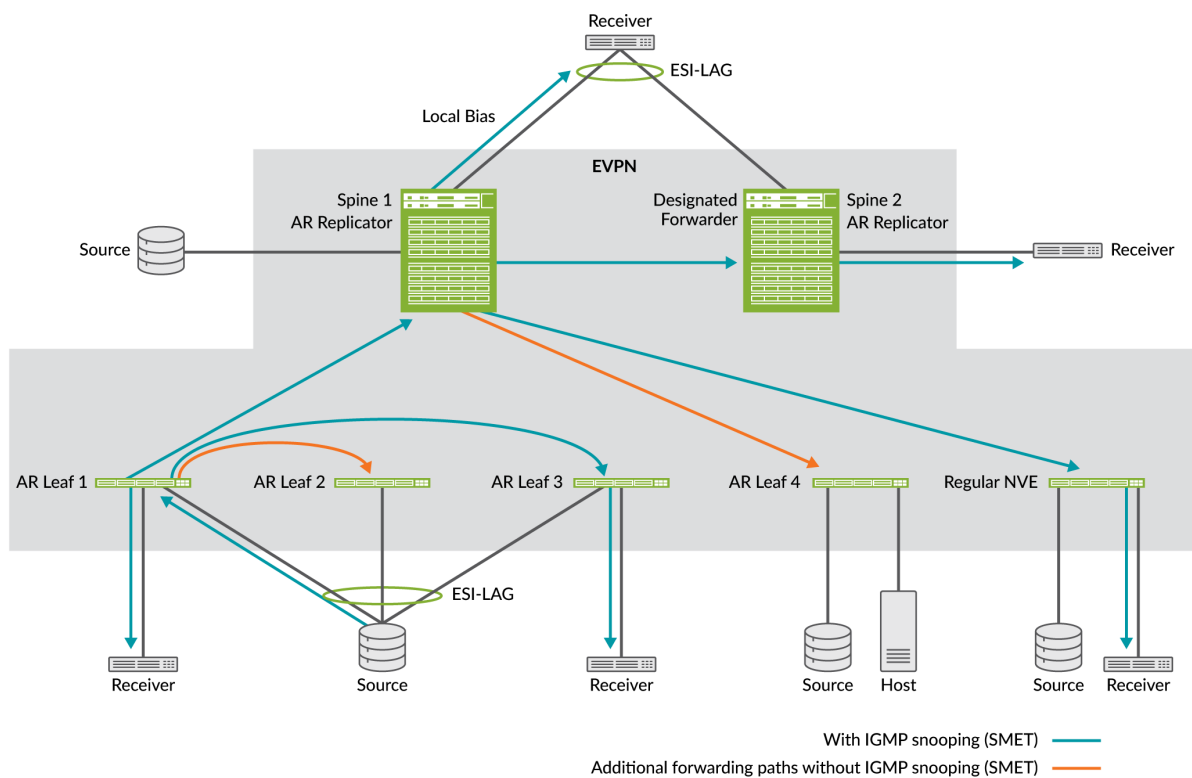
Spine 1 also replicates and forwards the traffic to its local receivers and toward multihomed receivers or external gateways using the usual EVPN network ingress replication local bias or DF behaviors. In this case, Spine 1 forwards the traffic to a local receiver and uses local bias to forward to a multihomed receiver (although it is not the DF for that Ethernet segment).

Spine 2 forwards the traffic received from Spine 1 to its local receiver, does a local bias check for its multihomed receiver, and skips forwarding to the multihomed receiver even though it is the DF for that Ethernet segment.

Source Behind an AR Leaf Device (Multihomed Ethernet Segment) with Extended AR Mode

Figure 75 on page 765 shows an EVPN network with multicast traffic from a source on a multihomed Ethernet segment operating in extended AR mode. The source is multihomed to three AR leaf devices and might send the traffic to any one of them. In this case, AR Leaf 1 is the ingress device.

Figure 75: Source Traffic Ingress at an AR Leaf Device (Multihomed Source)



Without IGMP or MLD snooping and SMET forwarding enabled:

- AR Leaf 1 forwards one copy of the multicast traffic to one of the advertised AR replicator devices, in this case, Spine 1, using the AR overlay tunnel secondary IP address that you configured on the loopback interface lo0 for Spine 1.
- Operating in extended AR mode:
 - AR Leaf 1 also replicates and forwards the multicast traffic to all of its multihoming peers (AR Leaf 2 and AR Leaf 3) for the source Ethernet segment.
 - Spine 1 receives the multicast traffic on the AR overlay tunnel and replicates and forwards it using the usual ingress replication overlay tunnels to the other devices in the EVPN network *except* AR Leaf 1 and its multihoming peers AR Leaf 2 and AR Leaf 3.

With IGMP or MLD snooping and SMET forwarding enabled, the AR leaf and AR replicator devices further optimize multicast replication in extended AR mode as follows:

- Besides sending to AR replicator Spine 1, AR Leaf 1 also replicates and forwards the multicast traffic only to its multihoming peers that have active listeners (AR Leaf 3).
- Spine 1 replicates traffic for AR Leaf 1 only to other devices in the EVPN network with active listeners. In this case, that includes only the regular NVE device and Spine 2.

Spine 1 also replicates and forwards the traffic to its local receivers and toward multihomed receivers or external gateways using the usual EVPN network ingress replication local bias or DF behaviors. In this case, Spine 1 uses local bias to forward to a multihomed receiver although it is not the DF for that Ethernet segment. Spine 2 receives the traffic from Spine 1, forwards the traffic to its local receiver, does a local bias check for its multihomed receiver, and skips forwarding to the multihomed receiver even though it is the DF for that Ethernet segment.

Configure Assisted Replication

IN THIS SECTION

- [Configure an AR Replicator Device | 766](#)
- [Configure an AR Leaf Device | 768](#)

Assisted replication (AR) helps optimize multicast traffic flow in EVPN networks. To enable AR, you configure devices in the EVPN network to operate as AR replicator and AR leaf devices. Available AR replicator devices that are better able to handle the processing load help perform multicast traffic replication and forwarding tasks for AR leaf devices.

You don't need to configure any options on AR replicator or AR leaf devices to accommodate devices that don't support AR, which we refer to as regular network virtualization edge (NVE) devices. Regular NVE devices use the usual EVPN network and overlay tunnel ingress replication independently of AR operation within the same EVPN network. AR replicator devices also don't need to distinguish between AR leaf and regular NVE devices when forwarding multicast traffic, because AR replicator devices also use the existing ingress replication overlay tunnels for all destinations.

Configure an AR Replicator Device

Devices you configure as AR replicator devices advertise their AR capabilities and AR IP address to the EVPN network. The AR IP address is a loopback interface address you configure on the AR replicator device. AR leaf devices receive these advertisements to learn about available AR replicators to which

they can offload multicast replication and forwarding tasks. AR leaf devices automatically load-balance among multiple available AR replicator devices.

To configure an AR replicator device:

1. Configure the loopback interface lo0 with a secondary IP address specifically for AR functions. The AR replicator advertises this IP address to the network in the EVPN Type 3 AR tunnel routes. (See ["AR Route Advertisements" on page 757](#) for details.)

```
user@ar-replicator# set interfaces lo0 unit
0 family inet address AR-IP-address
```

2. Configure the AR replicator device using the loopback interface you configured in Step 1.

```
user@ar-replicator# set protocols evpn assisted-replication replicator inet AR-IP-address
```

3. Set the type of IP address the AR replicator uses as the ingress source address in the overlay tunnel encapsulation for the replicated traffic.

```
user@ar-replicator# set protocols evpn assisted-replication replicator vxlan-encapsulation-
source-ip
ingress-replication-ip
```

NOTE: The default value for vxlan-encapsulation-source-ip is retain-incoming-source-ip. (See [replicator](#).) With AR replicators that are not capable of retaining the source IP address, such as QFX Series devices in an EVPN-VXLAN network, the vxlan-encapsulation-source-ip statement is mandatory and must be set to the ingress-replication-ip option instead of the default value.

When you set the ingress-replication-ip VXLAN encapsulation option, the AR replicator advertises that it is not capable of retaining the source IP address of the originating AR leaf device when replicating and forwarding the traffic to the regular ingress replication overlay tunnels. This means it supports only extended AR mode. (See ["Extended AR Mode for Multihomed Ethernet Segments" on page 758](#).) Other AR devices in the EVPN network receive these advertisements and also use extended AR mode to operate compatibly.

The retain-incoming-source-ip setting is reserved for future compatibility with EVPN network devices and overlays that are capable of retaining the incoming source IP address in replicated traffic.

Configure an AR Leaf Device

Devices you configure as AR leaf devices receive traffic from multicast sources and offload the replication and forwarding to an AR replicator device. AR replicator devices advertise their AR capabilities and AR IP address, and AR leaf devices automatically load-balance among the available AR replicators.

To configure an AR leaf device:

1. Configure the device into the AR leaf role.

```
[edit]user@ar-leaf# set protocols
evpn assisted-replication leaf
```

2. (Optional) By default, AR leaf devices delay for 10 seconds after receiving an AR replicator advertisement before starting to send traffic to that AR replicator device. If needed, you can adjust the delay (in seconds) to make sure AR replicator devices have fully learned the current EVPN state from the network (such as after an AR replicator goes down and comes up again).

For example, to change the delay to 30 seconds:

```
[edit]user@ar-leaf# set protocols
evpn assisted-replication leaf replicator-activation-delay 30
```

Verify Assisted Replication Setup and Operation

Several commands help verify that AR replicator devices have advertised their AR IP address and capabilities, and AR leaf devices have learned how to reach available AR replicator devices and EVPN multihoming peers to operate in extended AR mode.

1. View AR information received from AR replicator EVPN Type 3 (IMET) route advertisements.

```
user@device> show route table bgp.evpn.0 match-prefix
3:* extensive
```

This command displays the Type 3 routes for an EVPN instance to the available AR replicator devices (AR overlay tunnel next hops), including Type ASSISTED-REPLICATION and Role AR-REPLICATOR in the PMSI section of the output.

Session Id: 0\

...

2. Check the available AR replicators and AR mode in which they are operating.

```
user@device> show evpn multicast-snooping assisted-replication
replicators
```

You can run this command on an AR leaf or AR replicator device. The output shows the available AR replicator devices, their AR operating mode, and the next hops and overlay tunnel interfaces used to reach each AR replicator device from the device where you run the command.

For example:

```
user@ar-leaf> show evpn multicast-snooping assisted-replication
replicators
Instance: default-switch
AR Role: AR Leaf

VN Identifier: 100
Operational Mode: Extended AR
  Replicator IP  Nexthop Index  Interface  Mode
  192.168.102.1  1772             vtep.32770 Extended AR
  192.168.102.2  1797             vtep.32772 Extended AR
```

3. (Available when IGMP or MLD snooping is enabled) View AR leaf device overlay tunnel next hops to load-balance among the advertised AR replicator devices.

```
user@ar-leaf> show
evpn multicast-snooping assisted-replication next-hops
```

The following sample command shows the load-balancing next hops toward the advertised AR replicators that the AR leaf device detected in Step 2.

Also, when an AR leaf device load-balances by assigning an AR replicator device to each VLAN or VNI, this command tags that AR replicator as the (Designated Node) in the output (which the example below also shows). AR leaf devices that load-balance based on active traffic flow don't display this tag. (See ["AR Leaf Device Load Balancing With Multiple Replicators" on page 759.](#))

```
user@ar-leaf> show evpn multicast-snooping assisted-replication
next-hops
Instance: default-switch
```

```

VN Identifier: 100
Load Balance Nexthop Index: 131091
Load balance to:
Nexthop Index    Interface    AR IP
1772             vtep.32770   192.168.102.1 (Designated Node)
1797             vtep.32772   192.168.102.2

```

4. (Available when IGMP or MLD snooping is enabled) View AR leaf device multihomed peers. Include the extensive option to also see the multihomed ESs shared with peers.

```

user@ar-leaf> show
evpn multicast-snooping assisted-replication multihomed-peers <extensive>

```

For example:

```

user@ar-leaf> show evpn multicast-snooping assisted-replication
multihomed-peers extensive
Instance: default-switch

Neighbor address: 192.168.0.2
Nexthop Index: 1746
Interface: vtep.32768
Local Multi-homed ESIs
00:11:22:33:44:55:66:77:88:99
00:11:22:33:44:55:66:77:88:aa

```

5. (Available when IGMP or MLD snooping is enabled) View IGMP or MLD snooping core next hops on the AR replicator for each multicast group and VLAN or VNI, and follow those to corresponding AR load-balancing and multihoming peer device next hops. Use the following commands:

- `show evpn igmp-snooping proxy extensive`
- `show evpn mld-snooping proxy extensive`
- `show evpn multicast-snooping next-hops detail`

For example, in the output from the following commands, you can see the Downstream interface Addr field for next-hop ID 131092 shows the following:

- The load-balancing next-hop index 131091, which you also see with the `show evpn multicast-snooping assisted-replication next-hops` command in Step 3.
- The multihoming peer device interface and next hop index vtep.32768-(1746), which you also see with the `show evpn multicast-snooping assisted-replication next-hops` command in Step 4.

```
user@ar-replicator> show evpn igmp-snooping
proxy extensive

Instance: default-switch
VN Identifier: 100
      Group      Source      Local      Remote      Corenh
      233.252.0.1  0.0.0.0      1          5           131092
VN Identifier: 200
      Group      Source      Local      Remote      Corenh
      233.252.0.1  0.0.0.0      1          5           131092

user@ar-replicator> show evpn multicast-snooping
next-hops detail
Family: INET
ID      Refcount KRefCount  Downstream interface Addr
131092      6        2    vtep.32768-(1746)
Flags 0x2100 type 0x18 members 0/0/0/5/0
Address 0xc54ba44
```

Release History Table

Release	Description
18.4R2	Starting in Junos OS Releases 18.4R2 and 19.4R1, you can enable AR on QFX Series switches that support EVPN-VXLAN, which includes the QFX10000 line of switches and QFX5110 and QFX5120 switches.

RELATED DOCUMENTATION

- [Overview of Multicast Forwarding with IGMP Snooping or MLD Snooping in an EVPN-VXLAN Environment | 675](#)
- [Overview of Selective Multicast Forwarding | 747](#)
- [assisted-replication | 1547](#)

[igmp-snooping](#)[mld-snooping](#)

Optimized Inter-Subnet Multicast in EVPN Networks

SUMMARY

Enable inter-subnet multicast (OISM) to optimize multicast traffic routing and forwarding in an EVPN edge-routed bridging (ERB) overlay fabric. With OISM, your network can also support multicast traffic flow between devices inside and outside of the EVPN data center.

IN THIS SECTION

- [Overview of OISM | 773](#)
- [OISM Components | 775](#)
- [How OISM Works | 786](#)
- [Configure Common OISM Elements on Border Leaf Devices and Server Leaf Devices \(Symmetric Bridge Domains\) | 798](#)
- [Configure Border Leaf Device OISM Elements \(Symmetric Bridge Domains\) | 801](#)
- [Configure Server Leaf Device OISM Elements \(Symmetric Bridge Domains\) | 804](#)
- [CLI Commands to Verify the OISM Configuration | 804](#)

Overview of OISM

IN THIS SECTION

- [Benefits of OISM | 774](#)
- [OISM with Other Multicast Protocols and Optimizations in EVPN Fabrics | 774](#)

Optimized inter-subnet multicast (OISM) is a multicast traffic optimization feature. It operates at Layer 2 and Layer 3 in EVPN-VXLAN edge-routed bridging (ERB) overlay fabrics. The design is based on the IETF draft specification <https://datatracker.ietf.org/doc/html/draft-ietf-bess-evpn-irb-mcast>. You can apply OISM configuration and operation to multicast traffic but not to broadcast or unknown unicast traffic..

In EVPN ERB overlay fabric designs, the leaf devices in the fabric route traffic between tenant bridge domains (that is, between VLANs). When you enable OISM, the leaf devices route inter-subnet multicast traffic locally through IRB interfaces using the control plane multicast states. With local routing between VLANs, the receiver IRB interface doesn't send the routed multicast traffic out into the EVPN core. The local routing model helps minimize the traffic load within the EVPN core. It also avoids traffic hairpinning.

OISM leaf devices also selectively forward traffic into the EVPN core only toward other EVPN devices with interested receivers. Selective forwarding further improves multicast traffic performance in the EVPN data center.

Finally, with OISM enabled, data centers with the ERB overlay model support multicast traffic flow between devices inside and outside of the EVPN fabric. Without OISM, data center designers must use the centrally routed bridging (CRB) overlay model to support multicast with external sources or receivers. OISM uses IRB interfaces on a multicast VLAN (M-VLAN) on the border leaf devices to route traffic to and from an external Protocol-Independent Multicast (PIM) domain. The border leaf devices employ a single supplemental bridge domain (SBD) to carry the traffic from external sources toward receivers within the EVPN fabric.

NOTE: You can also use OISM to handle external multicast traffic using pure Layer 3 interfaces rather than M-VLAN IRB interfaces. This documentation focuses on the M-VLAN use case.

Benefits of OISM

- Enables EVPN-VXLAN data centers with the ERB overlay model to support multicast traffic with sources and receivers outside of the data center.
- Minimizes multicast control packets and replicated data packets in the EVPN fabric core to optimize data center multicast performance in scaled designs.

OISM with Other Multicast Protocols and Optimizations in EVPN Fabrics

OISM works with the following multicast protocols:

- IGMPv2.
- PIM, which facilitates both local routing and external multicast traffic routing.

OISM works with the following other EVPN multicast optimizations:

- IGMP snooping on the access side on the leaf devices.

With IGMP snooping enabled, a leaf device that receives multicast traffic forwards it only toward other devices with interested receivers.

- Multihoming support in an Ethernet segment using EVPN Type 7 (Join Sync) and Type 8 (Leave Sync) routes.

EVPN fabric devices advertise these route types to synchronize the multicast state among EVPN devices that are multihoming peers.

- Selective multicast Ethernet tag (SMET) forwarding in the EVPN fabric core using EVPN Type 6 routes.

EVPN devices use Type 6 routes to limit forwarding within the EVPN core only to receivers interested in receiving traffic for a multicast group. You can use OISM to make this optimization work in EVPN ERB overlay fabrics. When you configure IGMP snooping, the fabric enables SMET forwarding with OISM automatically.

OISM Components

IN THIS SECTION

- [OISM Device Roles | 776](#)
- [PIM Domain with External Multicast Sources and Receivers | 777](#)
- [OISM Bridge Domains \(VLANs\) | 778](#)
- [Configuration Elements for OISM Devices \(Symmetric Bridge Domains\) | 781](#)

The OISM environment includes:

- Leaf devices in the EVPN fabric that function in border roles and server access roles.
- External multicast sources and receivers in an external Layer 3 PIM domain.
- Bridge domain (VLAN) configurations that enable the fabric to route multicast traffic between internal and external devices.

The EVPN-VXLAN ERB overlay design includes lean spine devices that support Layer 3 transit functions for the leaf devices. The lean spine devices don't usually perform any OISM functions.

The following sections describe these OISM components.

OISM Device Roles

Figure 76 on page 776 shows a simple EVPN-VXLAN ERB overlay fabric and the OISM device roles in the fabric.

Figure 76: EVPN Fabric with OISM

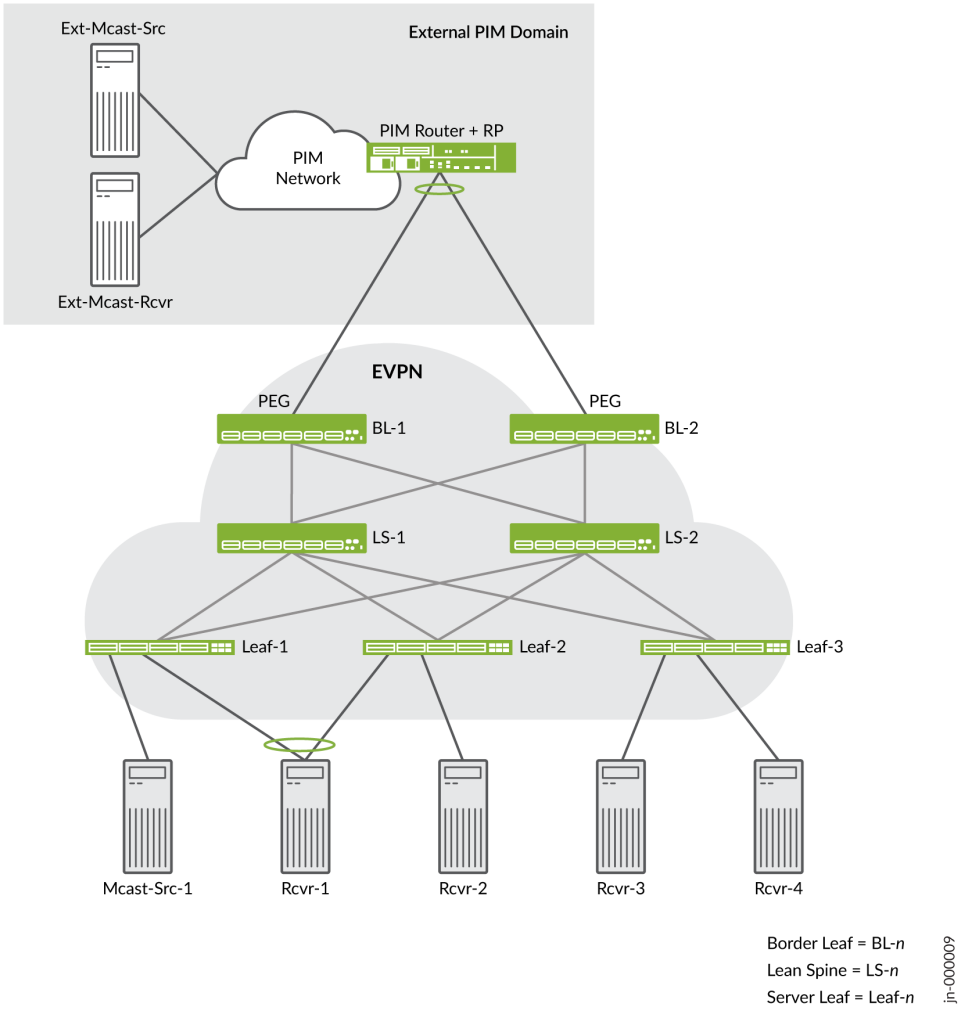


Table 34 on page 777 summarizes the device roles.

Table 34: EVPN Fabric OISM Device Roles

Device Role	Description
Border leaf (BL)	OISM leaf devices in the EVPN fabric underlay and overlay. Border leaf devices function as gateways interconnecting the EVPN fabric to multicast devices (sources and receivers) outside the data center in an external PIM domain. These devices serve in the PIM EVPN gateway (PEG) role.
Lean spine (LS)	Spine devices in the underlay of the EVPN data center fabric. These devices usually operate as lean spines that support the EVPN underlay as IP transit devices. The lean spines might also act as route reflectors in the fabric. NOTE: You don't need to configure OISM on the lean spine devices unless the devices also serve as border devices for external multicast traffic. In that case, you configure the same OISM elements as you configure on border leaf devices.
Server leaf (Leaf)	OISM leaf devices on the access side in the EVPN fabric underlay and overlay. Server leaf devices are often top-of-rack (ToR) switches. These devices connect the EVPN fabric to multicast sources and multicast receiver hosts on bridge domains or VLANs within the data center.

PIM Domain with External Multicast Sources and Receivers

In [Figure 76 on page 776](#), the OISM border leaf devices connect to multicast sources and receivers outside the EVPN fabric in a representative external PIM domain. The multicast devices in the external PIM domain follow standard PIM protocol procedures; their operation is not specific to OISM. External multicast source and receiver traffic flow at Layer 3 through the PIM domain.

You can use OISM to route and to forward multicast traffic in an EVPN-VXLAN ERB overlay fabric between devices in the following use cases:

- Internal multicast sources and internal multicast receivers
- Internal multicast sources and external multicast receivers
- External multicast sources and internal multicast receivers

For simplicity, the external PIM domain represented in this documentation comprises:

- A PIM router (a device such as an MX Series router) that doubles as the PIM rendezvous point (RP).
- An external source.
- An external receiver.

NOTE: The OISM border leaf devices can route multicast traffic from or to devices outside of the fabric using either of the following:

- IRB interfaces on a multicast VLAN (M-VLAN)
- Pure Layer 3 interfaces

This document focuses on the M-VLAN solution. The next sections describe more about how OISM uses an M-VLAN for external multicast traffic.

See ["Overview of Multicast Forwarding with IGMP Snooping or MLD Snooping in an EVPN-VXLAN Environment" on page 675](#) for an overview of connecting EVPN-VXLAN fabrics to an external PIM domain using a Layer 2 M-VLAN or multiple Layer 3 links.

OISM Bridge Domains (VLANs)

[Table 35 on page 778](#) summarizes the OISM bridge domains or VLANs and describes how OISM uses them.

NOTE: References in this document to *all OISM devices* correspond to the border leaf and server leaf devices on which you enable OISM.

Table 35: OISM Bridge Domains or VLANs

Bridge Domain/ VLAN	Description	Configure on:
Multicast VLAN (M-VLAN) (Also called the <i>external VLAN</i>)	<p>VLAN that connects the EVPN fabric to multicast sources and multicast receivers outside the data center through an external multicast router.</p> <p>You can multihome the external multicast router to multiple border leaf device M-VLAN IRB interfaces in the same EVPN Ethernet segment. The usual EVPN multihoming designated forwarder (DF) rules apply to send only one copy of the traffic in the ES on the M-VLAN.</p> <p>Configure the M-VLAN as a VLAN that is not the SBD or any of the revenue bridge domains (or VLANs) in the EVPN data center.</p>	Border leaf devices

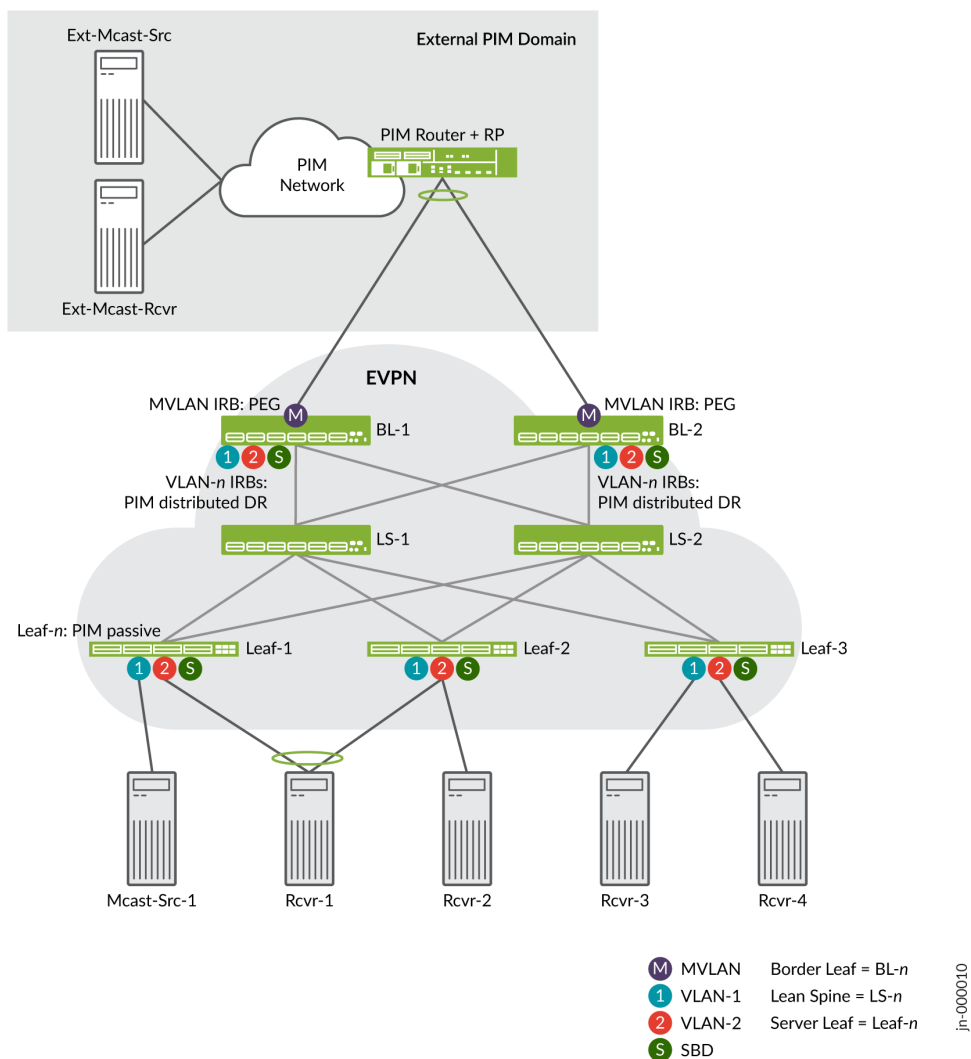
Table 35: OISM Bridge Domains or VLANs (*Continued*)

Bridge Domain/ VLAN	Description	Configure on:
Revenue bridge domains (VLANs)	Bridge domains or VLANs for subscribers to the services that the data center provides. We sometimes call revenue bridge domains <i>customer VLANs</i> . These VLANs are not specific to OISM, but the multicast sources and receivers in the data center are in these bridge domains.	All OISM devices
Supplemental bridge domain (SBD)	<p>Bridge domain (VLAN) that enables support for external multicast traffic and implements SMET optimization in the EVPN core. One SBD serves all OISM leaf devices in the fabric. The SBD carries:</p> <ul style="list-style-type: none"> • Routed multicast data traffic from external multicast sources to EVPN devices in the fabric with interested receivers. • SMET EVPN Type 6 routes from originating leaf devices to other EVPN leaf devices, which enables selective forwarding in the EVPN core. <p>NOTE: The SBD is central to OISM operation. The SBD IRB interface must always be up for OISM to work.</p> <p>Configure the SBD as a distinct VLAN that is not the M-VLAN or a revenue bridge domain in the EVPN data center.</p>	All OISM devices

The OISM implementation uses a *symmetric bridge domains* model.

The following figure shows how you configure the bridge domains and VLANs in the EVPN fabric on the OISM devices with this model.

Figure 77: EVPN Fabric with OISM Symmetric Bridge Domains Model



In the symmetric bridge domains model, you must configure the SBD and all revenue bridge domains on all OISM border leaf and server leaf devices. You configure the M-VLAN on the border leaf devices that connect to the external PIM domain. The lean spine devices in the fabric usually serve only as IP transit devices and possibly as route reflectors. As a result, you don't need to configure these elements on the lean spine devices.

Configuration Elements for OISM Devices (Symmetric Bridge Domains)

This section summarizes the elements you need to configure on:

- All OISM devices—devices in both the border leaf and server leaf roles.
- Border leaf devices only.
- Server leaf devices only.

Some elements are optional, which the description notes.

[Table 36 on page 781](#) lists the elements you configure on all OISM devices.

Table 36: Configuration Elements on All OISM Devices

Configuration Element	Description
OISM	<p>Enable OISM globally, and enable OISM routing functions in Layer 3 virtual routing and forwarding (VRF) instances.</p> <p>A device with OISM enabled advertises EVPN Type 3 inclusive multicast Ethernet tag (IMET) routes as follows:</p> <ul style="list-style-type: none"> • The device advertises an IMET route for each configured revenue bridge domain (VLAN) and the SBD. • The IMET routes include the EVPN multicast flags extended community with a flag indicating OISM support.
Revenue bridge domains (customer VLANs) and corresponding IRB interfaces	<p>Configure revenue bridge domains (customer VLANs) according to your data center services requirements. You must configure all revenue bridge domain VLANs and corresponding IRB interfaces on all OISM devices. See "OISM Bridge Domains (VLANs)" on page 778 for more information.</p>
SBD (VLAN) and corresponding IRB interface	<p>Configure the SBD and its IRB interface on all OISM devices. The SBD can be any VLAN that is distinct from the M-VLAN or any revenue bridge domain VLANs in the EVPN data center. See "OISM Bridge Domains (VLANs)" on page 778 for more information.</p> <p>You identify this VLAN as the SBD in the Layer 3 virtual routing and forwarding (VRF) instance that supports OISM routing. EVPN IMET routes for the SBD IRB interface include the OISM SBD flag in the multicast flags extended community.</p>

Table 36: Configuration Elements on All OISM Devices *(Continued)*

Configuration Element	Description
Layer 2 multicast optimizations—IGMP snooping in proxy mode, SMET	<p>Enable IGMP snooping as part of OISM optimizations. With IGMP snooping enabled, the device routes multicast traffic or forwards the traffic only toward interested access-side receivers. (Receivers send IGMP reports to express interest in receiving traffic for a multicast group.)</p> <p>Enabling IGMP snooping also automatically enables the device to advertise SMET Type 6 routes. With SMET, the device sends copies of the traffic into the EVPN core only toward other devices that have interested receivers.</p>
Layer 3 virtual routing and forwarding (VRF) instance	Configure a routing instance (instance type vrf) that supports the Layer 3 OISM routing functions.
originate-smet-on-revenue-vlan-too	<p>(Optional) Enable the device to originate SMET Type 6 routes for revenue bridge domains (as well as on the SBD) upon receiving local IGMP reports. By default, OISM devices originate Type 6 routes only on the SBD.</p> <p>Use this option for compatibility with other vendor devices that don't support OISM. Those devices can't create the right states for the revenue bridge domain VLANs upon receiving Type 6 routes on the SBD.</p>
install-star-g-routes	<p>(Required on the QFX10000 line of switches, optional on the QFX5000 line of switches) Enable the Routing Engine on the device to install (*,G) multicast routes on the Packet Forwarding Engine for all of the revenue bridge domain VLANs in the routing instance immediately upon receiving an EVPN Type 6 route. Setting this option helps minimize traffic loss when multicast traffic first arrives.</p> <p>See "Latency and Scaling Trade-Offs for Installing Multicast Routes with OISM (install-star-g-routes Option)" on page 796 for details on when to set this option.</p>

Table 37 on page 783 list the elements you configure on the border leaf devices.

Table 37: Configuration Elements on OISM Border Leaf Devices

Configuration Element	Description
M-VLAN (external VLAN) and corresponding IRB interfaces	<p>Configure a VLAN to serve as the M-VLAN. This VLAN must be distinct from the SBD or any revenue bridge domain VLANs in the EVPN data center. See "OISM Bridge Domains (VLANs)" on page 778 for more information.</p> <p>You can link multiple border leaf device M-VLAN IRB interfaces to the external multicast router in the same EVPN Ethernet segment. The usual EVPN multihoming DF rules apply to prevent sending duplicate traffic on the M-VLAN.</p>
Layer 2 multicast router interface on M-VLAN ports	<p>Configure this setting on the interfaces that link the border leaf device to the external PIM domain at Layer 2. These interfaces support multicast traffic when the external domain router is multihomed to the border leaf devices. As a result, multihomed M-VLAN use cases require this configuration.</p> <p>You configure the Layer 2 port on which you enable IGMP snooping for the M-VLAN as the multicast router interface.</p>
PIM on M-VLAN IRB interfaces	<p>Configure standard PIM mode on an M-VLAN IRB interface. With this setting, the device:</p> <ul style="list-style-type: none"> Forms PIM neighbor relationships with the other border leaf devices and the external PIM router. The external PIM router might be multihomed in an ES. As a result, EVPN-forwarded traffic might come to a different peer border leaf device. Elects a single last-hop router (LHR) designated router (DR) for the M-VLAN, so only one device does source registration for an internal source toward the PIM RP.
PIM on SBD IRB interfaces	<p>Configure standard PIM mode for SBD IRB interface routing and forwarding. with this setting, the device:</p> <ul style="list-style-type: none"> Routes external multicast source traffic from the M-VLAN to the SBD, and forwards copies toward server leaf devices with multicast receivers. Forms PIM neighbor relationships with the other border leaf devices. Elects a single LHR DR on the SBD among the peer border leaf devices. Only this elected PIM DR forwards the external multicast source traffic on the SBD. This election prevents peer border leaf devices from forwarding duplicate traffic into the EVPN core.

Table 37: Configuration Elements on OISM Border Leaf Devices *(Continued)*

Configuration Element	Description
PIM-EVPN gateway (PEG) role on M-VLAN IRB interfaces	<p>Configure this role on the interfaces that connect the border leaf device to the external PIM router at Layer 2 over the M-VLAN. In this role, the M-VLAN IRB interface uses traditional PIM routing behavior and does local routing as follows:</p> <ul style="list-style-type: none"> • Routes external source traffic from the M-VLAN to local receivers on revenue bridge domains. • Routes external source traffic from the M-VLAN to the SBD. Forward the traffic to the EVPN core only on the SBD to other OISM leaf devices. • Locally routes internal source traffic that arrives on a revenue bridge domain to other revenue bridge domains and the M-VLAN. The device doesn't forward the traffic back into the EVPN core. <p>EVPN IMET routes for PEG interfaces include the OISM PEG flag in the multicast flags extended community field of the route.</p>
PIM distributed designated router (DR) mode on revenue bridge domain IRB interfaces	<p>Configure PIM distributed DR mode () to support revenue bridge domain multicast routing and forwarding. In this mode, the device:</p> <ul style="list-style-type: none"> • Forms PIM neighbor relationships with other PIM devices to support multihomed external PIM router connections. • Acts as the LHR DR on its revenue bridge domain IRB interfaces, and creates the PIM state locally from received IGMP reports. As a result: <ul style="list-style-type: none"> • The device can do local multicast routing between the revenue bridge domains (VLANs). • One peer border leaf device becomes the PIM DR that performs source registration toward the PIM RP. PIM hello messages and the PIM DR election process determine the PIM DR.

Table 37: Configuration Elements on OISM Border Leaf Devices *(Continued)*

Configuration Element	Description
PIM accept-join-always-from option and policy on M-VLAN IRB interfaces	<p>Set this option on the M-VLAN IRB interfaces when the external PIM router is multihomed to more than one EVPN border leaf device. With this option, the device can accept and install the same PIM (S,G) join states on multihoming peer border leaf devices. This option supports sending multicast traffic from sources inside the fabric to receivers in the external PIM domain.</p> <p>With multihoming on the M-VLAN, the usual EVPN multihoming DF rules apply in an ES to prevent sending duplicate traffic. If peer border leaf devices have the same valid join states in place, any device that is the EVPN DF can forward the multicast traffic.</p> <p>Configure this statement with policies that specify the interface should always install PIM joins from upstream neighbor addresses that correspond to the external PIM router.</p>

Table 38 on page 785 lists the elements you configure on the server leaf devices.

Table 38: Configuration Elements on OISM Server Leaf Devices

Configuration Element	Description
PIM in passive mode on all revenue bridge domain VLANs and the SBD	<p>Configure this mode to facilitate local routing without all of the traditional PIM protocol functions. The server leaf device:</p> <ul style="list-style-type: none"> Doesn't form PIM neighbor relationships with any other devices (avoids sending or receiving PIM hello messages). Acts as a PIM local RP. The device creates the PIM state locally from IGMP reports. The device also doesn't do source registration. (Only the border leaf devices perform source registration toward the external PIM RP.)
PIM with the accept-remote-source option on SBD IRB interfaces	<p>This option enables an SBD IRB interface to accept multicast traffic from a source that isn't on the same subnet. The server leaf devices require this setting because:</p> <ul style="list-style-type: none"> The border leaf devices route the multicast source traffic from the M-VLAN to the SBD toward the server leaf devices. The traffic arrives at the server leaf device on the SBD IRB interface, which is not a PIM neighbor. The source is not a local source, so without this setting, the device would otherwise drop the traffic.

See the following sections for more details on configuring OISM devices:

- ["Configure Common OISM Elements on Border Leaf Devices and Server Leaf Devices \(Symmetric Bridge Domains\)" on page 798](#)
- ["Configure Border Leaf Device OISM Elements \(Symmetric Bridge Domains\)" on page 801](#)
- ["Configure Server Leaf Device OISM Elements \(Symmetric Bridge Domains\)" on page 804](#)

How OISM Works

IN THIS SECTION

- [Local Routing on OISM Devices | 786](#)
- [Multicast Traffic Forwarding and Routing Within the EVPN Data Center | 788](#)
- [Multicast Traffic to Receivers Outside the EVPN Data Center | 790](#)
- [Multicast Traffic from Sources Outside the EVPN Data Center | 793](#)
- [Latency and Scaling Trade-Offs for Installing Multicast Routes with OISM \(install-star-g-routes Option\) | 796](#)

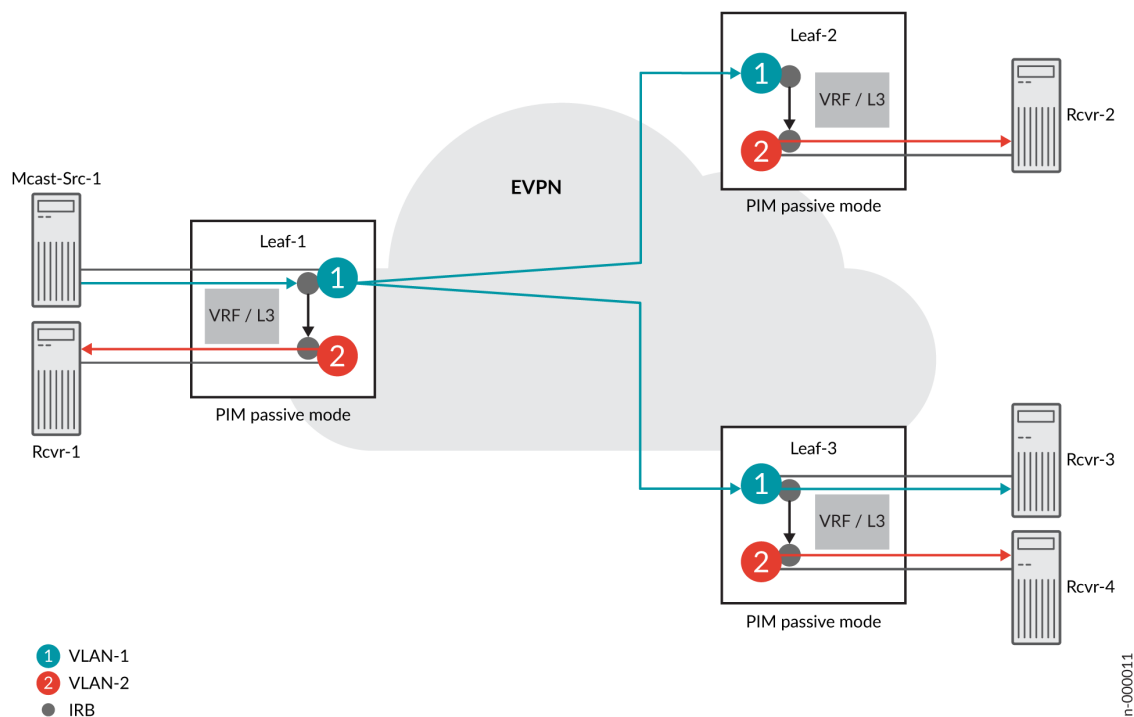
The following sections describe how OISM works and show how the multicast traffic flows in several common use cases.

Local Routing on OISM Devices

In [Figure 78 on page 787](#), we illustrate how local routing and forwarding works in general on OISM devices. As the figure shows, OISM local routing forwards the traffic on the source VLAN. Each leaf

device routes the traffic locally to its receivers on other VLANS, which avoids hairpinning for inter-subnet routing on the same device.

Figure 78: Local Routing with OISM



In this case, the source traffic comes from Mcast-Src-1 on VLAN-1, the blue VLAN. Server leaf devices use IRB interfaces and PIM in passive mode to route traffic between VLANs. With PIM in passive mode, server leaf devices:

- Don't become PIM neighbors with the other leaf devices.
- Act as a local PIM RP, create local PIM state upon receiving IGMP reports, and avoid doing source registration.

As a result, the server leaf devices forward and route multicast traffic within the fabric as follows to receivers interested in the multicast group:

- The ingress leaf device (Leaf-1) forwards the traffic on the source VLAN into the EVPN fabric toward the other leaf devices with interested receivers.
- All of the server leaf devices don't need to forward the traffic back into the EVPN core to another device that is a designated router. Server leaf devices can locally:

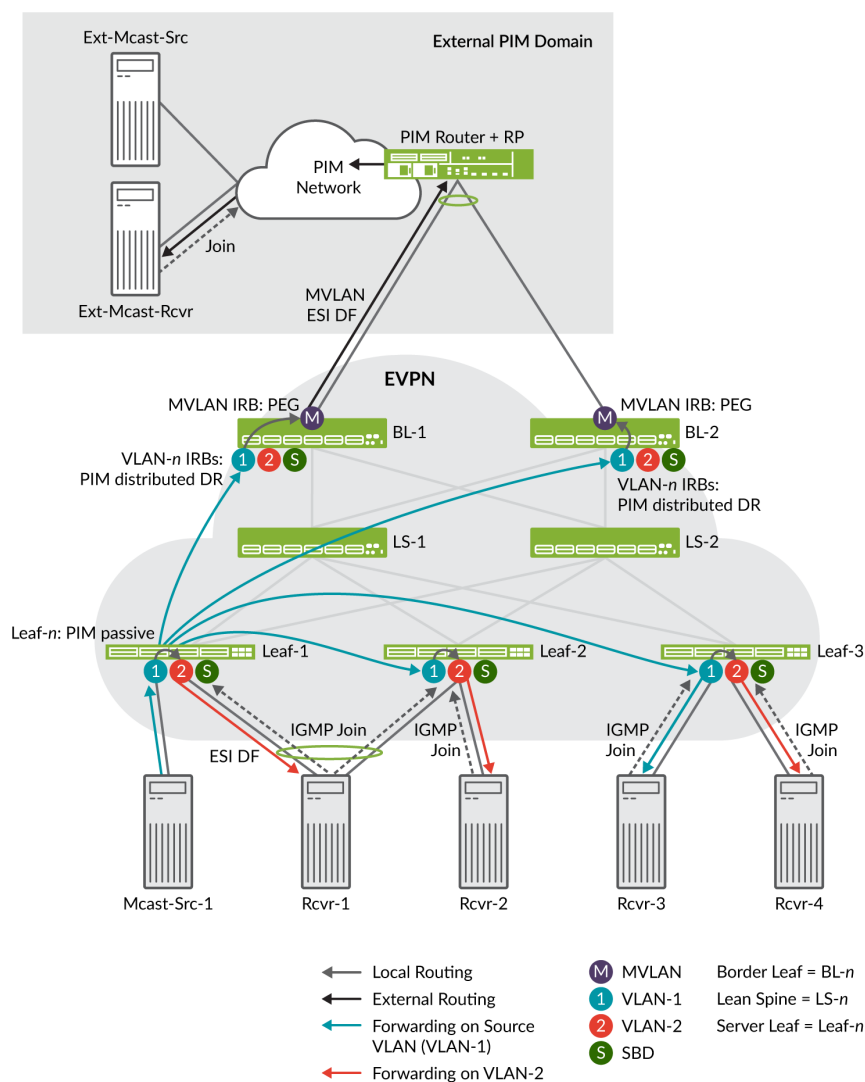
- Forward the traffic on the source VLAN toward local interested receivers on the source VLAN.
- Route the traffic from the source VLAN through the IRB interfaces toward local interested receivers in other VLANs.

Multicast Traffic Forwarding and Routing Within the EVPN Data Center

When the multicast source is inside the EVPN data center, the server leaf devices receive the multicast traffic on the source VLAN. Then they locally route or forward the traffic as described in ["Local Routing on OISM Devices" on page 786](#).

The following figure illustrate OISM local routing and forwarding within an EVPN data center in detail. The figure also shows how local routing works with EVPN multihoming for a multicast receiver.

Figure 79: OISM with an Internal Multicast Source and Multiple Internal Multicast Receivers



In [Figure 79](#) on page 789:

- Receivers on all three server leaf devices sent IGMP reports (join messages) expressing interest in receiving the traffic for a multicast group.
- The multicast source, Mcast-Src-1, is on Leaf-1. The source VLAN is VLAN-1 (the blue VLAN).

- Leaf-1 forwards the traffic on the source VLAN to both Leaf-2 and Leaf-3 because both leaf devices have interested receivers. In this case, both receivers use single-homing.
- Leaf-2 and Leaf-3 forward or locally route the traffic to their interested receivers (Rcvr-2, Rcvr-3, and Rcvr-4) as described in ["Local Routing on OISM Devices" on page 786](#).
- Rcvr-1 on VLAN-2 is multihomed to Leaf-1 and Leaf-2 in an EVPN Ethernet segment, and has expressed interest in receiving the multicast traffic, so:
 - Both server leaf devices, Leaf-1 and Leaf-2, receive the IGMP report.
 - Both Leaf-1 and Leaf-2 locally route the traffic from the source VLAN (VLAN-1) because each device has the PIM passive mode configuration.
 - However, because Leaf-1 is the DF for the EVPN ES, only Leaf-1 forwards the traffic to Rcvr-1.
- The border leaf devices receive the multicast traffic through the EVPN fabric on the source VLAN. Note that the border leaf devices could have local receivers, although we don't show that case. With local receivers, the device also locally routes or forwards the traffic to those receivers the same way the server leaf devices do.

[Figure 79 on page 789](#) also shows that the border leaf devices locally route the traffic from the source VLAN to the M-VLAN toward any remote multicast receivers. Because the M-VLAN has the PEG configuration on both border leaf devices, both devices route traffic to their M-VLAN IRB interface. However, only the DF for the M-VLAN Ethernet segment actually forwards the data on the M-VLAN toward the external PIM domain. EVPN DF rules ensure that the PIM domain doesn't receive duplicate copies of the traffic. The next section describes the multicast control and data traffic flow for an external receiver use case.

Multicast Traffic to Receivers Outside the EVPN Data Center

In [Figure 80 on page 791](#), we illustrate the OISM use case where a multicast source inside the EVPN data center sends multicast traffic to an interested receiver outside the data center. This use case has a multihomed multicast source, and also covers local routing to receivers inside the data center.

The border leaf devices you configure in the OISM PEG role receive the multicast traffic on the source VLAN through the EVPN core. Then the border leaf devices replicate the traffic and route it onto the M-VLAN toward the external PIM domain to reach the external receiver.

Figure 80: OISM with an Internal Multicast Source and an External Multicast Receiver

the multicast traffic for that multicast group (sends a join message). Internal receivers Rcvr-3 (on VLAN-1) and Rcvr-4 (on VLAN-2) also request to join the multicast group and receive the traffic.

Note that the PIM router is multihomed to both BL-1 and BL-2, the PEG devices, in the EVPN fabric. Those connections are in the same ES; the DF election process chooses one of these devices as the DF for the Ethernet segment. Only the DF will forward traffic (on the M-VLAN) toward external receivers.

The source traffic reaches the interested internal and external receivers as follows:

- In the external PIM domain, the PIM RP enters a PIM (*,G) multicast routing table entry. The entry includes the Layer 3 interface toward Ext-Mcast-Rcvr as the downstream interface.
- Mcast-Src-2 (also labeled Rcvr-1) originates the traffic on VLAN-2, the red VLAN. Because the device is multihomed to Leaf-1 and Leaf-2, the device hashes the traffic on VLAN-2 to one of those server leaf devices. In this case, Leaf-2 receives the traffic.
- The red arrows show that Leaf-2 forwards the traffic on the source VLAN, VLAN-2, to:
 - The other server leaf devices with interested receivers—in this case, only Leaf-3.
 - The border leaf devices, which both act in the OISM PEG role.

The figure doesn't show that Rcvr-2 or any receivers behind Leaf-1 sent an IGMP report to join the multicast group. With IGMP snooping and SMET forwarding, Leaf-2 doesn't forward the traffic to Leaf-1 because Leaf-1 has no interested receivers. Leaf-2 also doesn't locally route the traffic to Rcvr-2 for the same reason.

- Leaf-3 receives the source traffic on VLAN-2. Then Leaf-3 routes the traffic locally to VLAN-1 to Rcvr-3. Leaf-3 also forwards the traffic to Rcvr-4 on VLAN-2.
- Both border leaf devices BL-1 and BL-2 also receive the source traffic from the EVPN core. The IRB interface on VLAN-2 on one of these border leaf devices is the PIM DR for VLAN-2. In this case, The PIM DR is on BL-1, so BL-1 sends a PIM Register message toward the PIM RP on the M-VLAN IRB interface.
- The PIM RP sends a PIM Join message back toward BL-1. BL-1 creates an (S,G) multicast routing table entry as follows:
 - The source address is the IP address of Mcast-Src-2 in VLAN-2.
 - The downstream interface is the M-VLAN IRB interface.
- Both BL-1 and BL-2 are PEG devices and configured in PIM distributed DR mode. As a result, both BL-1 and BL-2 receive the PIM Join and create a similar (S,G) state. Both devices route the traffic locally from VLAN-2 to the M-VLAN.

However, only the DF for the M-VLAN ES actually forwards the data on the M-VLAN to the external PIM domain. In this case, BL-1 is the DF and sends the traffic toward the external receiver. (See the

label "MVLAN ESI DF" and the black arrow between BL-1 and the PIM router in [Figure 80 on page 791](#).)

- The PIM RP receives the traffic from the OISM M-VLAN. The PIM router sends the traffic to a Layer 3 interface toward the external receiver.

Multicast Traffic from Sources Outside the EVPN Data Center

[Figure 81 on page 794](#) illustrates the OISM use case where a multicast source outside the EVPN data center sends multicast traffic to receivers inside the data center. This use case exhibits the two main ways OISM uses the SBD in the EVPN core—to carry external multicast source traffic and to advertise SMET Type 6 routes. The Type 6 routes ensure that the border leaf devices only forward the traffic toward the EVPN devices with interested receivers.

OISM border leaf devices receive external multicast source traffic through the M-VLAN IRB interfaces. OISM devices use the SBD to forward traffic toward EVPN server leaf devices with interested receivers on the revenue bridge domains (VLANs). Each leaf device then locally forwards or routes the traffic on the revenue bridge domains (VLANs).

- Both Leaf-1 and Leaf-2 generate an EVPN Type 6 route toward the EVPN core on the SBD. The Type 6 route advertises that Rcvr-1 is interested in the multicast data.
- Both border leaf devices BL-1 and BL-2 receive the Type 6 route on the SBD.
- The Type 6 route (on the SBD) signals the border leaf devices to create a PIM join toward the PIM RP (reachable through the M-VLAN). However, to avoid duplicate join messages, only the border leaf device that is the PIM DR for the SBD generates the PIM join message. In this case, the figure shows the PIM DR for the SBD is BL-1. BL-1 sends the PIM join message toward the PIM RP by way of its neighbor, the M-VLAN IRB interface.
- The PIM RP receives the join message. Then the PIM RP creates a PIM (*,G) entry in the multicast routing table with the M-VLAN IRB interface as the downstream interface.
- The external source Ext-Mcast-Src registers with the PIM RP. The PIM RP has a multicast route for the group with the M-VLAN IRB interface as the downstream interface. As a result, the PIM RP routes the multicast traffic coming in at Layer 3 onto the M-VLAN toward BL-1 or BL-2. In this case, BL-1 receives the traffic on its M-VLAN IRB interface.
- For simplicity, here BL-1 is the PIM DR for the SBD, so BL-1 then routes and forwards the external source traffic as follows:
 - BL-1 forwards a copy of the traffic *on the M-VLAN* toward BL-2 because both border leaf devices are in the PEG role.

See the black arrow from BL-1 toward BL-2.

- BL-1 routes the traffic locally to the SBD IRB interface because BL-1 is the PIM DR for the SBD.

See the small gray arrow from the M-VLAN to the SBD on BL-1.

- BL-1 then forwards a copy of the traffic *on the SBD* into the EVPN core to BL-2.

BL-2 drops the traffic because as a PEG device, BL-2 expects external source traffic only on the M-VLAN IRB interface. BL-1 doesn't expect external source traffic on the SBD IRB interface from BL-1. In other words, BL-2 sees a source interface mismatch (a reverse path forwarding [RFP] failure). See the green arrow from BL-1 toward BL-2.

NOTE: One reason why the ingress border leaf device also forwards a copy on the SBD is to ensure that the other border leaf device can receive external source traffic if its M-VLAN interface is down. Then any interested local receivers on the other border leaf device can still get the traffic.

- BL-1 also selectively forwards copies to the server leaf devices with interested receivers, based on the advertised Type 6 routes. In this case, Leaf-1 and Leaf-2 have a multihomed interested receiver, Rcvr-1, on VLAN-2 (the red VLAN).

See the green arrows from BL-1 toward Leaf-1 and Leaf-2.

NOTE: In a similar use case, BL-1 can receive the external multicast source traffic, but BL-2 is the PIM DR on the SBD. In that case, the green arrows you see in [Figure 81 on page 794](#) flow from BL-2 (instead of BL-1) toward the other EVPN devices.

- Leaf-1 and Leaf-2 locally route the traffic from the SBD IRB interface to the revenue bridge domain IRB interface for VLAN-2 (the red VLAN). However, only the EVPN DF in the ES forwards the traffic toward Rcvr-1. In this case, Leaf-1 is the EVPN DF.

Local Receivers on Border Leaf Devices

[Figure 81 on page 794](#) doesn't show local receivers attached to the border leaf devices. However, let's look briefly at PIM join message flow and how external source traffic reaches local receivers on border leaf devices.

Consider that BL-1 or BL-2 has an interested receiver on a revenue bridge domain (VLAN) in the data center. In that case, both devices generate a PIM join on the IRB interfaces for the *revenue bridge domain* toward the PIM RP.

You configure the border leaf devices with PIM in distributed DR mode on the revenue bridge domain (VLAN) IRB interfaces. That way, neither BL-1 nor BL-2 acts as the PIM DR alone. Both devices can locally route external multicast source traffic coming in on the M-VLAN IRB interface to the appropriate revenue bridge domain (VLAN) IRB interface.

Latency and Scaling Trade-Offs for Installing Multicast Routes with OISM (install-star-g-routes Option)

Devices in an OISM-enabled fabric send EVPN Type 6 routes so other EVPN devices learn about receivers that are interested in the traffic for a multicast group. The receivers are in different OISM revenue bridge domains (VLANs) in a Layer 3 virtual routing and forwarding (VRF) instance. To save bandwidth in the EVPN fabric core, OISM devices send and receive the Type 6 routes only on the OISM SBD in the routing instance.

To help minimize packet loss at the onset of a multicast flow, you can configure the global `install-star-g-routes` option at the `[edit multicast-snooping-options oism]` hierarchy level (see ["oism \(Multicast Snooping Options\)" on page 1636](#)). When you configure this option, upon receiving a Type 6 route, the Routing Engine on the device immediately installs corresponding (*,G) multicast routes on the Packet Forwarding Engine for all of the revenue VLANs in the routing instance.

With this option, you trade off taking up extra Packet Forwarding Engine resources to improve network latency. Lower-scale deployments might have fewer multicast flows but have strict network latency requirements. To improve network latency in that case, the device installs the (*,G) routes in the data plane in advance of any incoming multicast traffic.

NOTE: Always configure this option with OISM on switches in the QFX10000 line. These switches easily operate at scale and multicast traffic flows well with this option's behavior. Without this option, QFX10000 switches have an issue with mesh group handling, so the Routing Engine doesn't properly install the multicast routes.

We don't recommend setting this option with OISM on switches in the QFX5000 line. Consider this option on those switches only if you have very stringent latency requirements and can trade off higher scaling to achieve better network latency.

Default Behavior without the install-star-g-routes Option

By default, without this option, the device prioritizes saving resources on the Packet Forwarding Engine by not installing multicast routes until the multicast traffic arrives. In this default case:

1. The Packet Forwarding Engine receives multicast traffic from source S for multicast group G.
2. The Packet Forwarding Engine doesn't have forwarding next-hop information for the traffic, so it signals the Routing Engine to get that information.

NOTE: The Packet Forwarding Engine drops multicast traffic until it gets the routing information.

3. The Routing Engine learns about the multicast flow for (S,G) from the Packet Forwarding Engine, and installs that route on the Packet Forwarding Engine.
4. The Packet Forwarding Engine sends the traffic on the next hop in the installed (S,G) route.

Behavior with the install-star-g-routes Option

With the `install-star-g-routes` option, the device prioritizes having multicast routing information available on the Packet Forwarding Engine before any traffic arrives. The device consumes extra Packet Forwarding Engine resources for routes that it isn't using yet (and might never be used). With this option:

1. The Routing Engine receives an EVPN Type 6 route for a receiver subscribing to traffic for a multicast group G on the OISM SBD in a routing instance.

2. The Routing Engine installs corresponding (*,G) routes on the Packet Forwarding Engine for all of the revenue VLANs in the routing instance.
3. At some later time, the Packet Forwarding Engine receives multicast traffic from source S for multicast group G.
4. The Packet Forwarding Engine has forwarding next hop information for traffic for (*,G). So it forwards the traffic to receivers on any revenue VLANs using the (*,G) route next hop.
5. The Packet Forwarding Engine also signals the Routing Engine that it has received multicast traffic from source S for multicast group G.
6. The Routing Engine learns about the multicast flow for (S,G) from the Packet Forwarding Engine. The Routing Engine installs the (S,G) route on the Packet Forwarding Engine.
7. The Packet Forwarding Engine continues sending the traffic, but now uses the (S,G) route and the next hop in that more specific route.

NOTE: The Packet Forwarding Engine still retains the (*,G) routes per revenue VLAN that the Routing Engine installed after receiving the Type 6 route.

Configure Common OISM Elements on Border Leaf Devices and Server Leaf Devices (Symmetric Bridge Domains)

Follow these steps to configure elements common to border leaf and server leaf devices in an EVPN-VXLAN fabric running optimized inter-subnet multicast (OISM) with the symmetric bridge domains model. This configuration assumes an EVPN-VXLAN fabric configuration that supports OISM and has:

- An EBGp underlay with Bidirectional Forwarding Detection (BFD) and Ethernet operations, administration and management (OAM) link detection.
- An edge-routed bridging overlay design.
- Lean spine devices that act only as IP transit nodes in the fabric.
- Server leaf devices configured as EVPN-VXLAN Layer 2 gateways.
- Border leaf devices configured as EVPN-VXLAN Layer 2 and Layer 3 gateways.

In the symmetric bridge domains model, you must configure all of the revenue bridge domains (VLANs) and the supplemental bridge domain (SBD) on all OISM devices in the fabric. See ["Configuration Elements for OISM Devices \(Symmetric Bridge Domains\)" on page 781](#) for a summary of what elements to configure on border leaf and server leaf devices, and why you configure those elements.

The sample configuration blocks that we provide for these configuration steps use an OISM environment with the following elements:

- M-VLAN: VLAN-900
M-VLAN IRB interface: irb.900
- SBD: VLAN-300
SBD IRB interface: irb.300
- Revenue bridge domains: VLAN-100 and VLAN-200
Revenue bridge domain IRB interfaces: irb.100 and irb.200
- Layer 3 VRF routing instance: L3VRF

Configure these OISM statements on both border leaf devices and server leaf devices:

1. Enable OISM globally on the device.

Configure the `evpn irb oism` option at the `[edit forwarding-options multicast-replication]` hierarchy level. This option enables OISM optimizations and external multicast capabilities in an edge-routed bridging (ERB) overlay fabric. This option takes the place of the `evpn irb local-only` option you would otherwise use in ERB overlay fabrics for multicast without OISM.

```
set forwarding-options multicast-replication evpn irb oism
```

2. Configure all of the revenue bridge domains (VLANs) with an IRB interface and a VXLAN network identifier (VNI) mapping for each revenue bridge domain. For example, if the revenue bridge domains are VLAN-100 and VLAN-200:

```
set vlans VLAN-100 vlan-id 100
set vlans VLAN-100 l3-interface irb.100
set vlans VLAN-100 vxlan vni 100

set vlans VLAN-200 vlan-id 200
set vlans VLAN-200 l3-interface irb.200
set vlans VLAN-200 vxlan vni 200
```

3. Configure a VLAN for the SBD with an IRB interface and VXLAN VNI mapping. For example, for SBD VLAN-300:

```
set vlans VLAN-300 vlan-id 300
set vlans VLAN-300 l3-interface irb.300
set vlans VLAN-300 vxlan vni 300
```

4. Configure the revenue bridge domain and the SBD IRB interface IP addresses. For example:

```
set interfaces irb unit 100 family inet address 10.0.100.1/24
set interfaces irb unit 200 family inet address 10.0.200.1/24
set interfaces irb unit 300 family inet address 10.0.300.1/24
```

5. Extend the revenue bridge domains and the SBD VNIs in the EVPN-VXLAN overlay. For example:

```
set protocols evpn extended-vni-list 100
set protocols evpn extended-vni-list 200
set protocols evpn extended-vni-list 300
```

Prior to this step, we assume you have set up the EVPN fabric with VXLAN encapsulation by configuring the `vxlan` statement at the `[edit protocols evpn encapsulation]` hierarchy level.

6. Enable IGMP snooping in proxy mode on all the configured VLANs using the statement. This statement also automatically enables advertising SMET Type 6 routes in the EVPN core based on received IGMP reports.

```
set protocols igmp-snooping vlan all proxy
```

7. Configure a Layer 3 VRF routing instance ("[instance-type](#)" on [page 1589](#)`vrf`) that you associate with OISM routing functions. Include the revenue bridge domain IRB interfaces and the SBD IRB interface in the routing instance. For example:

```
set routing-instances L3VRF interface irb.100
set routing-instances L3VRF interface irb.200
set routing-instances L3VRF interface irb.300
set routing-instances L3VRF interface lo0.1
set routing-instances L3VRF route-distinguisher 10.255.0.1:100
set routing-instances L3VRF vrf-target target:100:100
set routing-instances L3VRF instance-type vrf
```

8. In the Layer 3 VRF routing instance, specify the VLAN and its IRB interface that acts as the OISM SBD. For example:

```
set routing-instances L3VRF protocols evpn oism supplemental-bridge-domain-irb irb.300
```

9. (Required on the QFX10000 line of switches; optional but not recommended on the QFX5000 line of switches) To avoid traffic loss at the onset of multicast flows, enable the `install-star-g-routes` option at the `[edit multicast-snooping-options oism]` hierarchy level on all OISM devices. With this option, upon receiving an EVPN Type 6 route on the SBD, the device immediately installs corresponding (*,G) routes on the Packet Forwarding Engine for the revenue bridge domain VLANs in the Layer 3 VRF instance. For full details on OISM device requirements, recommendations, and behavior with or without this option, see ["Latency and Scaling Trade-Offs for Installing Multicast Routes with OISM \(install-star-g-routes Option\)" on page 796](#).

```
set multicast-snooping-options oism install-star-g-routes
```

You might choose not to enable this option if space on the Packet Forwarding Engine is at a premium and you don't have stringent latency requirements.

SEE ALSO

[oism | 1634](#)

[oism \(Multicast Snooping Options\) | 1636](#)

Configure Border Leaf Device OISM Elements (Symmetric Bridge Domains)

First configure the optimized inter-subnet multicast (OISM) elements described in ["Configure Common OISM Elements on Border Leaf Devices and Server Leaf Devices \(Symmetric Bridge Domains\)" on page 798](#) in an EVPN-VXLAN fabric.

Then follow these steps to configure the required OISM elements on the border leaf devices. The same EVPN-VXLAN fabric base and sample OISM environment apply to the additional border leaf configuration steps here.

See ["Configuration Elements for OISM Devices \(Symmetric Bridge Domains\)" on page 781](#) for more information about why OISM border leaf devices require these settings.

1. Configure the M-VLAN similarly to how you configure the revenue bridge domains and the SBD.

- a. Configure the M-VLAN with an IRB interface and a VXLAN network identifier (VNI) mapping. For example, if the M-VLAN is VLAN-900:

```
set vlans VLAN-900 vlan-id 900
set vlans VLAN-900 l3-interface irb.900
set vlans VLAN-900 vxlan vni 900
```

- b. Configure the M-VLAN IRB interface IP address. For example:

```
set interfaces irb unit 900 family inet address 10.0.900.1/24
```

- c. Extend the M-VLAN in the EVPN-VXLAN overlay. For example:

```
set protocols evpn extended-vni-list 900
```

2. In the common leaf device configuration steps, you enable IGMP snooping in proxy mode for all VLANs. On border leaf devices, also configure the *multicast-router-interface* option on the Layer 2 ports that connect to the external multicast PIM router. For example, if xe-0/0/4.0 is the Layer 2 interface that connects the EVPN fabric to the external multicast router on the M-VLAN:

```
set protocols igmp-snooping vlan VLAN-900 interface xe-0/0/4.0 multicast-router-interface
```

3. In the common leaf device configuration steps, you configure a Layer 3 VRF routing instance associated with OISM routing functions. On the border leaf devices, also include the M-VLAN IRB interfaces in that Layer 3 VRF. The following configuration block shows the common Layer 3 VRF configuration with the additional M-VLAN IRB interface configuration statement highlighted:

```
set routing-instances L3VRF interface irb.100
set routing-instances L3VRF interface irb.200
set routing-instances L3VRF interface irb.300
set routing-instances L3VRF interface irb.900
set routing-instances L3VRF interface lo0.1
set routing-instances L3VRF route-distinguisher 10.255.0.1:100
set routing-instances L3VRF vrf-target target:100:100
set routing-instances L3VRF instance-type vrf
```

4. In the Layer 3 VRF routing instance, set the OISM PIM EVPN gateway (PEG) role on the M-VLAN IRB interface. For example:

```
set routing-instances L3VRF protocols evpn oism pim-evpn-gateway external-irb irb.900
```

5. Configure PIM in the Layer 3 VRF routing instance for a border leaf device.
 - a. In the Layer 3 VRF routing instance, configure PIM in distributed DR mode on the revenue bridge domain IRB interfaces using the statement at the [edit routing-instances *name* protocols pim interface *irb-interface-name*] hierarchy level. In this step, you configure PIM in standard mode on the SBD IRB interface and the M-VLAN IRB interface. You also configure a PIM static RP that corresponds to the external PIM RP router. In the sample use cases in this documentation, the external PIM router serves as the PIM RP. For example, if the revenue bridge domains are VLAN-100 and VLAN-200, the SBD is VLAN-300, and the M-VLAN is VLAN-900:

```
set routing-instances L3VRF protocols pim rp static address external-PIM-RP-IP-address
set routing-instances L3VRF protocols pim interface irb.100 distributed-dr
set routing-instances L3VRF protocols pim interface irb.201 distributed-dr
set routing-instances L3VRF protocols pim interface irb.300
set routing-instances L3VRF protocols pim interface irb.900
set routing-instances L3VRF protocols pim interface xe-0/0/6.0
set routing-instances L3VRF protocols pim interface lo0.1
```

- b. In the Layer 3 VRF routing instance, set the option at the [edit routing-instances *name* protocols pim interface *irb-interface-name*] hierarchy level on the M-VLAN IRB interface. Configure a policy option along with this statement so that the device always installs PIM joins from the external PIM router. See ["Configuration Elements for OISM Devices \(Symmetric Bridge Domains\)" on page 781](#) for more information about why OISM border leaf devices require this configuration. This sample configuration block represents the external PIM router as an MX Series router. For the policy prefix list, include the IP address of the M-VLAN interface on the MX Series router that connects to the border leaf device. For example:

```
set routing-instances L3VRF protocols pim interface irb.900 accept-join-always-from
ajaf-900
set policy-options prefix-list mx-900 192.168.1.9/32
set policy-options policy-statement ajaf-900 term term1 from prefix-list mx-900
set policy-options policy-statement ajaf-900 term term1 then accept
set policy-options policy-statement ajaf-900 then reject
```

Configure Server Leaf Device OISM Elements (Symmetric Bridge Domains)

First configure the OISM elements described in ["Configure Common OISM Elements on Border Leaf Devices and Server Leaf Devices \(Symmetric Bridge Domains\)" on page 798](#) in an EVPN-VXLAN fabric.

Then follow these steps to configure the additional required OISM elements on the server leaf devices. The same EVPN-VXLAN fabric base and sample OISM environment apply to the additional server leaf configuration steps here.

See ["Configuration Elements for OISM Devices \(Symmetric Bridge Domains\)" on page 781](#) for more information about why OISM server leaf devices require these settings.

1. Configure PIM passive mode on server leaf devices. For example:

```
set routing-instances L3VRF protocols pim passive
set routing-instances L3VRF protocols pim interface all
```

2. Enable the server leaf device to accept multicast traffic from the SBD IRB interface as the source interface using the statement at the [edit routing-instances *name* protocols pim interface *irb-interface-name*] hierarchy level. For example, for our sample SBD, VLAN-300, the IRB interface is irb.300:

```
set routing-instances L3VRF protocols pim interface irb.300 accept-remote-source
```

CLI Commands to Verify the OISM Configuration

To verify your OISM configuration:

1. Use the ["show evpn oism" on page 1894](#) command to view the SBD IRB interface in each Layer 3 VRF routing instance that you configured on the device for OISM. For example:

```
user@Leaf-1> show evpn oism
L3 context          SBD
L3VRF-1             irb.300
L3VRF-2             irb.400
```

To view information for a specific routing instance or to see more details about the configuration, use the extensive option with this show command. For example:

```
user@Leaf-1> show evpn oism l3-context L3VRF-1 extensive
EVPN L3 context: L3VRF-1
OISM SBD interface: irb.300
```

2. Use the `show route table bgp.evpn.0 ...extensive` command to see the OISM capabilities you have enabled on an OISM leaf device. These capabilities are in the EVPN multicast flags extended community in EVPN Type 3 routes, which are also called inclusive multicast Ethernet tag (IMET) routes. The OISM-related flags include:
 - `igmp-snooping`—You enabled IGMP snooping. (The flags value for IGMP snooping without OISM or PEG configuration = 0x1.)
 - `oism`—You globally enabled OISM. (The flags value for OISM and IGMP snooping without PEG configuration = 0x9.)
 - `peg`—You configured PEG mode on the interface. (The flags value for OISM and IGMP snooping with PEG configuration = 0x19.)

For example, a route advertised for an M-VLAN IRB interface on a border leaf device has the `peg` flag set:

```
user@BL-1> show route table bgp.evpn.0 match-prefix 3:<...> extensive
...
          Communities: target:1:1 encapsulation:vxlan(0x8) evpn-mcast-flags:0x19:igmp-
snooping-enabled:oism:peg
...
```

You don't see the `peg` flag set for a revenue bridge domain interface on a border leaf device or a server leaf device:

```
user@Leaf-1> show route table bgp.evpn.0 match-prefix 3:<...> extensive
...
          Communities: target:1:1 encapsulation:vxlan(0x8) evpn-mcast-flags:0x9:igmp-
snooping-enabled:oism
...
```

3. Enter the `show igmp snooping evpn status <vlan name><detail>` command to see the EVPN-VXLAN Layer 2 multicast context for the OISM bridge domains (VLANs) on the device. The default output includes the VXLAN VNI mappings of the VLANs. For example:

```
user@Leaf-1> show igmp snooping evpn status
Instance: default-switch
  Bridge-Domain: VLAN-100, VN Identifier: 100
    OISM: Enabled
  Bridge-Domain: VLAN-200, VN Identifier: 200
    OISM: Enabled
```



```
Bridge-Domain: VLAN-300, VN Identifier: 300
OISM: Enabled
```

The detail option also displays whether a VLAN is the OISM SBD (Supplementary BD) or the M-VLAN (External VLAN). Both of those output fields display No for revenue bridge domains (VLANs) or if you don't enable OISM.

Here is an example on a server leaf device in a fabric where the SBD is VLAN-300:

```
user@Leaf-1> show igmp snooping evpn status vlan VLAN-300 detail
Instance: default-switch
  Bridge-Domain: VLAN-300, VN Identifier: 300
    OISM          : Enabled
    Supplementary BD: Yes
    External VLAN  : No
```

Here is another example on a border leaf device in a fabric where the M-VLAN is VLAN-900:

```
user@Leaf-1> show igmp snooping evpn status vlan VLAN-900 detail
Instance: default-switch
  Bridge-Domain: VLAN-900, VN Identifier: 900
    OISM          : Enabled
    Supplementary BD: No
    External VLAN  : Yes
```

4. Other show commands display multicast information that isn't specific to OISM elements but can help check configured multicast options and traffic flow.

For example, use the `show evpn multicast-snooping status` to see if you enabled IGMP snooping on VLANs you configured for OISM elements.

In the following sample command, if the revenue bridge domains on a server leaf device are VLAN-100 and VLAN-200, the SBD is VLAN-300, and you enabled IGMP snooping (but not MLD snooping):

```
user@Leaf-1> show evpn multicast-snooping status
Instance: __default_evpn__

Instance: default-switch
  VLAN ID: 100
    Multicast Address Family: INET
    SG Sync: Enabled
    Multicast Address Family: INET6
```

```
SG Sync: Disabled
```

```
VLAN ID: 200
```

```
Multicast Address Family: INET
```

```
SG Sync: Enabled
```

```
Multicast Address Family: INET6
```

```
SG Sync: Disabled
```

```
VLAN ID: 300
```

```
Multicast Address Family: INET
```

```
SG Sync: Enabled
```

```
Multicast Address Family: INET6
```

```
SG Sync: Disabled
```

- `show evpn multicast-snooping status`—See if you enabled IGMP snooping on VLANs you configured for OISM elements.

In the following case, if the revenue bridge domains on a server leaf device are VLAN-100 and VLAN-200, the SBD is VLAN-300, and you enabled IGMP snooping (but not MLD snooping):

```
user@Leaf-1> show evpn multicast-snooping status
```

```
Instance: __default_evpn__
```

```
Instance: default-switch
```

```
VLAN ID: 100
```

```
Multicast Address Family: INET
```

```
SG Sync: Enabled
```

```
Multicast Address Family: INET6
```

```
SG Sync: Disabled
```

```
VLAN ID: 200
```

```
Multicast Address Family: INET
```

```
SG Sync: Enabled
```

```
Multicast Address Family: INET6
```

```
SG Sync: Disabled
```

```
VLAN ID: 300
```

```
Multicast Address Family: INET
```

```
SG Sync: Enabled
```

```
Multicast Address Family: INET6
```

```
SG Sync: Disabled
```

SEE ALSO

[Multicast Support in EVPN-VXLAN Overlay Networks | 668](#)

[Overview of Multicast Forwarding with IGMP Snooping or MLD Snooping in an EVPN-VXLAN Environment | 675](#)

[Overview of Selective Multicast Forwarding | 747](#)

[Example: Configuring a QFX5110 Switch as Layer 2 and 3 VXLAN Gateways in an EVPN-VXLAN Edge-Routed Bridging Overlay | 630](#)

Configuring the Tunneling of Q-in-Q Traffic

IN THIS CHAPTER

- [Examples: Tunneling Q-in-Q Traffic in an EVPN-VXLAN Overlay Network | 809](#)

Examples: Tunneling Q-in-Q Traffic in an EVPN-VXLAN Overlay Network

IN THIS SECTION

- [Requirements | 811](#)
- [Overview and Topology | 811](#)
- [Configuring Traffic Pattern 1: Popping an S-VLAN Tag | 815](#)
- [Configuring Traffic Pattern 2: Mapping a Range of C-VLANs to an S-VLAN, and Pushing an S-VLAN Tag | 820](#)
- [Configuring Traffic Pattern 3: Retaining S-VLAN and C-VLAN Tags | 825](#)
- [Configuring Traffic Pattern 4: Popping and Later Pushing an S-VLAN Tag | 828](#)

The tunneling of Q-in-Q packets in an Ethernet VPN-Virtual Extensible LAN (EVPN-VXLAN) overlay network is supported as follows:

- Starting with Junos OS Release 17.2R1, QFX5100 switches that function as Layer 2 VXLAN tunnel endpoints (VTEPs) can tunnel single- and double-tagged Q-in-Q packets in an EVPN-VXLAN centrally-routed bridging overlay (EVPN-VXLAN network with a two-layer IP fabric).
- Starting with Junos OS Release 18.2R1, QFX5110, QFX5200, and EX4600 switches that function as Layer 2 VTEPs can tunnel single- and double-tagged Q-in-Q packets in a centrally-routed bridging overlay.

- Starting with Junos OS Release 18.3R1, the QFX10000 line of switches that function as Layer 2 VTEPs only can tunnel single- and double-tagged Q-in-Q packets in a centrally-routed bridging overlay.

In addition to tunneling Q-in-Q packets, the ingress and egress VTEPs can perform the following Q-in-Q actions:

- Delete, or pop, an outer service VLAN (S-VLAN) tag from an incoming packet.
- Add, or push, an outer S-VLAN tag onto an outgoing packet.
- Map a configured range of customer VLAN (C-VLAN) IDs to an S-VLAN.

NOTE: The QFX Series and EX4600 switches support the pop and push actions only with a specified VLAN. The switches do not support the pop and push actions with a configured range of VLANs.

The ingress and egress VTEPs support the tunneling of Q-in-Q packets and the Q-in-Q actions in the context of the traffic patterns described in this topic.

NOTE: This topic describes and shows how to configure the VXLAN tunneling of Q-in-Q packets for each traffic pattern. One or more of the traffic patterns might apply to your environment. Perform only those configurations that apply to your environment.

The ingress and egress VTEPs can also map a single- or double-tagged packet to a specified VLAN or to any VLAN specified in a configured list, and further map the VLAN to a VXLAN network identifier (VNI).

To enable the tunneling of Q-in-Q packets, you must configure a flexible VLAN tagging interface that can transmit 802.1Q VLAN single- and double-tagged packets on ingress and egress VTEPs.

Also, it is important that Q-in-Q packets retain the inner C-VLAN tag while they are tunneled between ingress and egress VTEPs. Therefore, on each VTEP, you must include the `encapsulate-inner-vlan` configuration statement, which retains the inner tag during packet encapsulation, at the `[edit vlans vlan-name vxlan]` hierarchy level and the `decapsulate-accept-inner-vlan` configuration statement, which retains the inner tag during packet de-encapsulation, at the `[edit protocols l2-learning]` hierarchy level.

NOTE: To configure a QFX10000 switch to retain the inner C-VLAN tag while it tunnels Q-in-Q packets, you must include only the `encapsulate-inner-vlan` configuration statement at the `[edit vlans`

vlan-name vxlan] hierarchy level. You do not need to include the `decapsulate-accept-inner-vlan` configuration statement at the `[edit protocols l2-learning]` hierarchy level.

This topic includes the following sections:

Requirements

These examples use the following hardware and software components:

- Two QFX5100 switches. One switch functions as the ingress VTEP; and the other as the egress VTEP.
- Junos OS Release 17.2R1 or later.

Overview and Topology

IN THIS SECTION

- [Understanding Traffic Pattern 1: Popping an S-VLAN Tag | 812](#)
- [Understanding Traffic Pattern 2: Mapping a Range of C-VLANs to an S-VLAN, and Pushing an S-VLAN Tag | 813](#)
- [Understanding Traffic Pattern 3: Retaining S-VLAN and C-VLAN Tags | 813](#)
- [Understanding Traffic Pattern 4: Popping and Later Pushing an S-VLAN Tag | 815](#)

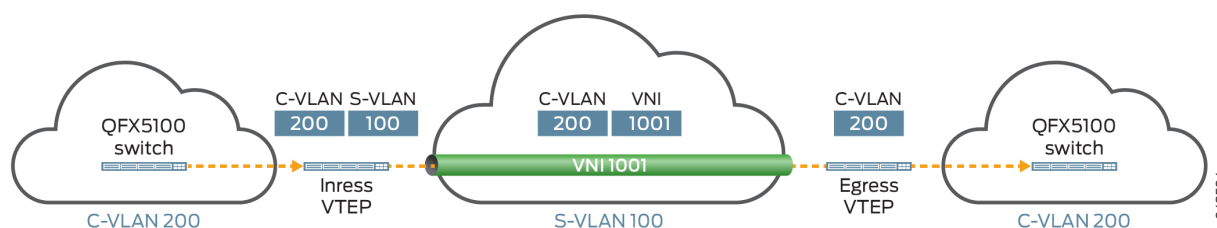
This section describes the traffic patterns in which the VXLAN tunneling of Q-in-Q traffic is supported in a EVPN-VXLAN overlay network.

NOTE: This topic describes and shows how to configure the VXLAN tunneling of Q-in-Q packets for each traffic pattern. One or more of the traffic patterns might apply to your environment. Perform only those configurations that apply to your environment.

Understanding Traffic Pattern 1: Popping an S-VLAN Tag

Figure 82 on page 812 shows Q-in-Q traffic flowing from one dispersed C-VLAN 200 site to another by way of S-VLAN 100.

Figure 82: Popping an S-VLAN Tag



When a packet flows from C-VLAN 200 to S-VLAN 100 to C-VLAN 200, the ingress VTEP:

- Receives a packet with two tags—an inner C-VLAN tag of 200 and an outer S-VLAN tag of 100.
- Takes note of S-VLAN tag 100, which is mapped to VNI 1001, and then pops the tag.
- Encapsulates the packet with a VXLAN header that includes VNI 1001, and sends the packet with inner C-VLAN tag 200 and the VXLAN header.

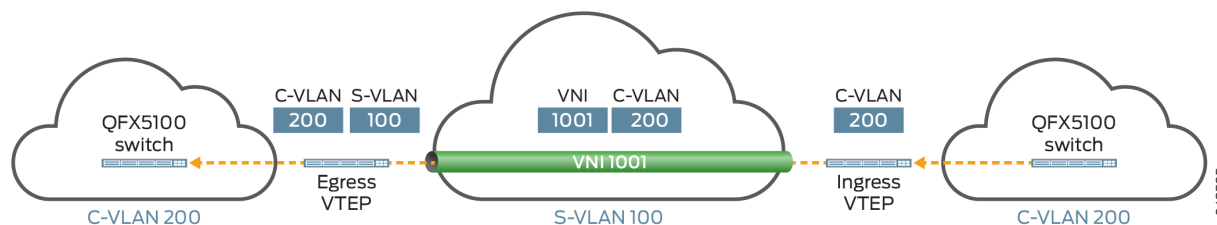
After the packet is tunneled over the Layer 3 underlay network, the egress VTEP:

- Removes the VXLAN header from the packet.
- Maps VNI 1001 back to S-VLAN 100.
- Sends the packet with C-VLAN tag 200.

Understanding Traffic Pattern 2: Mapping a Range of C-VLANs to an S-VLAN, and Pushing an S-VLAN Tag

Figure 83 on page 813 shows Q-in-Q traffic flowing from one dispersed C-VLAN 200 site to another by way of S-VLAN 100.

Figure 83: Mapping a Range of C-VLANs to an S-VLAN



When a single-tagged packet flows from C-VLAN 200 to S-VLAN 100 to C-VLAN 200, the ingress VTEP:

- Receives a packet with a C-VLAN tag of 200.
- Takes note of C-VLAN tag 200, which is in a configured VLAN ID range 100 through 200 that is mapped to S-VLAN 100 and VNI 1001.
- Encapsulates the packet with a VXLAN header that includes VNI 1001 and sends the packet with C-VLAN tag 200 and VNI 1001.

After the packet is tunneled over the Layer 3 underlay network, the egress VTEP:

- De-encapsulates the packet.
- Maps the packet to S-VLAN 100 through its association with VNI 1001.
- Pushes S-VLAN tag 100 on the packet, and sends the packet with inner C-VLAN tag 200 and outer S-VLAN tag 100.

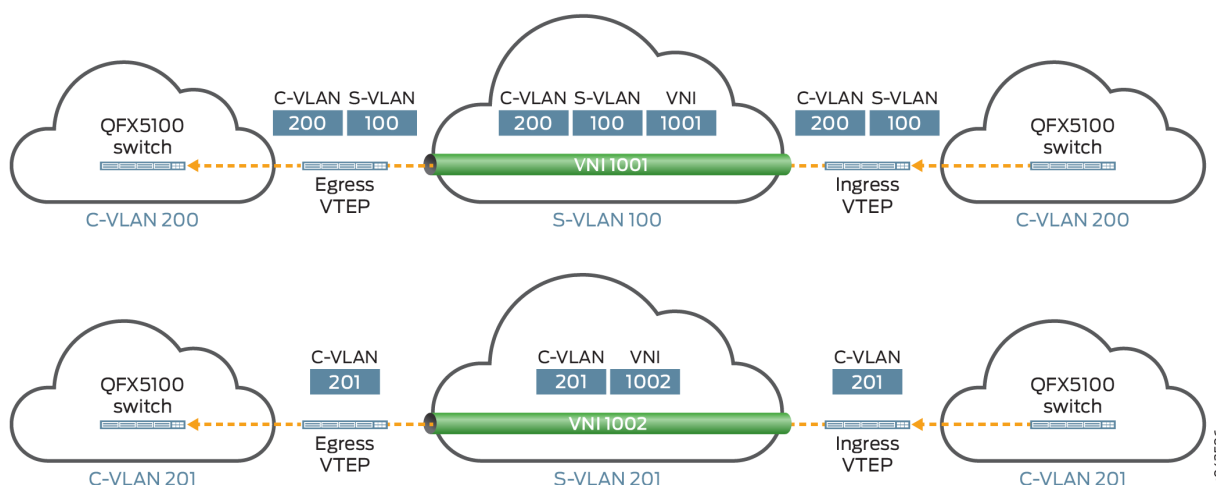
Understanding Traffic Pattern 3: Retaining S-VLAN and C-VLAN Tags

Figure 84 on page 814 shows the following Q-in-Q traffic flows:

- Double-tagged packets from C-VLAN 200 to S-VLAN 100 to C-VLAN 200.

- Single-tagged packets from C-VLAN 201 to S-VLAN 201 to C-VLAN 201.

Figure 84: Retaining S-VLAN and C-VLAN Tags



When a packet flows from either C-VLAN 200 or C-VLAN 201, the ingress VTEP:

- Receives a packet—either a double-tagged packet with an inner C-VLAN tag of 200 and an outer S-VLAN tag of 100 or a single-tagged packet with a C-VLAN tag of 201.
- Takes note of outer S-VLAN tag 100, which is mapped to VNI 1001, for the double-tagged packet. For the single-tagged packet, the ingress VTEP takes note of C-VLAN tag 201, which is mapped to VNI 1002.
- Encapsulates the packet with a VXLAN header that includes VNI 1001 for the double-tagged packet and VNI 1002 for the single-tagged packet. In addition to the VXLAN header, the ingress VTEP sends the double-tagged packet with inner C-VLAN tag 200 and outer S-VLAN tag 100, and the single-tagged packet with C-VLAN tag 201.

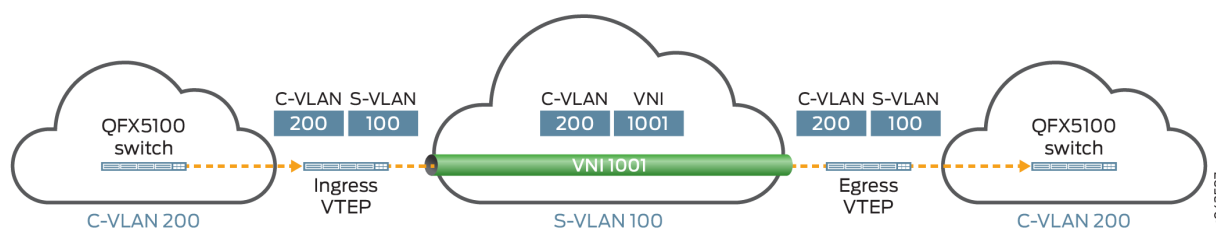
After the packet is tunneled over the Layer 3 underlay network, the egress VTEP:

- Removes the VXLAN header from the packet.
- For the double-tagged packet, maps VNI 1001 back to S-VLAN 100, and for the single-tagged packet, maps VNI 1002 back to C-VLAN 201.
- Sends the double-tagged packet with inner C-VLAN tag 200 and outer S-VLAN tag 100 and the single-tagged packet with C-VLAN tag 201.

Understanding Traffic Pattern 4: Popping and Later Pushing an S-VLAN Tag

Figure 85 on page 815 shows Q-in-Q traffic flowing from one dispersed C-VLAN 200 site to another by way of S-VLAN 100.

Figure 85: Popping and Later Pushing an S-VLAN Tag



When a packet flows from C-VLAN 200 to S-VLAN 100 to C-VLAN 200, the ingress VTEP:

- Receives a packet with two tags—an inner C-VLAN tag of 200 and an outer S-VLAN tag of 100.
- Takes note of S-VLAN tag 100, which is mapped to VNI 1001, then pops the tag.
- Encapsulates the packet with a VXLAN header that includes VNI 1001 and sends the packet with inner C-VLAN tag 200 and the VXLAN header.

After the packet is tunneled over the Layer 3 underlay network, the egress VTEP:

- De-encapsulates the packet.
- Maps the packet back to S-VLAN 100 through its association with VNI 1001.
- Pushes S-VLAN tag 100 on the packet, and sends the packet with inner C-VLAN tag 200 and outer S-VLAN tag 100.

Configuring Traffic Pattern 1: Popping an S-VLAN Tag

IN THIS SECTION

- [Requirements | 816](#)
- [Introduction | 816](#)
- [Ingress VTEP Configuration for Traffic Pattern 1 | 816](#)
- [Egress VTEP Configuration for Traffic Pattern 1 | 818](#)

Requirements

Introduction

For this traffic pattern, the ingress and egress VTEPs in an EVPN-VXLAN overlay network must handle double-tagged Q-in-Q traffic. The ingress VTEP retains the inner C-VLAN tag and removes, or pops, the outer S-VLAN tag. The egress VTEP also retains the inner C-VLAN tag but does not reinstate the outer S-VLAN tag.

NOTE: QFX Series and EX4600 switches support this traffic pattern on both aggregated Ethernet and non-aggregated Ethernet interfaces.

NOTE: This configuration focuses on traffic pattern 1 only. It does not provide the configuration for EVPN and all aspects of VXLAN. For a more comprehensive EVPN-VXLAN configuration for a centrally-routed bridging overlay, see ["Example: Configure an EVPN-VXLAN Centrally-Routed Bridging \(CRB\) Overlay" on page 530](#).

Ingress VTEP Configuration for Traffic Pattern 1

IN THIS SECTION

- [CLI Quick Configuration | 816](#)
- [Procedure | 817](#)

CLI Quick Configuration

To quickly configure the ingress VTEP, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

NOTE: To configure a QFX10000 switch to retain the inner C-VLAN tag while it tunnels Q-in-Q packets, you must include only the `encapsulate-inner-vlan` configuration statement at the [edit vlans

`vlan-name vxlan]` hierarchy level. You do not need to include the `decapsulate-accept-inner-vlan` configuration statement at the `[edit protocols l2-learning]` hierarchy level.

```
set protocols l2-learning decapsulate-accept-inner-vlan
set interfaces xe-0/0/0 flexible-vlan-tagging
set interfaces xe-0/0/0 encapsulation extended-vlan-bridge
set interfaces xe-0/0/0 unit 100 vlan-id 100
set interfaces xe-0/0/0 unit 100 input-vlan-map pop
set vlans vlan_1 interface xe-0/0/0.100
set vlans vlan_1 vxlan vni 1001
set vlans vlan_1 vxlan encapsulate-inner-vlan
```

Procedure

Step-by-Step Procedure

To configure the ingress VTEP for traffic pattern 1:

1. On all supported Juniper Networks switches except the QFX10000 line of switches, configure the VTEP to retain the inner C-VLAN tag while de-encapsulating a packet.

```
[edit protocols l2-learning]
user@switch# set decapsulate-accept-inner-vlan
```

NOTE: To support the tunneling of Q-in-Q packets, you must configure both ingress and egress VTEP to retain the inner C-VLAN tag while de-encapsulating a packet.

2. Configure the physical interface to support the simultaneous transmission of 802.1Q VLAN single-tagged and double-tagged packets on its logical interfaces and to accept packets carrying Tag Protocol Identifier (TPID) 0x8100.

```
[edit interfaces]
user@switch# set xe-0/0/0 flexible-vlan-tagging
user@switch# set xe-0/0/0 encapsulation extended-vlan-bridge
```

3. On physical interface xe-0/0/0, create logical interface 100, and associate it with S-VLAN 100. Also, assuming that the ingress VTEP receives a double-tagged packet as described in this traffic pattern, specify that the outer S-VLAN tag is popped on incoming packets.

```
[edit interfaces]
user@switch# set xe-0/0/0 unit 100 vlan-id 100
user@switch# set xe-0/0/0 unit 100 input-vlan-map pop
```

4. Create a VLAN named vlan_1, and map it to logical interface xe-0/0/0.100 and VNI 1001. Also specify that the logical interface retains the inner C-VLAN tag while encapsulating a packet.

```
[edit vlans]
user@switch# set vlan_1 interface xe-0/0/0.100
user@switch# set vlan_1 vxlan vni 1001
user@switch# set vlan_1 vxlan encapsulate-inner-vlan
```

Egress VTEP Configuration for Traffic Pattern 1

IN THIS SECTION

- [CLI Quick Configuration | 818](#)
- [Procedure | 819](#)

CLI Quick Configuration

To quickly configure the egress VTEP, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

NOTE: To configure a QFX10000 switch to retain the inner C-VLAN tag while it tunnels Q-in-Q packets, you must include only the encapsulate-inner-vlan configuration statement at the [edit vlans

vlan-name vxlan] hierarchy level. You do not need to include the `decapsulate-accept-inner-vlan` configuration statement at the [edit protocols l2-learning] hierarchy level.

```
set protocols l2-learning decapsulate-accept-inner-vlan
set interfaces xe-0/0/0 flexible-vlan-tagging
set interfaces xe-0/0/0 encapsulation extended-vlan-bridge
set interfaces xe-0/0/0 unit 100 vlan-id 100
set vlans vlan_1 interface xe-0/0/0.100
set vlans vlan_1 vxlan vni 1001
set vlans vlan_1 vxlan encapsulate-inner-vlan
```

Procedure

Step-by-Step Procedure

To configure the egress VTEP for traffic pattern 1:

1. On all Juniper Networks devices except the QFX10000 line of switches, configure the VTEP to retain the inner VLAN tag while de-encapsulating a packet.

```
[edit protocols l2-learning]
user@switch# set decapsulate-accept-inner-vlan
```

2. Configure the physical interface to support the simultaneous transmission of 802.1Q VLAN single-tagged and double-tagged packets on its logical interfaces and to accept packets carrying TPID 0x8100. Also, configure logical interface 100, and associate it with VLAN 100.

```
[edit interfaces]
user@switch# set xe-0/0/0 flexible-vlan-tagging
user@switch# set xe-0/0/0 encapsulation extended-vlan-bridge
user@switch# set xe-0/0/0 unit 100 vlan-id 100
```

3. Create a VLAN named `vlan_1`, and map it to logical interface `xe-0/0/0.100` and VNI 1001. Also specify that the logical interface retains the inner C-VLAN tag while encapsulating a packet.

```
[edit vlans]
user@switch# set vlan_1 interface xe-0/0/0.100
```

```
user@switch# set vlan_1 vxlan vni 1001
user@switch# set vlan_1 vxlan encapsulate-inner-vlan
```

NOTE: To support the tunneling of Q-in-Q packets, you must configure both ingress and egress VTEPs to retain the inner C-VLAN tag while encapsulating a packet.

Configuring Traffic Pattern 2: Mapping a Range of C-VLANs to an S-VLAN, and Pushing an S-VLAN Tag

IN THIS SECTION

- [Requirements | 820](#)
- [Introduction | 820](#)
- [Ingress VTEP Configuration for Traffic Pattern 2 | 821](#)
- [Egress VTEP Configuration for Traffic Pattern 2 | 823](#)

Requirements

Introduction

For this traffic pattern, the ingress VTEP in an EVPN-VXLAN overlay network receives a packet tagged with a C-VLAN ID, one of which is included in a configured range of C-VLAN IDs that are mapped to a particular S-VLAN. After the packet is tunneled over the Layer 3 network, the egress VTEP retains the C-VLAN tag and pushes an outer tag for that particular S-VLAN on the packet.

NOTE: The QFX10000 line of switches support this traffic pattern on aggregated Ethernet and non-aggregated Ethernet interfaces. The remaining QFX Series and EX4600 switches support this traffic pattern only on non-aggregated Ethernet interfaces.

NOTE: QFX Series and EX4600 switches do not support the pop and push actions with a configured range of VLANs.

Ingress VTEP Configuration for Traffic Pattern 2

IN THIS SECTION

- [CLI Quick Configuration | 821](#)
- [Procedure | 822](#)

NOTE: This configuration focuses on traffic pattern 2 only. It does not provide the configuration for EVPN and all aspects of VXLAN. For a more comprehensive EVPN-VXLAN configuration for a centrally-routed bridging overlay, see ["Example: Configure an EVPN-VXLAN Centrally-Routed Bridging \(CRB\) Overlay" on page 530](#).

CLI Quick Configuration

To quickly configure the ingress VTEP, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

NOTE: To configure a QFX10000 switch to retain the inner C-VLAN tag while it tunnels Q-in-Q packets, you must include only the encapsulate-inner-vlan configuration statement at the [edit vlans *vlan-name* vxlan] hierarchy level. You do not need to include the decapsulate-accept-inner-vlan configuration statement at the [edit protocols l2-learning] hierarchy level.

```
set protocols l2-learning decapsulate-accept-inner-vlan
set interfaces xe-0/0/5 flexible-vlan-tagging
set interfaces xe-0/0/5 encapsulation extended-vlan-bridge
set interfaces xe-0/0/5 unit 100 vlan-id-list 100-200
set vlans vlan_range1 interface xe-0/0/5.100
set vlans vlan_range1 vxlan vni 1001
set vlans vlan_range1 vxlan encapsulate-inner-vlan
```


Procedure

Step-by-Step Procedure

To configure the ingress VTEP for traffic pattern 2:

1. On all supported Juniper Networks switches except the QFX10000 line of switches, configure the VTEP to retain the inner VLAN tag while de-encapsulating a packet.

```
[edit protocols l2-learning]
user@switch# set decapsulate-accept-inner-vlan
```

NOTE: To support the tunneling of Q-in-Q packets, you must configure both ingress and egress VTEPs to retain the inner C-VLAN tag while de-encapsulating a packet.

2. Configure the physical interface to support the simultaneous transmission of 802.1Q VLAN single-tagged and double-tagged packets on its logical interfaces and to accept packets carrying TPID 0x8100. Also, for the physical interface, configure logical interface 100 and map it to C-VLANs 100 through 200.

```
[edit interfaces]
user@switch# set xe-0/0/5 flexible-vlan-tagging
user@switch# set xe-0/0/5 encapsulation extended-vlan-bridge
user@switch# set xe-0/0/5 unit 100 vlan-id-list 100-200
```

3. Create a VLAN named `vlan_range1`, and map it to logical interface 100 and VNI 1001. Also specify that the logical interface retains the inner VLAN tag while encapsulating a packet.

```
[edit vlans]
user@switch# set vlan_range1 interface xe-0/0/5.100
user@switch# set vlan_range1 vxlan vni 1001
user@switch# set vlan_range1 vxlan encapsulate-inner-vlan
```

Egress VTEP Configuration for Traffic Pattern 2

IN THIS SECTION

- [CLI Quick Configuration | 823](#)
- [Procedure | 823](#)

CLI Quick Configuration

To quickly configure the egress VTEP, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

NOTE: To configure a QFX10000 switch to retain the inner C-VLAN tag while it tunnels Q-in-Q packets, you must include only the `encapsulate-inner-vlan` configuration statement at the [edit vlans *vlan-name* vxlan] hierarchy level. You do not need to include the `decapsulate-accept-inner-vlan` configuration statement at the [edit protocols l2-learning] hierarchy level.

```
set protocols l2-learning decapsulate-accept-inner-vlan
set interfaces xe-0/0/0 flexible-vlan-tagging
set interfaces xe-0/0/0 encapsulation extended-vlan-bridge
set interfaces xe-0/0/0 unit 100 vlan-id 100
set interfaces xe-0/0/0 unit 100 input-vlan-map pop
set interfaces xe-0/0/0 unit 100 output-vlan-map push
set vlans v100 interface xe-0/0/0.100
set vlans v100 vxlan vni 1001
set vlans v100 vxlan encapsulate-inner-vlan
```

Procedure

Step-by-Step Procedure

To configure the egress VTEP for traffic pattern 2:

1. On all supported Juniper Networks switches except the QFX10000 line of switches, configure the VTEP to retain the inner VLAN tag while de-encapsulating a packet.

```
[edit protocols l2-learning]
user@switch# set decapsulate-accept-inner-vlan
```

2. Configure the physical interface to support the simultaneous transmission of 802.1Q VLAN single-tagged and double-tagged packets on its logical interfaces and to accept packets carrying TPID 0x8100.

```
[edit interfaces]
user@switch# set xe-0/0/0 flexible-vlan-tagging
user@switch# set xe-0/0/0 encapsulation extended-vlan-bridge
```

3. Create logical interface 100, and associate it with S-VLAN 100. Also, specify that when logical interface 100 receives a packet without an outer S-VLAN tag, the interface pushes outer S-VLAN tag 100 on the outgoing packet.

```
[edit interfaces]
user@switch# set xe-0/0/0 unit 100 vlan-id 100
user@switch# set xe-0/0/0 unit 100 input-vlan-map pop
user@switch# set xe-0/0/0 unit 100 output-vlan-map push
```

NOTE: If you include the push configuration statement at the [edit interfaces unit output-vlan-map] hierarchy level, you must also include the pop configuration statement at the [edit interfaces unit input-vlan-map] hierarchy level to prevent an error when committing the configuration.

4. Create a VLAN named v100, and map it to logical interface 100 and VNI 1001. Also specify that the logical interface retains the inner VLAN tag while encapsulating a packet.

```
[edit vlans]
user@switch# set v100 interface xe-0/0/0.100
user@switch# set v100 vxlan vni 1001
user@switch# set v100 vxlan encapsulate-inner-vlan
```

NOTE: To support the tunneling of Q-in-Q packets, you must configure both ingress and egress VTEPs to retain the inner C-VLAN tag while encapsulating a packet.

Configuring Traffic Pattern 3: Retaining S-VLAN and C-VLAN Tags

IN THIS SECTION

- [Requirements | 825](#)
- [Introduction | 825](#)
- [Ingress and Egress VTEP Configuration for Traffic Pattern 3 | 825](#)

Requirements

Introduction

For this traffic pattern, the ingress and egress VTEPs in an EVPN-VXLAN overlay network must handle Q-in-Q data packets that are single- or double-tagged. For both single- and double-tagged packets, the ingress and egress VTEPs encapsulate and de-encapsulate the packets without making any changes to the tag(s).

NOTE: QFX Series and EX4600 switches support this traffic pattern on both aggregated Ethernet and non-aggregated Ethernet interfaces.

Ingress and Egress VTEP Configuration for Traffic Pattern 3

IN THIS SECTION

- [CLI Quick Configuration | 826](#)
- [Procedure | 826](#)

NOTE: This configuration focuses on traffic pattern 3 only. It does not provide the configuration for EVPN and all aspects of VXLAN. For a more comprehensive EVPN-VXLAN configuration for a centrally-routed bridging overlay, see ["Example: Configure an EVPN-VXLAN Centrally-Routed Bridging \(CRB\) Overlay" on page 530.](#)

CLI Quick Configuration

To quickly configure the ingress and egress VTEPs, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

NOTE: To configure a QFX10000 switch to retain the inner C-VLAN tag while it tunnels Q-in-Q packets, you must include only the encapsulate-inner-vlan configuration statement at the [edit vlans *vlan-name* vxlan] hierarchy level. You do not need to include the decapsulate-accept-inner-vlan configuration statement at the [edit protocols l2-learning] hierarchy level.

```
set protocols l2-learning decapsulate-accept-inner-vlan
set interfaces xe-0/0/15 flexible-vlan-tagging
set interfaces xe-0/0/15 encapsulation extended-vlan-bridge
set interfaces xe-0/0/15 unit 100 vlan-id 100
set interfaces xe-0/0/15 unit 201 vlan-id 201
set vlans vlan_100 interface xe-0/0/15.100
set vlans vlan_100 vxlan vni 1001
set vlans vlan_100 vxlan encapsulate-inner-vlan
set vlans vlan_201 interface xe-0/0/15.201
set vlans vlan_201 vxlan vni 1002
set vlans vlan_201 vxlan encapsulate-inner-vlan
```

Procedure

Step-by-Step Procedure

To configure the ingress and egress VTEP for traffic pattern 3:

1. On all supported Juniper Networks switches except the QFX10000 line of switches, configure the VTEPs to retain the inner VLAN tag while de-encapsulating a packet.

```
[edit protocols l2-learning]
user@switch# set decapsulate-accept-inner-vlan
```

NOTE: To support the tunneling of Q-in-Q packets, you must configure both ingress and egress VTEPs to retain the inner C-VLAN tag while de-encapsulating a packet.

2. Configure the physical interface to support the simultaneous transmission of 802.1Q VLAN single- and double-tagged packets on its logical interfaces and to accept packets carrying TPID 0x8100. Also, on the physical interface, create logical interfaces 100 and 201, and associate them with S-VLAN 100 and C-VLAN 201, respectively.

```
[edit interfaces]
user@switch# set xe-0/0/15 flexible-vlan-tagging
user@switch# set xe-0/0/15 encapsulation extended-vlan-bridge
user@switch# set xe-0/0/15 unit 100 vlan-id 100
user@switch# set xe-0/0/15 unit 201 vlan-id 201
```

3. Create a VLAN named `vlan_100`, and map it to logical interface 100 and VNI 1001. Also create a VLAN named `vlan_201`, and map it to logical interface 201 and VNI 1002. Also specify that the logical interfaces retain the inner VLAN tag while encapsulating a packet.

```
[edit vlans]
user@switch# set vlan_100 interface xe-0/0/15.100
user@switch# set vlan_100 vxlan vni 1001
user@switch# set vlan_100 vxlan encapsulate-inner-vlan
user@switch# set vlan_201 interface xe-0/0/15.201
user@switch# set vlan_201 vxlan vni 1002
user@switch# set vlan_201 vxlan encapsulate-inner-vlan
```

NOTE: To support the tunneling of Q-in-Q packets, you must configure both ingress and egress VTEPs to retain the inner C-VLAN tag while encapsulating a packet.

Configuring Traffic Pattern 4: Popping and Later Pushing an S-VLAN Tag

IN THIS SECTION

- Requirements | 828
- Introduction | 828
- Configuration for Ingress VTEP for Traffic Pattern 4 | 828
- Configuration for Egress VTEP for Traffic Pattern 4 | 830

Requirements

Introduction

For this traffic pattern, the ingress and egress VTEPs in an EVPN-VXLAN overlay network must handle double-tagged Q-in-Q traffic. The ingress VTEP retains the inner C-VLAN tag and removes, or pops, the outer S-VLAN tag. After the packets are tunneled over the Layer 3 network, the egress VTEP pushes the S-VLAN tag back on the packet.

NOTE: QFX Series and EX4600 switches support this traffic patterns on both aggregated Ethernet and non-aggregated Ethernet interfaces.

NOTE: This configuration focuses on traffic pattern 4 only. It does not provide the configuration for EVPN and all aspects of VXLAN. For a more comprehensive EVPN-VXLAN configuration for a centrally-routed bridging overlay, see ["Example: Configure an EVPN-VXLAN Centrally-Routed Bridging \(CRB\) Overlay" on page 530](#).

Configuration for Ingress VTEP for Traffic Pattern 4

IN THIS SECTION

- CLI Quick Configuration | 829
- Procedure | 829

CLI Quick Configuration

To quickly configure the ingress VTEP, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

NOTE: To configure a QFX10000 switch to retain the inner C-VLAN tag while it tunnels Q-in-Q packets, you must include only the encapsulate-inner-vlan configuration statement at the [edit vlans *vlan-name* vxlan] hierarchy level. You do not need to include the decapsulate-accept-inner-vlan configuration statement at the [edit protocols l2-learning] hierarchy level.

```
set protocols l2-learning decapsulate-accept-inner-vlan
set interfaces xe-0/0/0 flexible-vlan-tagging
set interfaces xe-0/0/0 encapsulation extended-vlan-bridge
set interfaces xe-0/0/0 unit 100 vlan-id 100
set interfaces xe-0/0/0 unit 100 input-vlan-map pop
set interfaces xe-0/0/0 unit 100 output-vlan-map push
set vlans vlan_1 interface xe-0/0/0.100
set vlans vlan_1 vxlan vni 1001
set vlans vlan_1 vxlan encapsulate-inner-vlan
```

Procedure

Step-by-Step Procedure

To configure the ingress VTEP for traffic pattern 4:

1. On all supported Juniper Networks switches except the QFX10000 line of switches, configure the VTEP to retain the inner C-VLAN tag while de-encapsulating a packet.

```
[edit protocols l2-learning]
user@switch# set decapsulate-accept-inner-vlan
```

NOTE: To support the VXLAN tunneling of Q-in-Q packets, you must configure both ingress and egress VTEP to retain the inner C-VLAN tag while de-encapsulating a packet.

2. Configure the physical interface to support the simultaneous transmission of 802.1Q VLAN single-tagged and double-tagged packets on its logical interfaces and to accept packets carrying TPID 0x8100.

```
[edit interfaces]
user@switch# set xe-0/0/0 flexible-vlan-tagging
user@switch# set xe-0/0/0 encapsulation extended-vlan-bridge
```

3. On physical interface xe-0/0/0, create logical interface 100, and associate it with S-VLAN 100. Also, assuming that the ingress VTEP receives a double-tagged packet as described in this traffic pattern, specify that the outer S-VLAN tag is popped on incoming packets. To accommodate a scenario in which the traffic flow is reversed, and the VTEP functions as an egress VTEP that receives a single-tagged packet from C-VLAN 200, you can optionally specify that an outer S-VLAN tag is added, or pushed, on outgoing packets.

```
[edit interfaces]
user@switch# set xe-0/0/0 unit 100 vlan-id 100
user@switch# set xe-0/0/0 unit 100 input-vlan-map pop
user@switch# set xe-0/0/0 unit 100 output-vlan-map push
```

4. Create a VLAN named vlan_1, and map it to logical interface 100 and VNI 1001. Also specify that the logical interface retains the inner VLAN tag while encapsulating a packet.

```
[edit vlans]
user@switch# set vlan_1 interface xe-0/0/0.100
user@switch# set vlan_1 vxlan vni 1001
user@switch# set vlan_1 vxlan encapsulate-inner-vlan
```

Configuration for Egress VTEP for Traffic Pattern 4

IN THIS SECTION

- [CLI Quick Configuration | 831](#)
- [Procedure | 831](#)

CLI Quick Configuration

To quickly configure the egress VTEP, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

NOTE: To configure a QFX10000 switch to retain the inner C-VLAN tag while it tunnels Q-in-Q packets, you must include only the encapsulate-inner-vlan configuration statement at the [edit vlans *vlan-name* vxlan] hierarchy level. You do not need to include the decapsulate-accept-inner-vlan configuration statement at the [edit protocols l2-learning] hierarchy level.

```
set protocols l2-learning decapsulate-accept-inner-vlan
set interfaces xe-0/0/5 flexible-vlan-tagging
set interfaces xe-0/0/5 encapsulation extended-vlan-bridge
set interfaces xe-0/0/5 unit 100 vlan-id 100
set interfaces xe-0/0/5 unit 100 input-vlan-map pop
set interfaces xe-0/0/5 unit 100 output-vlan-map push
set vlans vlan_1 interface xe-0/0/5.100
set vlans vlan_1 vxlan vni 1001
set vlans vlan_1 vxlan encapsulate-inner-vlan
```

Procedure

Step-by-Step Procedure

To configure the egress VTEP for traffic pattern 4:

1. On all supported Juniper Networks switches except the QFX10000 line of switches, configure the VTEP to retain the inner VLAN tag while de-encapsulating a packet.

```
[edit protocols l2-learning]
user@switch# set decapsulate-accept-inner-vlan
```

2. Configure the physical interface to support the simultaneous transmission of 802.1Q VLAN single-tagged and double-tagged packets on its logical interfaces and to accept packets carrying TPID 0x8100.

```
[edit interfaces]
user@switch# set xe-0/0/5 flexible-vlan-tagging
user@switch# set xe-0/0/5 encapsulation extended-vlan-bridge
```

3. Create logical interface 100, and associate it with S-VLAN 100. Also, specify that when logical interface 100 receives a packet without an outer S-VLAN tag, the interface pushes outer S-VLAN tag 100 on the outgoing packet.

```
[edit interfaces]
user@switch# set xe-0/0/5 unit 100 vlan-id-list 100
user@switch# set xe-0/0/5 unit 100 input-vlan-map pop
user@switch# set xe-0/0/5 unit 100 output-vlan-map push
```

NOTE: If you include the push configuration statement at the [edit interfaces unit output-vlan-map] hierarchy level, you must also include the pop configuration statement at the [edit interfaces unit input-vlan-map] hierarchy level to prevent an error when committing the configuration.

4. Create a VLAN named vlan_1, and map it to logical interface 100 and VNI 1001. Also specify that the logical interface retains the inner VLAN tag while encapsulating a packet.

```
[edit vlans]
user@switch# set vlan_1 interface xe-0/0/5.100
user@switch# set vlan_1 vxlan vni 1001
user@switch# set vlan_1 vxlan encapsulate-inner-vlan
```

NOTE: To support the tunneling of Q-in-Q packets, you must configure both ingress and egress VTEPs to retain the inner C-VLAN tag while encapsulating a packet.

Release History Table

Release	Description
18.3R1	Starting with Junos OS Release 18.3R1, the QFX10000 line of switches that function as Layer 2 VTEPs only can tunnel single- and double-tagged Q-in-Q packets in a centrally-routed bridging overlay.
18.2R1	Starting with Junos OS Release 18.2R1, QFX5110, QFX5200, and EX4600 switches that function as Layer 2 VTEPs can tunnel single- and double-tagged Q-in-Q packets in a centrally-routed bridging overlay.
17.2R1	Starting with Junos OS Release 17.2R1, QFX5100 switches that function as Layer 2 VXLAN tunnel endpoints (VTEPs) can tunnel single- and double-tagged Q-in-Q packets in an EVPN-VXLAN centrally-routed bridging overlay (EVPN-VXLAN network with a two-layer IP fabric).

Tunnel Traffic Inspection on SRX Series Devices

IN THIS CHAPTER

- [Tunnel Inspection for EVPN-VXLAN by SRX Series Devices | 834](#)

Tunnel Inspection for EVPN-VXLAN by SRX Series Devices

SUMMARY

Read this topic to understand how to setup your security device to perform tunnel inspection for EVPN-VXLAN to provide embedded security.

IN THIS SECTION

- [Overview | 834](#)
- [Example - Configure Security Policies for EVPN-VXLAN Tunnel Inspection | 837](#)
- [Configuration for Zone-Level Inspection, IDP, UTM and Advanced Anti-Malware for Tunnel Inspection | 850](#)
- [Complete Device Configurations | 860](#)

Overview

IN THIS SECTION

- [Benefits | 836](#)

(Ethernet VPN) EVPN-(Virtual Extensible LAN) VXLAN provides enterprises a common framework used to manage their campus and data center networks.

The rapidly increasing usage of mobile and IoT devices adds a large number of endpoints to a network. Modern enterprise networks must scale rapidly to provide immediate access to devices and to extend security and control for these endpoints.

To provide endpoint flexibility, EVPN-VXLAN decouples the underlay network (physical topology) from the overlay network (virtual topology). By using overlays, you gain the flexibility of providing Layer 2/ Layer 3 connectivity between endpoints across campus and data centers, while maintaining a consistent underlay architecture.

You can use SRX Series devices in your EVPN-VXLAN solution to connect end-points in your campus, data center, branch and public cloud environments while providing embedded security.

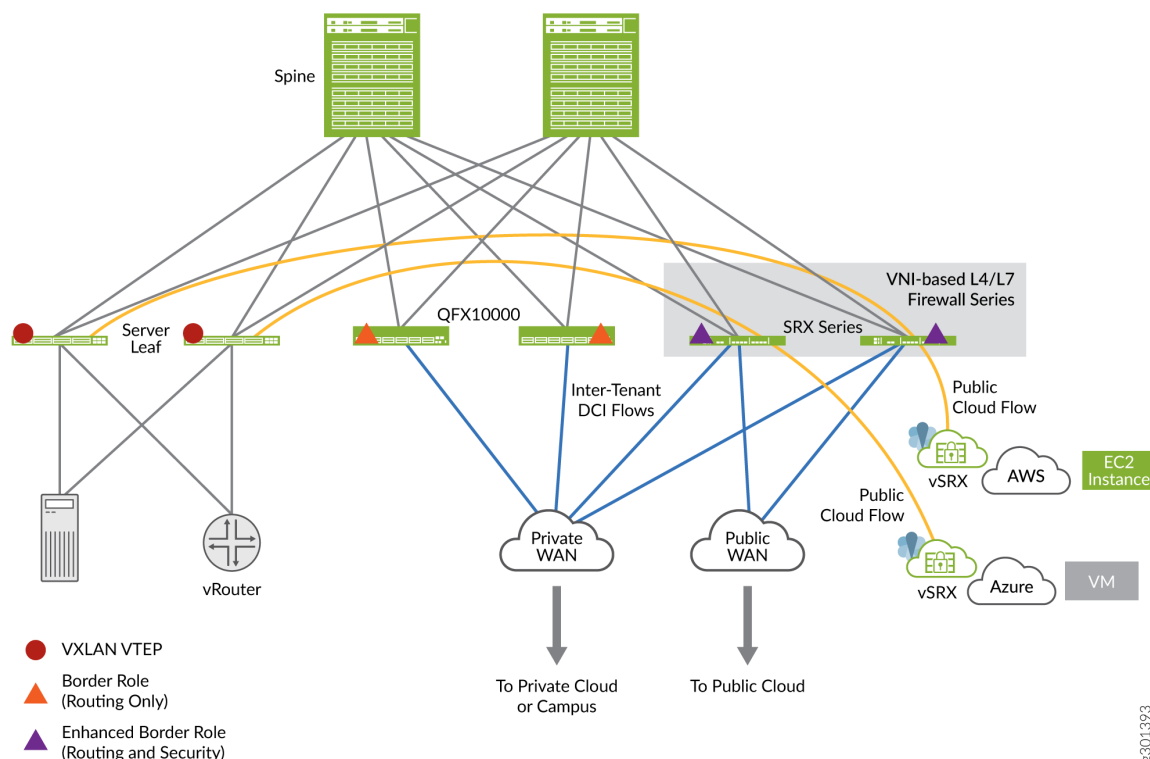
Starting in Junos OS Release 21.1R1, the SRX Series device can also apply following Layer 4/Layer 7 security services to the EVPN-VXLAN tunnel traffic:

- Application Identification
- IDP
- Juniper ATP (formerly known as SKY ATP)
- UTM

[Figure 86 on page 836](#) shows a typical deployment scenario of EVPN-VXLAN fabric based on Edge-routed bridging (ERB) with SRX Series devices functioning in an enhanced border leaf (EBL) role. EBL

enhances the traditional role of a border leaf with the ability to perform inspection of traffic in VXLAN tunnels.

Figure 86: EVPN-VXLAN Architecture with SRX Series Device



In the figure VXLAN traffic originating at the leaf 1 device traverses through the SRX Series devices that function as EBLs. In this use case, the SRX Series device is placed at the border, that is, at the entry and exit point of the campus or data center, to provide stateful inspection to the VXLAN encapsulated packets traversing through it.

In the architecture diagram, you can notice that an SRX Series device is placed between two VTEP devices (devices that perform VXLAN encapsulation and decapsulation for the network traffic). The SRX Series device performs stateful inspection when you enable the tunnel inspection feature with an appropriate security policy.

Benefits

Adding SRX Series device in EVPN VXLAN provides:

- Added security with the capabilities of an enterprise grade firewall in the EVPN-VXLAN overlay.
- Enhanced tunnel inspection for VXLAN encapsulated traffic with Layer 4/Layer 7 security services.

Example - Configure Security Policies for EVPN-VXLAN Tunnel Inspection

IN THIS SECTION

- [Requirements | 837](#)
- [Before you Begin | 837](#)
- [Overview | 838](#)
- [Configuration | 839](#)
- [CLI Quick Configuration | 840](#)
- [Results | 844](#)
- [Verification | 846](#)

Use this example to configure the security policies that enable inspection of EVPN EVPN-VXLAN tunnel traffic on your SRX Series devices.

Requirements

This example uses the following hardware and software components:

- An SRX Series device or vSRX
- Junos OS Release 20.4R1

This example assumes that you already have an EVPN-VXLAN based network and want to enable tunnel inspection on SRX Series device.

Before you Begin

- Ensure you have a valid application identification feature license on your SRX Series device and installed application signature pack on the device.
- Make sure you understand how EVPN and VXLAN works. See [EVPN-VXLAN Campus Architectures](#) to detail understanding EVPN-VXLAN
- This example assumes that you already have an EVPN-VXLAN based network fabric and want to enable tunnel inspection on the SRX Series device. You can see the sample configuration of leaf and spine devices used in this example at "[Complete Device Configurations](#)" on page 860.

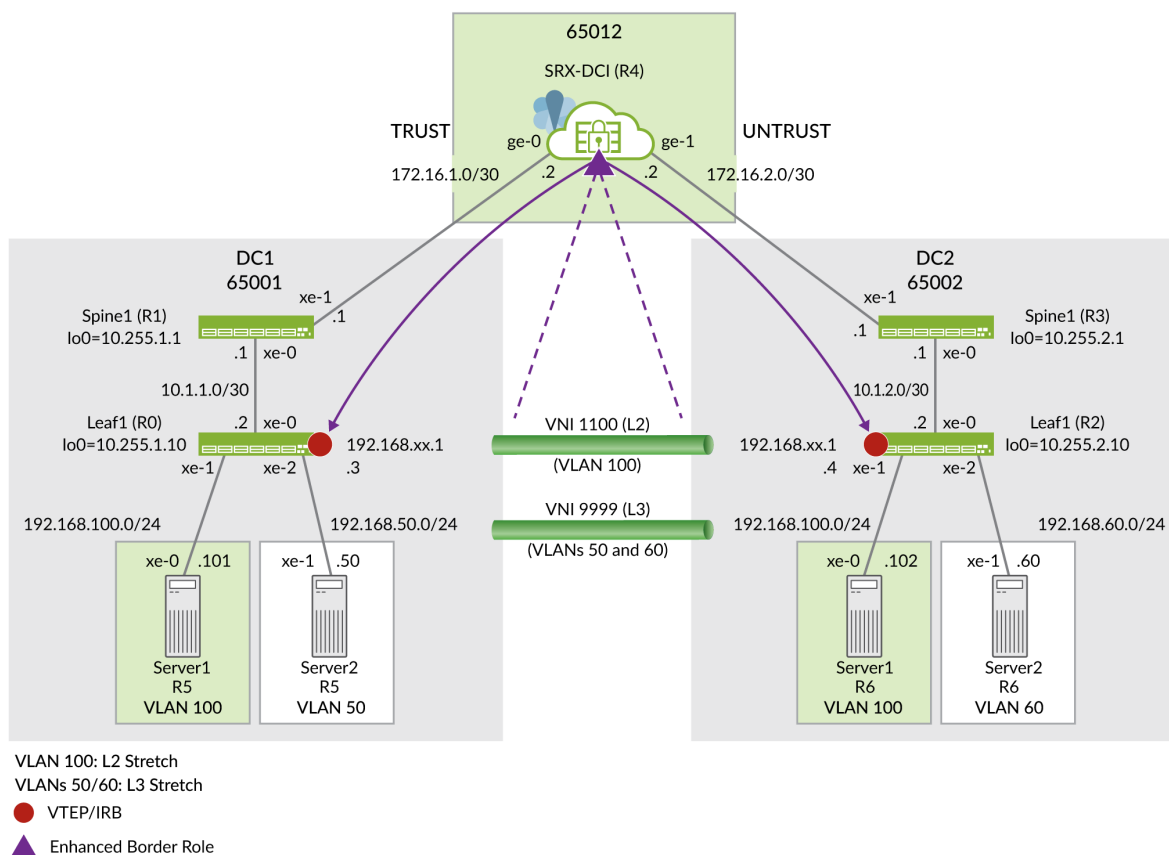
In this example, we are focusing on configuring the SRX Series device which is a part of a working EVPN-VXLAN network that consist of two DC locations each with an IP fabric. The SRX Series device is

placed in a Data Center Interconnect (DCI) role between the two DCs. In this configuration, the SRX Series device performs stateful inspection of VXLAN encapsulated traffic flowing between the DCs when you enable tunnel inspection.

We are using the topology shown in [Figure 87 on page 838](#) in this example.

Overview

Figure 87: Topology for VXLAN Tunnel Inspection



As given in the topology, the SRX Series device is inspecting transit VLAN encapsulated traffic from the VXLAN tunnel endpoint (VTEP) on the leaves in both the DC-1 and DC-2 data centers. Any Juniper Networks device, both physical and virtual, that functions as a Layer 2 or Layer 3 VXLAN gateway can act as VTEP device to perform encapsulation and de-encapsulation.

Upon receipt of a Layer 2 or Layer 3 data packet from server 1, The leaf 1 VTEP adds the appropriate VXLAN header and then encapsulates the packet with a IPv4 outer header to facilitate tunneling the packet through the IPv4 underlay network. The remote VTEP at leaf 2 then de-encapsulates the traffic and forwards the original packet towards the destination host. With the Junos software release 20.4

SRX Series devices are able to perform tunnel inspection for VXLAN encapsulated overlay traffic passing through it.

In this example, you'll create a security policy to enable inspection for traffic that is encapsulated in a VXLAN tunnel . We're using the parameters described [Table 39 on page 839](#) in this example.

Table 39: Configuration Parameters

Parameter	Description	Parameter Name
Security policy	Policy to create a flow session triggered by VXLAN overlay traffic. This policy references the outer IP source and destination address. That is, the IP addresses of the source and destination VTEPs. In this example this is the loopback address of the leaves.	P1
Policy set	Policy for the inspection of inner traffic. This policy operates on the contents of matching VXLAN tunnel traffic.	PSET-1
Tunnel inspection profile	Specifies parameters for security inspection on VXLAN tunnels.	TP-1
Name of a VXLAN network identifier (VNI) list or range	Used to uniquely identify a list or range of VXLAN tunnel IDs.	VLAN-100
VXLAN tunnel identifier name.	Used to symbolically name a VXLAN tunnel in a tunnel inspection profile.	VNI-1100

When you configure tunnel inspection security policies on the SRX Series device, it decapsulates the packet to access the inner header when a packet matches a security policy. Next, it applies the tunnel inspection profile to determine if the inner traffic is permitted. The security device uses inner packet content and the applied tunnel inspection profile parameters to do a policy lookup and to then perform stateful inspection for the inner session.

Configuration

In this example, you'll configure the following functionality on the SRX Series device:

1. Define a trust and untrust zone to permit all host traffic. This supports the BGP session to the spine devices and allow SSH etc from either zone (DC).
2. Inspect traffic flowing from DC1 to DC2 in VNI 1100 (Layer 2 stretched for VLAN 100) for all hosts in the 192.168.100.0/24 subnet. Your policy should permit pings but deny all other traffic.
3. Allow all return traffic from DC2 to DC1 with no tunnel inspection.
4. Allow all other underlay and overlay traffic without VXLAN tunnel inspection from DC1 to DC2.

Use the following steps to enable tunnel inspection on your security device in a VXLAN-EVPN environment:

NOTE: Complete functional configurations for all devices used in this example are provided "[Complete Device Configurations](#)" on [page 860](#) to assist the reader in testing this example. This example focuses on the configuration steps needed to enable and validate the VXLAN tunnel inspection feature. The SRX device is presumed to be configured with interface addressing, BGP peering, and policy to support its DCI role.

CLI Quick Configuration

To quickly configure this example on your SRX series device, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

Configuration on SRX Series Device

```
set system host-name r4-dci-ebr
set security address-book global address vtep-untrust 10.255.2.0/24
set security address-book global address vtep-trust 10.255.1.0/24
set security address-book global address vlan100 192.168.100.0/24
set security policies from-zone trust to-zone untrust policy P1 match source-address vtep-trust
security policies from-zone trust to-zone untrust policy P1 match destination-address vtep-
untrust
set security policies from-zone trust to-zone untrust policy P1 match application junos-vxlan
set security policies from-zone trust to-zone untrust policy P1 then permit tunnel-inspection
TP-1
set security policies from-zone untrust to-zone trust policy accept-all-dc2 match source-address
any
set security policies from-zone untrust to-zone trust policy accept-all-dc2 match destination-
address any
set security policies from-zone untrust to-zone trust policy accept-all-dc2 match application any
```

```

set security policies from-zone untrust to-zone trust policy accept-all-dc2 then permit
set security policies policy-set PSET-1 policy PSET-1-P1 match source-address vlan100
set security policies policy-set PSET-1 policy PSET-1-P1 match destination-address vlan100
set security policies policy-set PSET-1 policy PSET-1-P1 match application junos-icmp-all
set security policies policy-set PSET-1 policy PSET-1-P1 then permit
set security zones security-zone trust host-inbound-traffic system-services all
set security zones security-zone trust host-inbound-traffic protocols all
set security zones security-zone trust interfaces ge-0/0/0.0
set security zones security-zone untrust host-inbound-traffic system-services all
set security zones security-zone untrust host-inbound-traffic protocols all
set security zones security-zone untrust interfaces ge-0/0/1.0
set security tunnel-inspection inspection-profile TP-1 vxlan VNI-1100 policy-set PSET-1
set security tunnel-inspection inspection-profile TP-1 vxlan VNI-1100 vni VLAN-100
set security tunnel-inspection vni VLAN-100 vni-id 1100
set interfaces ge-0/0/0 description "Link to DC1 Spine 1"
set interfaces ge-0/0/0 mtu 9000
set interfaces ge-0/0/0 unit 0 family inet address 172.16.1.2/30
set interfaces ge-0/0/1 description "Link to DC2 Spine 1"
set interfaces ge-0/0/1 mtu 9000
set interfaces ge-0/0/1 unit 0 family inet address 172.16.2.2/30

```

Step-by-Step Procedure

1. Configure security zones, interfaces, and address-books.

```

[edit]
user@@r4-dci-ebr# set security zones security-zone trust
user@@r4-dci-ebr# set security zones security-zone untrust
user@@r4-dci-ebr# set interfaces ge-0/0/0 description "Link to DC1 Spine 1"
user@@r4-dci-ebr# set interfaces ge-0/0/0 mtu 9000
user@@r4-dci-ebr# set interfaces ge-0/0/0 unit 0 family inet address 172.16.1.2/30
user@@r4-dci-ebr# set interfaces ge-0/0/1 description "Link to DC2 Spine 1"
user@@r4-dci-ebr# set interfaces ge-0/0/1 mtu 9000
user@@r4-dci-ebr# set interfaces ge-0/0/1 unit 0 family inet address 172.16.2.2/30
user@@r4-dci-ebr# set security zones security-zone trust host-inbound-traffic system-services
all
user@@r4-dci-ebr# set security zones security-zone trust host-inbound-traffic protocols all
user@@r4-dci-ebr# set security zones security-zone trust interfaces ge-0/0/0.0
user@@r4-dci-ebr# set security zones security-zone untrust host-inbound-traffic system-
services all
user@@r4-dci-ebr# set security zones security-zone untrust host-inbound-traffic protocols all

```

```

user@r4-dci-ebr# set security zones security-zone untrust interfaces ge-0/0/1.0
user@r4-dci-ebr# set security address-book global address vtep-untrust 10.255.2.0/24
user@r4-dci-ebr# set security address-book global address vtep-trust 10.255.1.0/24
user@r4-dci-ebr# set security address-book global address vlan100 192.168.100.0/24

```

Note that /24 prefix lengths are used to specify the outer (VTEP) and inner (server) addresses. While you could use /32 host routes for this simple example, using a /24 will match traffic from other leaves (VTEPs) or hosts in the 192.168.100.0/24 subnet.

2. Define the tunnel-inspection profile. You can specify a range or a list of VNIs that should be inspected.

```

[edit]
user@r4-dci-ebr# set security tunnel-inspection vni VLAN-100 vni-id 1100
user@r4-dci-ebr# set security tunnel-inspection inspection-profile TP-1 vxlan VNI-1100 vni
VLAN-100
user@r4-dci-ebr# set security tunnel-inspection inspection-profile TP-1 vxlan VNI-1100
policy-set PSET-1

```

In this example only one VNI is needed so the `vni-id` keyword is used instead of the `vni-range` option.

The tunnel inspection profile links to both the VNI list/range as well as to the related policy that should be applied to VXLAN tunnel with matching VNIs.

3. Create a security policy to match on the outer session.

```

[edit]
user@r4-dci-ebr# set security policies from-zone trust to-zone untrust policy P1 match
source-address vtep-trust
user@r4-dci-ebr# set security policies from-zone trust to-zone untrust policy P1 match
destination-address vtep-untrust
user@r4-dci-ebr# set security policies from-zone trust to-zone untrust policy P1 match
application junos-vxlan
user@r4-dci-ebr# set security policies from-zone trust to-zone untrust policy P1 then permit
tunnel-inspection TP-1

```

This policy refers to the global address book entries you defined earlier to match source and destination VTEP addresses. These addresses are used in the underlay to support VXLAN tunnels in the overlay. Matching traffic is directed to the TP-1 tunnel inspection profile you defined in the previous step. In this example the goal is to inspect VXLAN tunnels that originate in DC1 and terminate in DC2. As a result a second policy to match on return traffic (with DC2 Leaf 1 the source VTEP) is not needed.

4. Create the policy-set for the inner session.

```
[edit]
user@@r4-dci-ebr# set security policies policy-set PSET-1 policy PSET-1-P1 match source-
address vlan100
user@@r4-dci-ebr# set security policies policy-set PSET-1 policy PSET-1-P1 match destination-
address vlan100
user@@r4-dci-ebr# set security policies policy-set PSET-1 policy PSET-1-P1 match application
junos-icmp-all
user@@r4-dci-ebr# set security policies policy-set PSET-1 policy PSET-1-P1 then permit
```

This policy performs security inspection against the payload of matching VXLAN traffic. In this example this is the traffic sent from Server 1 on VLAN 100 in DC1 to Server 1 in DC2. By specifying the `junos-icmp-all` match condition you ensure that both ping request and replies can pass from server 1 in DC1 to server 1 in DC2. If you specify `junos-icmp-ping` only pings that originate from DC1 will be permitted.

Recall that in this example only ping is permitted to help facilitate testing of the resulting functionality. You can match on `application any` to permit all traffic, or alter the match criteria to suit your specific security needs.

5. Define the policies needed to accept all other traffic between the data centers without any tunnel inspection.

```
[edit]
user@@r4-dci-ebr# set security policies from-zone trust to-zone untrust policy accept-rest
match source-address any
user@@r4-dci-ebr# set security policies from-zone trust to-zone untrust policy accept-rest
match destination-address any
user@@r4-dci-ebr# set security policies from-zone trust to-zone untrust policy accept-rest
match application any
user@@r4-dci-ebr# set security policies from-zone trust to-zone untrust policy accept-rest
then permit
user@@r4-dci-ebr# set security policies from-zone untrust to-zone trust policy accept-all-dc2
match source-address any
user@@r4-dci-ebr# set security policies from-zone untrust to-zone trust policy accept-all-dc2
match destination-address any
user@@r4-dci-ebr# set security policies from-zone untrust to-zone trust policy accept-all-dc2
match application any
user@@r4-dci-ebr# set security policies from-zone untrust to-zone trust policy accept-all-dc2
then permit
```

Results

From configuration mode, confirm your configuration by entering the `show security` command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

[edit]

user@host# **show security**

```
address-book {
  global {
    address vtep-untrust 10.255.2.0/24;
    address vtep-trust 10.255.1.0/24;
    address vlan100 192.168.100.0/24;
  }
}
policies {
  from-zone trust to-zone untrust {
    policy P1 {
      match {
        source-address vtep-trust;
        destination-address vtep-untrust;
        application junos-vxlan;
      }
      then {
        permit {
          tunnel-inspection {
            TP-1;
          }
        }
      }
    }
  }
  policy accept-rest {
    match {
      source-address any;
      destination-address any;
      application any;
    }
    then {
      permit;
    }
  }
}
```

```

}
from-zone untrust to-zone trust {
    policy accept-all-dc2 {
        match {
            source-address any;
            destination-address any;
            application any;
        }
        then {
            permit;
        }
    }
}
policy-set PSET-1 {
    policy PSET-1-P1 {
        match {
            source-address vlan100;
            destination-address vlan100;
            application junos-icmp-all;
        }
        then {
            permit;
        }
    }
}
zones {
    security-zone trust {
        host-inbound-traffic {
            system-services {
                all;
            }
            protocols {
                all;
            }
        }
        interfaces {
            ge-0/0/0.0;
        }
    }
    security-zone untrust {
        host-inbound-traffic {
            system-services {

```



```

        all;
    }
    protocols {
        all;
    }
}
interfaces {
    ge-0/0/1.0;
}
}
}
tunnel-inspection {
    inspection-profile TP-1 {
        vxlan VNI-1100 {
            policy-set PSET-1;
            vni VLAN-100;
        }
    }
    vni VLAN-100 {
        vni-id 1100;
    }
}
}

```

If you are done configuring the feature on your device, enter `commit` from configuration mode.

Verification

At this time you should generate ping traffic between server 1 in DC1 to server 1 in DC2. The pings should succeed. Allow this test traffic to run in the background while you complete the verification tasks.

```

r5-dc1_server1> ping 192.168.100.102
PING 192.168.100.102 (192.168.100.102): 56 data bytes
64 bytes from 192.168.100.102: icmp_seq=0 ttl=64 time=565.451 ms
64 bytes from 192.168.100.102: icmp_seq=1 ttl=64 time=541.035 ms
64 bytes from 192.168.100.102: icmp_seq=2 ttl=64 time=651.420 ms
64 bytes from 192.168.100.102: icmp_seq=3 ttl=64 time=303.533 ms
. . .

```

Verify Inner Policy Details

Purpose

Verify the details of the policy applied for the inner session.

Action

From operational mode, enter the `show security policies policy-set PSET-1` command.

```
From zone: PSET-1, To zone: PSET-1
  Policy: PSET-1-P1, State: enabled, Index: 7, Scope Policy: 0, Sequence number: 1, Log Profile
  ID: 0
    From zones: any
    To zones: any
    Source vrf group: any
    Destination vrf group: any
    Source addresses: vlan100
    Destination addresses: vlan100
    Applications: junos-icmp-all
    Source identity feeds: any
    Destination identity feeds: any
    Action: permit
```

Check Tunnel Inspection Traffic

Purpose

Display the tunnel inspection traffic details.

Action

From operational mode, enter the `show security flow tunnel-inspection statistics` command.

```
Flow Tunnel-inspection statistics:
Tunnel-inspection type VXLAN:
  overlay session active:          4
  overlay session create:         289
  overlay session close:          285
  underlay session active:         3
  underlay session create:        31
```

```

underlay session close:      28
input packets:              607
input bytes:                171835
output packets:             418
output bytes:               75627
bypass packets:             0
bypass bytes:               0

```

Check Tunnel Inspection Profile and VNI

Purpose

Display the tunnel inspection profile and VNI details.

Action

From operational mode, enter the `show security tunnel-inspection profiles` command.

```

Logical system: root-logical-system
Profile count: 1
Profile: TP-1
  Type: VXLAN
  Vxlan count: 1
  Vxlan name: VXT-1
  VNI count: 1
    VNI:VNI-1
  Policy set: PSET-1
  Inspection level: 1

```

From operational mode, enter the `show security tunnel-inspection vnis` command.

```

Logical system: root-logical-system
VNI count: 2
VNI name: VLAN-100
  VNI id count: 1
  [1100 - 1100]
VNI name: VNI-1
  VNI id count: 1
  [1100 - 1100]

```

Check Security Flows

Purpose

Display VXLAN security flow information on the SRX to confirm that VXLAN tunnel inspection is working.

Action

From operational mode, enter the `show security flow session vxlan-vni 1100` command.

```
Session ID: 3811, Policy name: PSET-1-P1/7, State: Stand-alone, Timeout: 2, Valid
  In: 192.168.100.101/47883 --> 192.168.100.102/82;icmp, Conn Tag: 0xfcd, If: ge-0/0/0.0, Pkts:
1, Bytes: 84,
  Type: VXLAN, VNI: 1100, Tunnel Session ID: 2193
  Out: 192.168.100.102/82 --> 192.168.100.101/47883;icmp, Conn Tag: 0xfcd, If: ge-0/0/1.0, Pkts:
0, Bytes: 0,
  Type: VXLAN, VNI: 0, Tunnel Session ID: 0

Session ID: 3812, Policy name: PSET-1-P1/7, State: Stand-alone, Timeout: 2, Valid
  In: 192.168.100.101/47883 --> 192.168.100.102/83;icmp, Conn Tag: 0xfcd, If: ge-0/0/0.0, Pkts:
1, Bytes: 84,
  Type: VXLAN, VNI: 1100, Tunnel Session ID: 2193
  Out: 192.168.100.102/83 --> 192.168.100.101/47883;icmp, Conn Tag: 0xfcd, If: ge-0/0/1.0, Pkts:
0, Bytes: 0,
  Type: VXLAN, VNI: 0, Tunnel Session ID: 0

. . .
```

Confirm That SSH is Blocked

Purpose

Try to establish an SSH session between server 1 in DC1 and server 2 in DC2. Based on the policy that allows only ping traffic this session should be blocked at the SRX.

Action

From operational mode, enter the `show security flow session vxlan-vni 1100` command.

```
r5-dc1_server1> ssh 192.168.100.102
ssh: connect to host 192.168.100.102 port 22: Operation timed out
r5_dc1_server1>
```

Configuration for Zone-Level Inspection, IDP, UTM and Advanced Anti-Malware for Tunnel Inspection

IN THIS SECTION

- [CLI Quick Configuration | 851](#)
- [Create Zone-Level Inspection for Tunnel Inspection | 852](#)
- [Create IDP, UTM and Advanced Anti-Malware for Tunnel Inspection | 852](#)
- [Results | 855](#)

Use this step if you want to configure zone-level inspection, and apply layer 7 services such as IDP, Juniper ATP, UTM, and advanced anti-malware to the tunnel traffic. This feature is supported from Junos OS Release 21.1R1 onwards.

This example uses the following hardware and software components:

- An SRX Series device or vSRX
- Junos OS Release 21.1R1

We are using the same configuration of address-books, security zones, interfaces, tunnel inspection profile, and security policy for the outer session as created in ["Configuration" on page 839](#)

This step assumes that you have Enrolled your SRX Series device with Juniper ATP. For details on how to enroll your SRX Series device, see [Enrolling an SRX Series Device With Juniper Advanced Threat Prevention Cloud](#).

In this configuration, you'll create a policy set for the inner session and apply IDP, UTM, advanced antimalware to the tunnel traffic.

CLI Quick Configuration

To quickly configure this example on your SRX series device , copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

Configuration on SRX Series Device

```

set system host-name r4-dci-ebc
set security address-book global address vtep-untrust 10.255.2.0/24
set security address-book global address vtep-trust 10.255.1.0/24
set security address-book global address vlan100 192.168.100.0/24
set security policies from-zone trust to-zone untrust policy P1 match source-address vtep-trust
security policies from-zone trust to-zone untrust policy P1 match destination-address vtep-
untrust
set security policies from-zone trust to-zone untrust policy P1 match application junos-vxlan
set security policies from-zone trust to-zone untrust policy P1 then permit tunnel-inspection
TP-1
set security policies from-zone untrust to-zone trust policy accept-all-dc2 match source-address
any
set security policies from-zone untrust to-zone trust policy accept-all-dc2 match destination-
address any
set security policies from-zone untrust to-zone trust policy accept-all-dc2 match application any
set security policies from-zone untrust to-zone trust policy accept-all-dc2 then permit
set security policies policy-set PSET-1 policy PSET-1-P1 match source-address vlan100
set security policies policy-set PSET-1 policy PSET-1-P1 match destination-address vlan100
set security policies policy-set PSET-1 policy PSET-1-P1 match application junos-icmp-all
set security policies policy-set PSET-1 policy PSET-1-P1 match dynamic-application any
set security policies policy-set PSET-1 policy PSET-1-P1 match url-category any
set security policies policy-set PSET-1 policy PSET-1-P1 match from-zone trust
set security policies policy-set PSET-1 policy PSET-1-P1 match to-zone untrust
set security policies policy-set PSET-1 policy PSET-1-P1 then permit
set security policies policy-set PSET-1 policy PSET-1-P1 then permit
set security zones security-zone trust host-inbound-traffic system-services all
set security zones security-zone trust host-inbound-traffic protocols all
set security zones security-zone trust interfaces ge-0/0/0.0
set security zones security-zone untrust host-inbound-traffic system-services all
set security zones security-zone untrust host-inbound-traffic protocols all
set security zones security-zone untrust interfaces ge-0/0/1.0
set security tunnel-inspection inspection-profile TP-1 vxlan VNI-1100 policy-set PSET-1
set security tunnel-inspection inspection-profile TP-1 vxlan VNI-1100 vni VLAN-100
set security tunnel-inspection vni VLAN-100 vni-id 1100
set interfaces ge-0/0/0 description "Link to DC1 Spine 1"

```

```

set interfaces ge-0/0/0 mtu 9000
set interfaces ge-0/0/0 unit 0 family inet address 172.16.1.2/30
set interfaces ge-0/0/1 description "Link to DC2 Spine 1"
set interfaces ge-0/0/1 mtu 9000
set interfaces ge-0/0/1 unit 0 family inet address 172.16.2.2/30

```

Create Zone-Level Inspection for Tunnel Inspection

You can add zone-level policy control for EVPN-VXLAN tunnel inspection for the inner traffic. This policy performs security inspection against the payload of matching VXLAN traffic. In the following step, you'll specify from-zone and to-zone for the traffic.

- [edit]

```

user@r4-dci-ebr# set security policies policy-set PSET-1 policy PSET-1-P1 match source-address vlan100
user@r4-dci-ebr# set security policies policy-set PSET-1 policy PSET-1-P1 match destination-address vlan100
user@r4-dci-ebr# set security policies policy-set PSET-1 policy PSET-1-P1 match application any
user@r4-dci-ebr# set security policies policy-set PSET-1 policy PSET-1-P1 match dynamic-application any
user@r4-dci-ebr# set security policies policy-set PSET-1 policy PSET-1-P1 match url-category any
user@r4-dci-ebr# set security policies policy-set PSET-1 policy PSET-1-P1 match from-zone trust
user@r4-dci-ebr# set security policies policy-set PSET-1 policy PSET-1-P1 match to-zone untrust
user@r4-dci-ebr# set security policies policy-set PSET-1 policy PSET-1-P1 then permit

```

Create IDP, UTM and Advanced Anti-Malware for Tunnel Inspection

You can add security services such as IDP, advanced anti-malware, UTM, SSL proxy for the EVPN-VXLAN tunnel inspection for the inner traffic. This policy performs security inspection against the payload of matching VXLAN traffic.

In the following step, you'll enable service such as IDP, UTM, SSL proxy, security-intelligence, advanced anti-malware services by specifying them in a security policy permit action, when the traffic matches the policy rule.

- [edit]

```

user@r4-dci-ebr# set security policies policy-set PSET-1 policy PSET-1-P1 match source-

```

```

address vlan100
user@r4-dci-ebr# set security policies policy-set PSET-1 policy PSET-1-P1 match destination-
address vlan100
user@r4-dci-ebr# set security policies policy-set PSET-1 policy PSET-1-P1 match application
any
user@r4-dci-ebr# set security policies policy-set PSET-1 policy PSET-1-P1 then permit
application-services ssl-proxy profile-name ssl-inspect-profile-1
user@r4-dci-ebr# set security policies policy-set PSET-1 policy PSET-1-P1 then permit
application-services security-intelligence-policy secintel1
user@r4-dci-ebr# set security policies policy-set PSET-1 policy PSET-1-P1 then permit
application-services advanced-anti-malware-policy P3
user@r4-dci-ebr# set security policies policy-set PSET-1 policy PSET-1-P1 then permit
application-services idp-policy idp123
user@r4-dci-ebr# set security policies policy-set PSET-1 policy PSET-1-P1 then permit
application-services utm-policy P1

```

The following steps show configuration snippets of UTM, IDP, and advanced anti-malware policies at glance.

- Configure advanced anti-malware policy.

```

[edit]
user@r4-dci-ebr# set services advanced-anti-malware policy P3 http inspection-profile
scripts
user@r4-dci-ebr# set services advanced-anti-malware policy P3 http action block
user@r4-dci-ebr# set services advanced-anti-malware policy P3 http notification log
user@r4-dci-ebr# set services advanced-anti-malware policy P3 http client-notify message
"AAMW Blocked!"
user@r4-dci-ebr# set services advanced-anti-malware policy P3 verdict-threshold recommended
user@r4-dci-ebr# set services advanced-anti-malware policy P3 fallback-options action permit
user@r4-dci-ebr# set services advanced-anti-malware policy P3 fallback-options notification
log

```

- Configure security intelligence profile.

```

[edit]
user@r4-dci-ebr# set services security-intelligence url https://
cloudfeeds.argonqa.junipersecurity.net/api/manifest.xml
user@r4-dci-ebr# set services security-intelligence authentication tls-profile aamw-ssl
user@r4-dci-ebr# set services security-intelligence profile cc_profile category CC
user@r4-dci-ebr# set services security-intelligence profile cc_profile rule cc_rule match

```



```

threat-level 1
user@r4-dci-ebr# set services security-intelligence profile cc_profile rule cc_rule match
threat-level 2
user@r4-dci-ebr# set services security-intelligence profile cc_profile rule cc_rule match
threat-level 4
user@r4-dci-ebr# set services security-intelligence profile cc_profile rule cc_rule match
threat-level 5
user@r4-dci-ebr# set services security-intelligence profile cc_profile rule cc_rule match
threat-level 6
user@r4-dci-ebr# set services security-intelligence profile cc_profile rule cc_rule match
threat-level 7
user@r4-dci-ebr# set services security-intelligence profile cc_profile rule cc_rule match
threat-level 8
user@r4-dci-ebr# set services security-intelligence profile cc_profile rule cc_rule match
threat-level 9
user@r4-dci-ebr# set services security-intelligence profile cc_profile rule cc_rule match
threat-level 10
user@r4-dci-ebr# set services security-intelligence profile cc_profile rule cc_rule then
action block close
user@r4-dci-ebr# set services security-intelligence profile cc_profile rule cc_rule then log
user@r4-dci-ebr# set services security-intelligence profile ih_profile category Infected-Hosts
user@r4-dci-ebr# set services security-intelligence profile ih_profile rule ih_rule match
threat-level 7
user@r4-dci-ebr# set services security-intelligence profile ih_profile rule ih_rule match
threat-level 8
user@r4-dci-ebr# set services security-intelligence profile ih_profile rule ih_rule match
threat-level 9
user@r4-dci-ebr# set services security-intelligence profile ih_profile rule ih_rule match
threat-level 10
user@r4-dci-ebr# set services security-intelligence profile ih_profile rule ih_rule then
action block close http message "Blocked!"
user@r4-dci-ebr# set services security-intelligence profile ih_profile rule ih_rule then log
user@r4-dci-ebr# set services security-intelligence policy secintel1 CC cc_profile
user@r4-dci-ebr# set services security-intelligence policy secintel1 Infected-Hosts ih_profile

```

- Configure IDP policy.

```

[edit]
user@r4-dci-ebr# set security idp idp-policy idp123 rulebase-ips rule rule1 match application
junos-icmp-all

```

```
user@r4-dci-ebr# set security idp idp-policy idp123 rulebase-ips rule rule1 then action no-action
```

- Configure UTM policy.

```
[edit]
user@r4-dci-ebr# set security utm default-configuration anti-virus type sophos-engine
user@r4-dci-ebr## set security utm utm-policy P1 anti-virus http-profile junos-sophos-av-defaults
```

- Configure SSL profiles.

```
[edit]
user@r4-dci-ebr# set services ssl initiation profile aamw-ssl
user@r4-dci-ebr# set services ssl proxy profile ssl-inspect-profile-1 root-ca VJSA
```

Results

From configuration mode, confirm your configuration by entering the `show security` command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

[edit]

```
user@host# show security
```

```
address-book {
  global {
    address vtep-untrust 10.255.2.0/24;
    address vtep-trust 10.255.1.0/24;
    address vlan100 192.168.100.0/24;
  }
}
policies {
  from-zone trust to-zone untrust {
    policy P1 {
      match {
        source-address vtep-trust;
        destination-address vtep-untrust;
        application junos-vxlan;
      }
    }
  }
}
```

```

    }
    then {
        permit {
            tunnel-inspection {
                TP-1;
            }
        }
    }
}
policy accept-rest {
    match {
        source-address any;
        destination-address any;
        application any;
    }
    then {
        permit;
    }
}
}
from-zone untrust to-zone trust {
    policy accept-all-dc2 {
        match {
            source-address any;
            destination-address any;
            application any;
        }
        then {
            permit;
        }
    }
}
policy-set PSET-1 {
    policy PSET-1-P1 {
        match {
            source-address vlan100;
            destination-address vlan100;
            application junos-icmp-all;
            dynamic-application any;
            url-category any;
            from-zone trust;
            to-zone untrust;
        }
    }
}

```

```

        then {
            permit {
                application-services {
                    idp-policy idp123;
                    ssl-proxy {
                        profile-name ssl-inspect-profile-1;
                    }
                    utm-policy P1;
                    security-intelligence-policy secintel1;
                    advanced-anti-malware-policy P3;
                }
            }
        }
    }
}
zones {
    security-zone trust {
        host-inbound-traffic {
            system-services {
                all;
            }
            protocols {
                all;
            }
        }
        interfaces {
            ge-0/0/0.0;
        }
    }
    security-zone untrust {
        host-inbound-traffic {
            system-services {
                all;
            }
            protocols {
                all;
            }
        }
        interfaces {
            ge-0/0/1.0;
        }
    }
}

```

```

    }
}
tunnel-inspection {
    inspection-profile TP-1 {
        vxlan VNI-1100 {
            policy-set PSET-1;
            vni VLAN-100;
        }
    }
    vni VLAN-100 {
        vni-id 1100;
    }
}
}

```

[edit]

user@host# **show services**

```

application-identification;
ssl {
    initiation {
        profile aamw-ssl;
    }
    proxy {
        profile ssl-inspect-profile-1 {
            root-ca VJSA;
        }
    }
}
advanced-anti-malware {
    policy P3 {
        http {
            inspection-profile scripts;
            action block;
            client-notify {
                message "AAMW Blocked!";
            }
            notification {
                log;
            }
        }
        verdict-threshold recommended;
    }
}

```

```

        fallback-options {
            action permit;
            notification {
                log;
            }
        }
    }
}

security-intelligence {
    url https://cloudfeeds.argonqa.junipersecurity.net/api/manifest.xml;
    authentication {
        tls-profile aamw-ssl;
    }
    profile cc_profile {
        category CC;
        rule cc_rule {
            match {
                threat-level [ 1 2 4 5 6 7 8 9 10 ];
            }
            then {
                action {
                    block {
                        close;
                    }
                }
                log;
            }
        }
    }
}

profile ih_profile {
    category Infected-Hosts;
    rule ih_rule {
        match {
            threat-level [ 7 8 9 10 ];
        }
        then {
            action {
                block {
                    close {
                        http {
                            message "Blocked!";
                        }
                    }
                }
            }
        }
    }
}

```

```

    }
  }
  log;
}
}
}
policy secintel1 {
  cc {
    cc_profile;
  }
  Infected-Hosts {
    ih_profile;
  }
}
}
}

```

If you are done configuring the feature on your device, enter `commit` from configuration mode.

Complete Device Configurations

IN THIS SECTION

- [Configuration on Leaf 1 Device | 861](#)
- [Configuration on Spine 1 Device | 863](#)
- [Configuration on Leaf 2 Device | 865](#)
- [Configuration on Spine 2 Device | 867](#)
- [Basic Tunnel Inspection Configuration on SRX Series Device | 868](#)
- [Tunnel Inspection Configuration on SRX Series Device with Layer 7 Security Services | 870](#)

Refer to these configurations to better understand or recreate the context of this example. They include the complete ERB based EVPN-VXLAN configurations for the QFX Series switches that form the DC fabrics, as well as the ending state of the SRX Series device for both the basic and advanced VXLAN tunnel inspection examples.

NOTE: The provided configurations do not show user login, system logging, or management related configuration as this varies by location is not related to the VXLAN tunnel inspection feature.

For more details and example on configuring EVPN-VXLAN, see the network configuration example at [Configuring an EVPN-VXLAN Fabric for a Campus Network with ERB](#).

Configuration on Leaf 1 Device

```
set system host-name r0_dc1_leaf1
set interfaces xe-0/0/0 mtu 9000
set interfaces xe-0/0/0 unit 0 family inet address 10.1.1.2/30
set interfaces xe-0/0/1 unit 0 family ethernet-switching vlan members v100
set interfaces xe-0/0/2 unit 0 family ethernet-switching vlan members v50
set interfaces irb unit 50 virtual-gateway-accept-data
set interfaces irb unit 50 family inet address 192.168.50.3/24 preferred
set interfaces irb unit 50 family inet address 192.168.50.3/24 virtual-gateway-address
192.168.50.1
set interfaces irb unit 100 virtual-gateway-accept-data
set interfaces irb unit 100 family inet address 192.168.100.3/24 preferred
set interfaces irb unit 100 family inet address 192.168.100.3/24 virtual-gateway-address
192.168.100.1
set interfaces lo0 unit 0 family inet address 10.255.1.10/32
set interfaces lo0 unit 1 family inet address 10.255.10.10/32
set forwarding-options vxlan-routing next-hop 32768
set forwarding-options vxlan-routing overlay-ecmp
set policy-options policy-statement ECMP-POLICY then load-balance per-packet
set policy-options policy-statement FROM_Lo0 term 10 from interface lo0.0
set policy-options policy-statement FROM_Lo0 term 10 then accept
set policy-options policy-statement FROM_Lo0 term 20 then reject
set policy-options policy-statement OVERLAY_IMPORT term 5 from community comm_pod1
set policy-options policy-statement OVERLAY_IMPORT term 5 then accept
set policy-options policy-statement OVERLAY_IMPORT term 10 from community comm_pod2
set policy-options policy-statement OVERLAY_IMPORT term 10 then accept
set policy-options policy-statement OVERLAY_IMPORT term 20 from community shared_100_fm_pod2
set policy-options policy-statement OVERLAY_IMPORT term 20 from community shared_100_fm_pod1
set policy-options policy-statement OVERLAY_IMPORT term 20 then accept
set policy-options policy-statement T5_EXPORT term fm_direct from protocol direct
set policy-options policy-statement T5_EXPORT term fm_direct then accept
set policy-options policy-statement T5_EXPORT term fm_static from protocol static
```



```

set policy-options policy-statement T5_EXPORT term fm_static then accept
set policy-options policy-statement T5_EXPORT term fm_v4_host from protocol evpn
set policy-options policy-statement T5_EXPORT term fm_v4_host from route-filter 0.0.0.0/0 prefix-
length-range /32-/32
set policy-options policy-statement T5_EXPORT term fm_v4_host then accept
set policy-options policy-statement VRF1_T5_RT_EXPORT term t1 then community add target_t5_pod1
set policy-options policy-statement VRF1_T5_RT_EXPORT term t1 then accept
set policy-options policy-statement VRF1_T5_RT_IMPORT term t1 from community target_t5_pod1
set policy-options policy-statement VRF1_T5_RT_IMPORT term t1 then accept
set policy-options policy-statement VRF1_T5_RT_IMPORT term t2 from community target_t5_pod2
set policy-options policy-statement VRF1_T5_RT_IMPORT term t2 then accept
set policy-options community comm_pod1 members target:65001:1
set policy-options community comm_pod2 members target:65002:2
set policy-options community shared_100_fm_pod1 members target:65001:100
set policy-options community shared_100_fm_pod2 members target:65002:100
set policy-options community target_t5_pod1 members target:65001:9999
set policy-options community target_t5_pod2 members target:65002:9999
set routing-instances TENANT_1_VRF routing-options multipath
set routing-instances TENANT_1_VRF protocols evpn ip-prefix-routes advertise direct-nexthop
set routing-instances TENANT_1_VRF protocols evpn ip-prefix-routes encapsulation vxlan
set routing-instances TENANT_1_VRF protocols evpn ip-prefix-routes vni 9999
set routing-instances TENANT_1_VRF protocols evpn ip-prefix-routes export T5_EXPORT
set routing-instances TENANT_1_VRF instance-type vrf
set routing-instances TENANT_1_VRF interface irb.50
set routing-instances TENANT_1_VRF interface irb.100
set routing-instances TENANT_1_VRF interface lo0.1
set routing-instances TENANT_1_VRF route-distinguisher 10.255.1.10:9999
set routing-instances TENANT_1_VRF vrf-import VRF1_T5_RT_IMPORT
set routing-instances TENANT_1_VRF vrf-export VRF1_T5_RT_EXPORT
set routing-instances TENANT_1_VRF vrf-table-label
set routing-options router-id 10.255.1.10
set routing-options autonomous-system 65001
set routing-options forwarding-table export ECMP-POLICY
set routing-options forwarding-table ecmp-fast-reroute
set routing-options forwarding-table chained-composite-next-hop ingress evpn
set protocols bgp group EVPN_FABRIC type internal
set protocols bgp group EVPN_FABRIC local-address 10.255.1.10
set protocols bgp group EVPN_FABRIC family evpn signaling
set protocols bgp group EVPN_FABRIC multipath
set protocols bgp group EVPN_FABRIC bfd-liveness-detection minimum-interval 1000
set protocols bgp group EVPN_FABRIC bfd-liveness-detection multiplier 3
set protocols bgp group EVPN_FABRIC neighbor 10.255.1.1
set protocols bgp group UNDERLAY type external

```

```

set protocols bgp group UNDERLAY family inet unicast
set protocols bgp group UNDERLAY export FROM_Lo0
set protocols bgp group UNDERLAY local-as 65510
set protocols bgp group UNDERLAY multipath multiple-as
set protocols bgp group UNDERLAY bfd-liveness-detection minimum-interval 350
set protocols bgp group UNDERLAY bfd-liveness-detection multiplier 3
set protocols bgp group UNDERLAY neighbor 10.1.1.1 peer-as 65511
set protocols evpn encapsulation vxlan
set protocols evpn default-gateway no-gateway-community
set protocols evpn vni-options vni 150 vrf-target target:65001:150
set protocols evpn vni-options vni 1100 vrf-target target:65001:100
set protocols evpn extended-vni-list 1100
set protocols evpn extended-vni-list 150
set protocols lldp interface all
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 10.255.1.10:1
set switch-options vrf-import OVERLAY_IMPORT
set switch-options vrf-target target:65001:1
set vlans v100 vlan-id 100
set vlans v100 l3-interface irb.100
set vlans v100 vxlan vni 1100
set vlans v50 vlan-id 50
set vlans v50 l3-interface irb.50
set vlans v50 vxlan vni 150

```

Configuration on Spine 1 Device

```

set system host-name r1_dc1_spine11
set interfaces xe-0/0/0 mtu 9000
set interfaces xe-0/0/0 unit 0 family inet address 10.1.1.1/30
set interfaces xe-0/0/1 mtu 9000
set interfaces xe-0/0/1 unit 0 family inet address 172.16.1.1/30
set interfaces lo0 unit 0 family inet address 10.255.1.1/32
set policy-options policy-statement ECMP-POLICY then load-balance per-packet
set policy-options policy-statement FROM_Lo0 term 10 from interface lo0.0
set policy-options policy-statement FROM_Lo0 term 10 then accept
set policy-options policy-statement FROM_Lo0 term 20 then reject
set policy-options policy-statement UNDERLAY-EXPORT term LOOPBACK from route-filter
10.255.0.0/16 orlonger
set policy-options policy-statement UNDERLAY-EXPORT term LOOPBACK from route-filter 10.1.0.0/16

```

```

orlonger
set policy-options policy-statement UNDERLAY-EXPORT term LOOPBACK then accept
set policy-options policy-statement UNDERLAY-EXPORT term DEFAULT then reject
set policy-options policy-statement UNDERLAY-IMPORT term LOOPBACK from route-filter
10.255.0.0/16 orlonger
set policy-options policy-statement UNDERLAY-IMPORT term LOOPBACK from route-filter 10.1.0.0/16
orlonger
set policy-options policy-statement UNDERLAY-IMPORT term LOOPBACK then accept
set policy-options policy-statement UNDERLAY-IMPORT term DEFAULT then reject
set routing-options autonomous-system 65001
set routing-options forwarding-table export ECMP-POLICY
set routing-options forwarding-table ecmp-fast-reroute
set protocols bgp group EVPN_FABRIC type internal
set protocols bgp group EVPN_FABRIC local-address 10.255.1.1
set protocols bgp group EVPN_FABRIC family evpn signaling
set protocols bgp group EVPN_FABRIC cluster 10.255.1.1
set protocols bgp group EVPN_FABRIC multipath
set protocols bgp group EVPN_FABRIC bfd-liveness-detection minimum-interval 1000
set protocols bgp group EVPN_FABRIC bfd-liveness-detection multiplier 3
set protocols bgp group EVPN_FABRIC neighbor 10.255.1.10
set protocols bgp group EVPN_FABRIC vpn-apply-export
set protocols bgp group UNDERLAY type external
set protocols bgp group UNDERLAY import UNDERLAY-IMPORT
set protocols bgp group UNDERLAY family inet unicast
set protocols bgp group UNDERLAY export UNDERLAY-EXPORT
set protocols bgp group UNDERLAY local-as 65511
set protocols bgp group UNDERLAY multipath multiple-as
set protocols bgp group UNDERLAY bfd-liveness-detection minimum-interval 350
set protocols bgp group UNDERLAY bfd-liveness-detection multiplier 3
set protocols bgp group UNDERLAY neighbor 10.1.1.2 peer-as 65510
set protocols bgp group UNDERLAY neighbor 172.16.1.2 peer-as 65012
set protocols bgp group OVERLAY_INTERDC type external
set protocols bgp group OVERLAY_INTERDC multihop no-nexthop-change
set protocols bgp group OVERLAY_INTERDC local-address 10.255.1.1
set protocols bgp group OVERLAY_INTERDC family evpn signaling
set protocols bgp group OVERLAY_INTERDC multipath multiple-as
set protocols bgp group OVERLAY_INTERDC neighbor 10.255.2.1 peer-as 65002
set protocols lldp interface all

```

Configuration on Leaf 2 Device

```

set system host-name r2_dc2_leaf1
set interfaces xe-0/0/0 mtu 9000
set interfaces xe-0/0/0 unit 0 family inet address 10.1.2.2/30
set interfaces xe-0/0/1 unit 0 family ethernet-switching vlan members v100
set interfaces xe-0/0/2 unit 0 family ethernet-switching vlan members v60
set interfaces irb unit 60 virtual-gateway-accept-data
set interfaces irb unit 60 family inet address 192.168.60.3/24 preferred
set interfaces irb unit 60 family inet address 192.168.60.3/24 virtual-gateway-address
192.168.60.1
set interfaces irb unit 100 virtual-gateway-accept-data
set interfaces irb unit 100 family inet address 192.168.100.4/24 preferred
set interfaces irb unit 100 family inet address 192.168.100.4/24 virtual-gateway-address
192.168.100.1
set interfaces lo0 unit 0 family inet address 10.255.2.10/32
set interfaces lo0 unit 1 family inet address 10.255.20.10/32
set forwarding-options vxlan-routing next-hop 32768
set forwarding-options vxlan-routing overlay-ecmp
set policy-options policy-statement ECMP-POLICY then load-balance per-packet
set policy-options policy-statement FROM_Lo0 term 10 from interface lo0.0
set policy-options policy-statement FROM_Lo0 term 10 then accept
set policy-options policy-statement FROM_Lo0 term 20 then reject
set policy-options policy-statement OVERLAY_IMPORT term 5 from community comm_pod1
set policy-options policy-statement OVERLAY_IMPORT term 5 then accept
set policy-options policy-statement OVERLAY_IMPORT term 10 from community comm_pod2
set policy-options policy-statement OVERLAY_IMPORT term 10 then accept
set policy-options policy-statement OVERLAY_IMPORT term 20 from community shared_100_fm_pod2
set policy-options policy-statement OVERLAY_IMPORT term 20 from community shared_100_fm_pod1
set policy-options policy-statement OVERLAY_IMPORT term 20 then accept
set policy-options policy-statement T5_EXPORT term fm_direct from protocol direct
set policy-options policy-statement T5_EXPORT term fm_direct then accept
set policy-options policy-statement T5_EXPORT term fm_static from protocol static
set policy-options policy-statement T5_EXPORT term fm_static then accept
set policy-options policy-statement T5_EXPORT term fm_v4_host from protocol evpn
set policy-options policy-statement T5_EXPORT term fm_v4_host from route-filter 0.0.0.0/0 prefix-
length-range /32-/32
set policy-options policy-statement T5_EXPORT term fm_v4_host then accept
set policy-options policy-statement VRF1_T5_RT_EXPORT term t1 then community add target_t5_pod1
set policy-options policy-statement VRF1_T5_RT_EXPORT term t1 then accept
set policy-options policy-statement VRF1_T5_RT_IMPORT term t1 from community target_t5_pod1

```

```

set policy-options policy-statement VRF1_T5_RT_IMPORT term t1 then accept
set policy-options policy-statement VRF1_T5_RT_IMPORT term t2 from community target_t5_pod2
set policy-options policy-statement VRF1_T5_RT_IMPORT term t2 then accept
set policy-options community comm_pod1 members target:65001:1
set policy-options community comm_pod2 members target:65002:2
set policy-options community shared_100_fm_pod1 members target:65001:100
set policy-options community shared_100_fm_pod2 members target:65002:100
set policy-options community target_t5_pod1 members target:65001:9999
set policy-options community target_t5_pod2 members target:65002:9999
set routing-instances TENANT_1_VRF routing-options multipath
set routing-instances TENANT_1_VRF protocols evpn ip-prefix-routes advertise direct-nexthop
set routing-instances TENANT_1_VRF protocols evpn ip-prefix-routes encapsulation vxlan
set routing-instances TENANT_1_VRF protocols evpn ip-prefix-routes vni 9999
set routing-instances TENANT_1_VRF protocols evpn ip-prefix-routes export T5_EXPORT
set routing-instances TENANT_1_VRF instance-type vrf
set routing-instances TENANT_1_VRF interface irb.60
set routing-instances TENANT_1_VRF interface irb.100
set routing-instances TENANT_1_VRF interface lo0.1
set routing-instances TENANT_1_VRF route-distinguisher 10.255.1.2:9999
set routing-instances TENANT_1_VRF vrf-import VRF1_T5_RT_IMPORT
set routing-instances TENANT_1_VRF vrf-export VRF1_T5_RT_EXPORT
set routing-instances TENANT_1_VRF vrf-table-label
set routing-options router-id 10.255.2.10
set routing-options autonomous-system 65002
set routing-options forwarding-table export ECMP-POLICY
set routing-options forwarding-table ecmp-fast-reroute
set routing-options forwarding-table chained-composite-next-hop ingress evpn
set protocols bgp group EVPN_FABRIC type internal
set protocols bgp group EVPN_FABRIC local-address 10.255.2.10
set protocols bgp group EVPN_FABRIC family evpn signaling
set protocols bgp group EVPN_FABRIC multipath
set protocols bgp group EVPN_FABRIC bfd-liveness-detection minimum-interval 1000
set protocols bgp group EVPN_FABRIC bfd-liveness-detection multiplier 3
set protocols bgp group EVPN_FABRIC neighbor 10.255.2.1
set protocols bgp group UNDERLAY type external
set protocols bgp group UNDERLAY family inet unicast
set protocols bgp group UNDERLAY export FROM_Lo0
set protocols bgp group UNDERLAY local-as 65522
set protocols bgp group UNDERLAY multipath multiple-as
set protocols bgp group UNDERLAY bfd-liveness-detection minimum-interval 350
set protocols bgp group UNDERLAY bfd-liveness-detection multiplier 3
set protocols bgp group UNDERLAY neighbor 10.1.2.1 peer-as 65523
set protocols evpn encapsulation vxlan

```

```

set protocols evpn default-gateway no-gateway-community
set protocols evpn vni-options vni 160 vrf-target target:65002:160
set protocols evpn vni-options vni 1100 vrf-target target:65002:100
set protocols evpn extended-vni-list 1100
set protocols evpn extended-vni-list 160
set protocols lldp interface all
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 10.255.2.10:1
set switch-options vrf-import OVERLAY_IMPORT
set switch-options vrf-target target:65002:1
set vlans v100 vlan-id 100
set vlans v100 l3-interface irb.100
set vlans v100 vxlan vni 1100
set vlans v60 vlan-id 60
set vlans v60 l3-interface irb.60
set vlans v60 vxlan vni 160

```

Configuration on Spine 2 Device

```

set system host-name r3_dc2_spine1
set interfaces xe-0/0/0 mtu 9000
set interfaces xe-0/0/0 unit 0 family inet address 10.1.2.1/30
set interfaces xe-0/0/1 mtu 9000
set interfaces xe-0/0/1 unit 0 family inet address 172.16.2.1/30
set interfaces lo0 unit 0 family inet address 10.255.2.1/32
set policy-options policy-statement ECMP-POLICY then load-balance per-packet
set policy-options policy-statement FROM_Lo0 term 10 from interface lo0.0
set policy-options policy-statement FROM_Lo0 term 10 then accept
set policy-options policy-statement FROM_Lo0 term 20 then reject
set policy-options policy-statement UNDERLAY-EXPORT term LOOPBACK from route-filter
10.255.0.0/16 orlonger
set policy-options policy-statement UNDERLAY-EXPORT term LOOPBACK from route-filter 10.1.0.0/16
orlonger
set policy-options policy-statement UNDERLAY-EXPORT term LOOPBACK then accept
set policy-options policy-statement UNDERLAY-EXPORT term DEFAULT then reject
set policy-options policy-statement UNDERLAY-IMPORT term LOOPBACK from route-filter
10.255.0.0/16 orlonger
set policy-options policy-statement UNDERLAY-IMPORT term LOOPBACK from route-filter 10.1.0.0/16
orlonger
set policy-options policy-statement UNDERLAY-IMPORT term LOOPBACK then accept

```

```

set policy-options policy-statement UNDERLAY-IMPORT term DEFAULT then reject
set routing-options autonomous-system 65002
set routing-options forwarding-table export ECMP-POLICY
set routing-options forwarding-table ecmp-fast-reroute
set protocols bgp group EVPN_FABRIC type internal
set protocols bgp group EVPN_FABRIC local-address 10.255.2.1
set protocols bgp group EVPN_FABRIC family evpn signaling
set protocols bgp group EVPN_FABRIC cluster 10.255.2.1
set protocols bgp group EVPN_FABRIC multipath
set protocols bgp group EVPN_FABRIC bfd-liveness-detection minimum-interval 1000
set protocols bgp group EVPN_FABRIC bfd-liveness-detection multiplier 3
set protocols bgp group EVPN_FABRIC neighbor 10.255.2.10
set protocols bgp group EVPN_FABRIC vpn-apply-export
set protocols bgp group UNDERLAY type external
set protocols bgp group UNDERLAY import UNDERLAY-IMPORT
set protocols bgp group UNDERLAY family inet unicast
set protocols bgp group UNDERLAY export UNDERLAY-EXPORT
set protocols bgp group UNDERLAY local-as 65523
set protocols bgp group UNDERLAY multipath multiple-as
set protocols bgp group UNDERLAY bfd-liveness-detection minimum-interval 350
set protocols bgp group UNDERLAY bfd-liveness-detection multiplier 3
set protocols bgp group UNDERLAY neighbor 10.1.2.2 peer-as 65522
set protocols bgp group UNDERLAY neighbor 172.16.2.2 peer-as 65012
set protocols bgp group OVERLAY_INTERDC type external
set protocols bgp group OVERLAY_INTERDC multihop no-nexthop-change
set protocols bgp group OVERLAY_INTERDC local-address 10.255.2.1
set protocols bgp group OVERLAY_INTERDC family evpn signaling
set protocols bgp group OVERLAY_INTERDC multipath multiple-as
set protocols bgp group OVERLAY_INTERDC neighbor 10.255.1.1 peer-as 65001
set protocols lldp interface all

```

Basic Tunnel Inspection Configuration on SRX Series Device

```

set system host-name r4-dci-ubr
set security address-book global address vtep-untrust 10.255.2.0/24
set security address-book global address vtep-trust 10.255.1.0/24
set security address-book global address vlan100 192.168.100.0/24
set security policies from-zone trust to-zone untrust policy P1 match source-address vtep-trust
set security policies from-zone trust to-zone untrust policy P1 match destination-address vtep-untrust
set security policies from-zone trust to-zone untrust policy P1 match application junos-vxlan

```

```

set security policies from-zone trust to-zone untrust policy P1 then permit tunnel-inspection
TP-1
set security policies from-zone trust to-zone untrust policy accept-rest match source-address any
set security policies from-zone trust to-zone untrust policy accept-rest match destination-
address any
set security policies from-zone trust to-zone untrust policy accept-rest match application any
set security policies from-zone trust to-zone untrust policy accept-rest then permit
set security policies from-zone untrust to-zone trust policy accept-return match source-address
any
set security policies from-zone untrust to-zone trust policy accept-return match destination-
address any
set security policies from-zone untrust to-zone trust policy accept-return match application any
set security policies from-zone untrust to-zone trust policy accept-return then permit
set security policies policy-set PSET-1 policy PSET-1-P1 match source-address vlan100
set security policies policy-set PSET-1 policy PSET-1-P1 match destination-address vlan100
set security policies policy-set PSET-1 policy PSET-1-P1 match application junos-icmp-all
set security policies policy-set PSET-1 policy PSET-1-P1 then permit
set security zones security-zone trust host-inbound-traffic system-services all
set security zones security-zone trust host-inbound-traffic protocols all
set security zones security-zone trust interfaces ge-0/0/0.0
set security zones security-zone untrust host-inbound-traffic system-services all
set security zones security-zone untrust host-inbound-traffic protocols all
set security zones security-zone untrust interfaces ge-0/0/1.0
set security tunnel-inspection inspection-profile TP-1 vxlan VNI-1100 policy-set PSET-1
set security tunnel-inspection inspection-profile TP-1 vxlan VNI-1100 vni VLAN-100
set security tunnel-inspection vni VLAN-100 vni-id 1100
set interfaces ge-0/0/0 description "Link to DC2 Spine 1"
set interfaces ge-0/0/0 mtu 9000
set interfaces ge-0/0/0 unit 0 family inet address 172.16.1.2/30
set interfaces ge-0/0/1 mtu 9000
set interfaces ge-0/0/1 unit 0 family inet address 172.16.2.2/30
set policy-options policy-statement ECMP-POLICY then load-balance per-packet
set policy-options policy-statement dci term 1 from protocol direct
set policy-options policy-statement dci term 1 then accept
set protocols bgp group UNDERLAY export dci
set protocols bgp group UNDERLAY multipath multiple-as
set protocols bgp group UNDERLAY bfd-liveness-detection minimum-interval 350
set protocols bgp group UNDERLAY bfd-liveness-detection multiplier 3
set protocols bgp group UNDERLAY neighbor 172.16.1.1 peer-as 65511
set protocols bgp group UNDERLAY neighbor 172.16.2.1 peer-as 65523
set routing-options autonomous-system 65012
set routing-options forwarding-table export ECMP-POLICY

```


Tunnel Inspection Configuration on SRX Series Device with Layer 7 Security Services

```

set system host-name r4-dci-ebr
set services application-identification
set services ssl initiation profile aamw-ssl
set services ssl proxy profile ssl-inspect-profile-1 root-ca VJSA
set services advanced-anti-malware policy P3 http inspection-profile scripts
set services advanced-anti-malware policy P3 http action block
set services advanced-anti-malware policy P3 http client-notify message "AAMW Blocked!"
set services advanced-anti-malware policy P3 http notification log
set services advanced-anti-malware policy P3 verdict-threshold recommended
set services advanced-anti-malware policy P3 fallback-options action permit
set services advanced-anti-malware policy P3 fallback-options notification log
set services security-intelligence url https://cloudfeeds.argonqa.junipersecurity.net/api/manifest.xml
set services security-intelligence authentication tls-profile aamw-ssl
set services security-intelligence profile cc_profile category CC
set services security-intelligence profile cc_profile rule cc_rule match threat-level 1
set services security-intelligence profile cc_profile rule cc_rule match threat-level 2
set services security-intelligence profile cc_profile rule cc_rule match threat-level 4
set services security-intelligence profile cc_profile rule cc_rule match threat-level 5
set services security-intelligence profile cc_profile rule cc_rule match threat-level 6
set services security-intelligence profile cc_profile rule cc_rule match threat-level 7
set services security-intelligence profile cc_profile rule cc_rule match threat-level 8
set services security-intelligence profile cc_profile rule cc_rule match threat-level 9
set services security-intelligence profile cc_profile rule cc_rule match threat-level 10
set services security-intelligence profile cc_profile rule cc_rule then action block close
set services security-intelligence profile cc_profile rule cc_rule then log
set services security-intelligence profile ih_profile category Infected-Hosts
set services security-intelligence profile ih_profile rule ih_rule match threat-level 7
set services security-intelligence profile ih_profile rule ih_rule match threat-level 8
set services security-intelligence profile ih_profile rule ih_rule match threat-level 9
set services security-intelligence profile ih_profile rule ih_rule match threat-level 10
set services security-intelligence profile ih_profile rule ih_rule then action block close http
message "Blocked!"
set services security-intelligence profile ih_profile rule ih_rule then log
set services security-intelligence policy secintel1 CC cc_profile
set services security-intelligence policy secintel1 Infected-Hosts ih_profile
set security pki ca-profile aamw-ca ca-identity deviceCA
set security pki ca-profile aamw-ca enrollment url http://ca.junipersecurity.net:8080/ejbca/publicweb/apply/scep/SRX/pkiclient.exe
set security pki ca-profile aamw-ca revocation-check disable
set security pki ca-profile aamw-ca revocation-check crl url http://va.junipersecurity.net/ca/

```

```

deviceCA.crl
set security pki ca-profile aamw-secintel-ca ca-identity JUNIPER
set security pki ca-profile aamw-secintel-ca revocation-check crl url http://
va.junipersecurity.net/ca/current.crl
set security pki ca-profile aamw-cloud-ca ca-identity JUNIPER_CLOUD
set security pki ca-profile aamw-cloud-ca revocation-check crl url http://
va.junipersecurity.net/ca/cloudCA.crl
set security idp idp-policy idp123 rulebase-ips rule rule1 match application junos-icmp-all
set security idp idp-policy idp123 rulebase-ips rule rule1 then action no-action
set security address-book global address vtep-untrust 10.255.2.0/24
set security address-book global address vtep-trust 10.255.1.0/24
set security address-book global address vlan100 192.168.100.0/24
set security utm default-configuration anti-virus type sophos-engine
set security utm utm-policy P1 anti-virus http-profile junos-sophos-av-defaults
set security policies from-zone trust to-zone untrust policy P1 match source-address vtep-trust
set security policies from-zone trust to-zone untrust policy P1 match destination-address vtep-
untrust
set security policies from-zone trust to-zone untrust policy P1 match application junos-vxlan
set security policies from-zone trust to-zone untrust policy P1 then permit tunnel-inspection
TP-1
set security policies from-zone trust to-zone untrust policy accept-rest match source-address any
set security policies from-zone trust to-zone untrust policy accept-rest match destination-
address any
set security policies from-zone trust to-zone untrust policy accept-rest match application any
set security policies from-zone trust to-zone untrust policy accept-rest then permit
set security policies from-zone untrust to-zone trust policy accept-return match source-address
any
set security policies from-zone untrust to-zone trust policy accept-return match destination-
address any
set security policies from-zone untrust to-zone trust policy accept-return match application any
set security policies from-zone untrust to-zone trust policy accept-return then permit
set security policies policy-set PSET-1 policy PSET-1-P1 match source-address vlan100
set security policies policy-set PSET-1 policy PSET-1-P1 match destination-address vlan100
set security policies policy-set PSET-1 policy PSET-1-P1 match application any
set security policies policy-set PSET-1 policy PSET-1-P1 match dynamic-application any
set security policies policy-set PSET-1 policy PSET-1-P1 match url-category any
set security policies policy-set PSET-1 policy PSET-1-P1 match from-zone trust
set security policies policy-set PSET-1 policy PSET-1-P1 match to-zone untrust
set security policies policy-set PSET-1 policy PSET-1-P1 then permit application-services idp-
policy idp123
set security policies policy-set PSET-1 policy PSET-1-P1 then permit application-services ssl-
proxy profile-name ssl-inspect-profile-1
set security policies policy-set PSET-1 policy PSET-1-P1 then permit application-services utm-

```

```

policy P1
set security policies policy-set PSET-1 policy PSET-1-P1 then permit application-services
security-intelligence-policy secintel1
set security policies policy-set PSET-1 policy PSET-1-P1 then permit application-services
advanced-anti-malware-policy P3
set security zones security-zone trust host-inbound-traffic system-services all
set security zones security-zone trust host-inbound-traffic protocols all
set security zones security-zone trust interfaces ge-0/0/0.0
set security zones security-zone untrust host-inbound-traffic system-services all
set security zones security-zone untrust host-inbound-traffic protocols all
set security zones security-zone untrust interfaces ge-0/0/1.0
set security tunnel-inspection inspection-profile TP-1 vxlan VNI-1100 policy-set PSET-1
set security tunnel-inspection inspection-profile TP-1 vxlan VNI-1100 vni VLAN-100
set security tunnel-inspection vni VLAN-100 vni-id 1100
set interfaces ge-0/0/0 description "Link to DC2 Spine 1"
set interfaces ge-0/0/0 mtu 9000
set interfaces ge-0/0/0 unit 0 family inet address 172.16.1.2/30
set interfaces ge-0/0/1 mtu 9000
set interfaces ge-0/0/1 unit 0 family inet address 172.16.2.2/30
set policy-options policy-statement ECMP-POLICY then load-balance per-packet
set policy-options policy-statement dci term 1 from protocol direct
set policy-options policy-statement dci term 1 then accept
set protocols bgp group UNDERLAY export dci
set protocols bgp group UNDERLAY multipath multiple-as
set protocols bgp group UNDERLAY bfd-liveness-detection minimum-interval 350
set protocols bgp group UNDERLAY bfd-liveness-detection multiplier 3
set protocols bgp group UNDERLAY neighbor 172.16.1.1 peer-as 65511
set protocols bgp group UNDERLAY neighbor 172.16.2.1 peer-as 65523
set routing-options autonomous-system 65012
set routing-options static route 0.0.0.0/0 next-hop 10.9.159.252
set routing-options forwarding-table export ECMP-POLICY

```

3

PART

EVPN-MPLS

[Overview | 874](#)

[Convergence in an EVPN MPLS Network | 891](#)

[Pseudowire Termination at an EVPN | 893](#)

[Configuring the Distribution of Routes | 900](#)

[Configuring VLAN Services and Virtual Switch Support | 912](#)

[Configuring Integrated Bridging and Routing | 946](#)

[Configuring IGMP or MLD Snooping with EVPN-MPLS | 1013](#)

CHAPTER 21

Overview

IN THIS CHAPTER

- [EVPN Overview | 874](#)
- [EVPN-MPLS Caveats | 877](#)
- [EVPN Overview for Switches | 878](#)
- [Supported EVPN Standards | 879](#)
- [Migrating from FEC128 LDP-VPLS to EVPN Overview | 881](#)

EVPN Overview

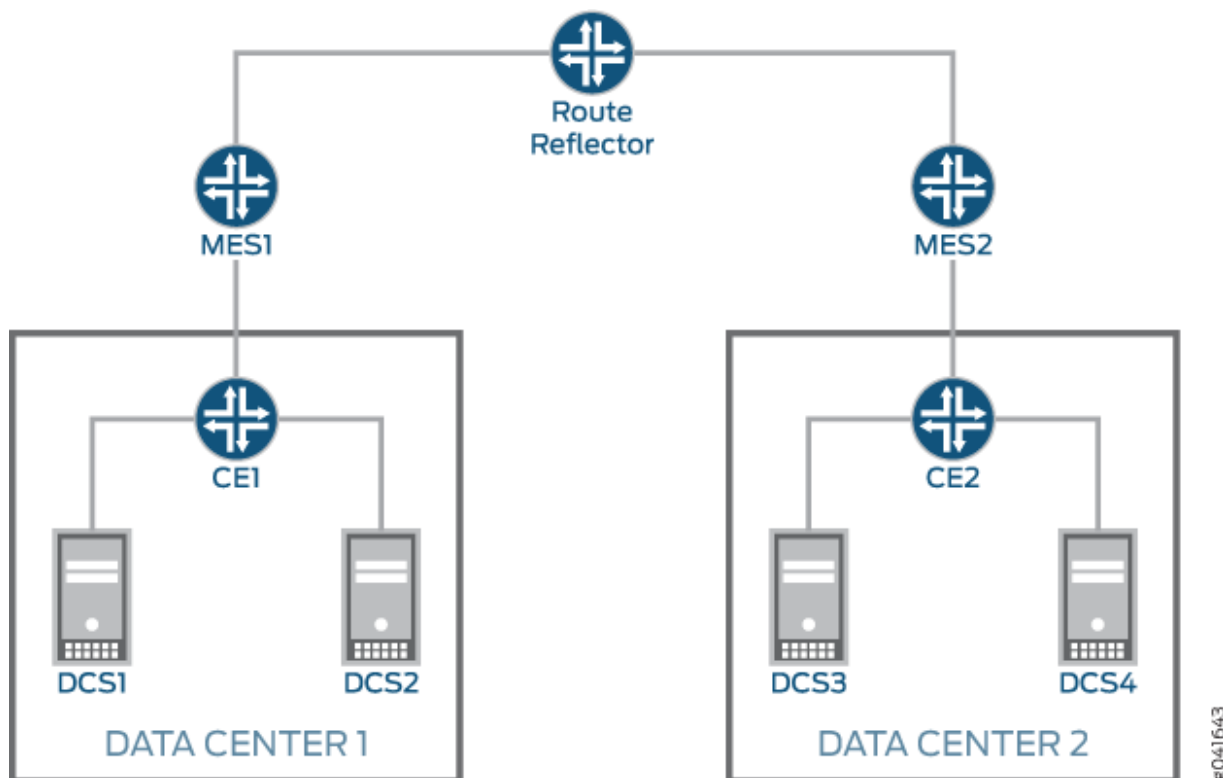
IN THIS SECTION

- [EVPN MPLS Features Supported by QFX10000 Switches | 876](#)

An Ethernet VPN (EVPN) enables you to connect dispersed customer sites using a Layer 2 virtual bridge. As with other types of VPNs, an EVPN consists of customer edge (CE) devices (host, router, or switch) connected to provider edge (PE) routers. The PE routers can include an MPLS edge switch (MES) that acts at the edge of the MPLS infrastructure. Either an MX Series 5G Universal Routing Platform or a standalone switch can be configured to act as an MES. You can deploy multiple EVPNs within a service provider network, each providing network connectivity to a customer while ensuring that the traffic sharing on that network remains private. [Figure 88 on page 875](#) illustrates a typical EVPN deployment.

Traffic from Data Center 1 is transported over the service provider's network through MES1 to MES2 and then onto Data Center 2. DCS1, DCS2, DCS3, and DCS4 are the data center switches.

Figure 88: EVPN Connecting Data Center 1 and Data Center 2



The MESs are interconnected within the service provider's network using label-switched paths (LSPs). The MPLS infrastructure allows you to take advantage of the MPLS functionality provided by the Junos OS, including fast reroute, node and link protection, and standby secondary paths. For EVPNs, learning between MESs takes place in the control plane rather than in the data plane (as is the case with traditional network bridging). The control plane provides greater control over the learning process, allowing you to restrict which devices discover information about the network. You can also apply policies on the MESs, allowing you to carefully control how network information is distributed and processed. EVPNs utilize the BGP control plane infrastructure, providing greater scale and the ability to isolate groups of devices (hosts, servers, virtual machines, and so on) from each other.

The MESs attach an MPLS label to each MAC address learned from the CE devices. This label and MAC address combination is advertised to the other MESs in the control plane. Control plane learning enables load balancing and improves convergence times in the event of certain types of network failures. The learning process between the MESs and the CE devices is completed using the method best suited to each CE device (data plane learning, IEEE 802.1, LLDP, 802.1aq, and so on).

The policy attributes of an EVPN are similar to an IP VPN (for example, Layer 3 VPNs). Each EVPN routing instance requires that you configure a route distinguisher (RD) and one or more route targets (RTs). A CE device attaches to an EVPN routing instance on an MES through an Ethernet interface that might be configured for one or more VLANs.

The following features are available for EVPNs:

- Ethernet connectivity between data centers spanning metropolitan area networks (*MANs*) and *WANs*
- One VLAN for each MAC VPN
- Automatic RDs
- Dual-homed EVPN connection with active/standby multihoming
- The following Juniper Networks devices support active/active multihoming:
 - Starting in Junos OS Releases 14.2R6 and 16.1R1, MX Series routers. It is not supported in Junos OS Release 15.1.
 - Starting in Junos OS Releases 16.1R4 and 16.2R2, EX9200 switches.
- Ethernet VPN (EVPN) support, including EVPN-MPLS, EVPN + VXLAN, and PBB EVPN, has been extended to logical systems running only on MX devices. The same EVPN options and performance that are available in the default EVPN instance are available in a logical system. Configure EVPN on a logical system under the [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols evpn] hierarchy.

The following feature is not supported for EVPNs:

- Graceful restart, graceful Routing Engine switchover (GRES), and nonstop active routing (NSR) is not supported in releases prior to Junos OS Release 16.1.

EVPN MPLS Features Supported by QFX10000 Switches

Starting in Junos OS 17.4R1, QFX10000 switches support EVPN with MPLS as its data plane, as defined in [RFC 7432](#). The following features are supported:

- Layer 2 VLANs using the default-switch routing instance. An EVPN instance (EVI) is not supported.
- EVPN MPLS multihoming active-active support
- VLAN-aware bundle service interface without translation (support is limited to 4K VLANs as only the default-switching instance is supported)
- Ingress multicast replication

- Mac mobility

Release History Table

Release	Description
17.4	Ethernet VPN (EVPN) support, including EVPN-MPLS, EVPN + VXLAN, and PBB EVPN, has been extended to logical systems running only on MX devices. The same EVPN options and performance that are available in the default EVPN instance are available in a logical system. Configure EVPN on a logical system under the [edit logical-systems logical-system-name routing-instances routing-instance-name protocols evpn] hierarchy.

RELATED DOCUMENTATION

[Supported EVPN Standards | 879](#)

[EVPN Multihoming Overview | 96](#)

[Overview of MAC Mobility | 16](#)

EVPN-MPLS Caveats

IN THIS SECTION

- [Caveats for ACX7100-32C and ACX7100-48L Devices | 877](#)

Caveats for ACX7100-32C and ACX7100-48L Devices

The following EVPN-MPLS features are not supported on ACX7100-32C and ACX7100-48L devices running Junos OS Evolved Release 21.3R1.

- VLAN-aware bundle service
- MAC address pinning
- Access and trunk interfaces
- ERPS over EVPN
- EVPN, virtual-switch, and default-switch instances types

- EVPN with P2MP (point-to-multipoint) LSPs (link-state protocols)
- PBB-EVPN
- EVPN NSR

Also, If there are multiple paths from an ingress EVPN PE (provider edge) to any remote EVPN PE, then load balancing FRR (free-range routing) over those paths is not supported on BUM traffic.

EVPN Overview for Switches

An Ethernet VPN (EVPN) enables you to connect a group of dispersed customer sites using a Layer 2 virtual bridge. As with other types of VPNs, an EVPN comprises customer edge (CE) devices (host, router, or switch) connected to provider edge (PE) devices. The PE devices can include an MPLS edge switch (MES) that acts at the edge of the MPLS infrastructure. For the initial deployment of EVPNs using Juniper Networks equipment, you can configure an EX9200 switch to act as an MES. You can deploy multiple EVPNs within the network, each providing network connectivity to customers while ensuring that traffic sharing that network remains private.

The MESs are interconnected within the network by using label-switched paths (LSPs). The MPLS infrastructure allows you to take advantage of the MPLS functionality provided by the Junos operating system (Junos OS), including fast reroute, node and link protection, and standby secondary paths. For EVPNs, learning between MESs takes place in the control plane rather than in the data plane (as is the case with traditional network bridging). The control plane provides greater control over the learning process, allowing you to restrict which devices discover information about the network. You can also apply policies on the MESs, allowing you to carefully control how network information is distributed and processed. EVPNs utilize the BGP control plane infrastructure, providing greater scale and the ability to isolate groups of devices (hosts, servers, virtual machines, and so on) from each other.

The MESs attach an MPLS label to each MAC address learned from the CE devices. This label and MAC address combination is advertised to the other MESs in the control plane. Control plane learning enables load balancing and improves convergence times in the event of certain types of network failures. The learning process between the MESs and the CE devices is completed using the method best suited to each CE device (data plane learning, IEEE 802.1, LLDP, 802.1aq, and so on).

The policy attributes of an EVPN are similar to an IP VPN (for example, Layer 3 VPNs). Each EVPN routing instance requires that you configure a route distinguisher and one or more route targets. A CE device attaches to an EVPN routing instance on an MES through an Ethernet interface that might be configured for one or more VLANs.

The following features are available for EVPNs:

- Ethernet connectivity between data centers spanning metropolitan area networks (*MANs*) and *WANs*

- One or more VLANs for each MAC VPN
- Automatic route distinguishers
- Dual-homed EVPN connection with active standby multihoming
- Starting with Junos OS Releases 16.1R4 and 16.2R2, the active-active mode for EVPN multihoming is supported.
- Starting with Junos OS Release 17.3R1, both pure type-5 routes and standard type-5 routes are supported on EX9200 switches. Use this feature, which advertises IP prefixes through EVPN, when the Layer 2 domain does not exist at the remote data centers or metro network peering points. For more information about how to configure, see ["ip-prefix-routes" on page 1606](#).
- Starting with Junos OS OS Release 17.3R1, VXLAN encapsulation is supported. Previously, only MPLS encapsulation is supported.

The following features are not supported for EVPNs:

- Graceful restart, *graceful Routing Engine switchover (GRES)*, and *nonstop active routing (NSR)*

Release History Table

Release	Description
17.3R1	Starting with Junos OS Release 17.3R1, both pure type-5 routes and standard type-5 routes are supported on EX9200 switches.
17.3R1	Starting with Junos OS OS Release 17.3R1, VXLAN encapsulation is supported. Previously, only MPLS encapsulation is supported.
16.1R4	Starting with Junos OS Releases 16.1R4 and 16.2R2, the active-active mode for EVPN multihoming is supported.

RELATED DOCUMENTATION

[Supported EVPN Standards | 879](#)

[Example: Configuring EVPN Active-Active Multihoming | 213](#)

Supported EVPN Standards

RFCs and Internet drafts that define standards for EVPNs:

- RFC 4364, *BGP/MPLS IP Virtual Private Networks (VPNs)*
- RFC 4761, *Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling*
- RFC 7209, *Requirements for Ethernet VPN (EVPN)*
- RFC 7432, *BGP MPLS-Based Ethernet VPN*

The following features are not supported:

- Automatic derivation of Ethernet segment (ES) values. Only static ES configurations are supported.
- Host proxy ARP.
- RFC 7623, *Provider Backbone Bridging Combined with Ethernet VPN (PBB-EVPN)*
- RFC 8214, *Virtual Private Wire Service Support in Ethernet VPN*
- RFC 8317, *Ethernet-Tree (E-Tree) Support in Ethernet VPN (EVPN) and Provider Backbone Bridging EVPN (PBB-EVPN)*
- RFC 8365, *A Network Virtualization Overlay Solution Using Ethernet VPN (EVPN)*
- Internet draft draft-ietf-bess-evpn-igmp-ml-d-proxy, *IGMP and MLD Proxy for EVPN*
- Internet draft draft-ietf-bess-evpn-inter-subnet-forwarding, *Integrated Routing and Bridging in EVPN*
- Internet draft draft-ietf-bess-evpn-na-flags, *Propagation of IPv6 Neighbor Advertisement Flags in EVPN*
- Internet draft draft-ietf-bess-evpn-oam-req-frmwk, *EVPN Operations, Administration and Maintenance Requirements and Framework*
- Internet draft draft-ietf-bess-evpn-optimized-ir, *Optimized Ingress Replication solution for EVPN*
- Internet draft draft-ietf-bess-evpn-pref-df, *Preference-based EVPN DF Election*
- Internet draft draft-ietf-bess-evpn-prefix-advertisement, *IP Prefix Advertisement in EVPN*
- Internet draft draft-ietf-bess-evpn-proxy-arp-nd, *Operational Aspects of Proxy-ARP/ND in EVPN Networks*
- Internet draft draft-ietf-bess-evpn-virtual-eth-segment, *EVPN Virtual Ethernet Segment*
- Internet draft draft-ietf-bess-evpn-vpls-seamless-integ, *(PBB-)EVPN Seamless Integration with (PBB-)VPLS*
- Internet draft draft-ietf-bess-evpn-vpws-fxc, *EVPN VPWS Flexible Cross-Connect Service*

- Internet draft draft-ietf-bess-evpn-yang, *Yang Data Model for EVPN*
- Internet draft draft-ietf-isis-segment-routing-extensions-13, *IS-IS Extensions for Segment Routing*
- Internet draft draft-ietf-spring-segment-routing-13, *Segment Routing Architecture*
- Internet draft draft-ietf-spring-segment-routing-mpls-11, *Segment Routing with MPLS data plane*
- Internet draft draft-wsv-bess-extended-evpn-optimized-ir, *Extended Procedures for EVPN Optimized Ingress Replication*

RELATED DOCUMENTATION

[EVPN Overview | 874](#)

[Accessing Standards Documents on the Internet](#)

Migrating from FEC128 LDP-VPLS to EVPN Overview

IN THIS SECTION

- [Technology Overview and Benefits | 882](#)
- [FEC128 LDP-VPLS to EVPN Migration | 882](#)
- [Sample Configuration for LDP-VPLS to EVPN Migration | 884](#)
- [Reverting to VPLS | 888](#)
- [LDP-VPLS to EVPN Migration and Other Features | 888](#)

For service providers with virtual private LAN service (VPLS) networks and Ethernet VPN (EVPN) networks, there is a need to interconnect these networks. Prior to Junos OS Release 17.3, a logical tunnel interface on the interconnection point of the VPLS and EVPN routing instances was used for this purpose. In this case, the provider edge (PE) devices in each network were unaware of the PE devices in the other technology network. Starting in Junos OS Release 17.3, a solution is introduced for enabling staged migration from FEC128 LDP-VPLS toward EVPN on a site-by-site basis for every VPN routing instance. In this solution, the PE devices running EVPN and VPLS for the same VPN routing instance and single-homed segments can coexist. During migration, there is minimal impact to the customer edge (CE) device-to-CE device traffic forwarding for affected customers.

The following sections describe the migration from LDP-VPLS to EVPN:

Technology Overview and Benefits

Virtual private LAN service (VPLS) is an Ethernet-based point-to-multipoint Layer 2 VPN. This technology allows you to connect geographically dispersed data center LANs to each other across an MPLS backbone while maintaining Layer 2 connectivity. The high availability features defined in VPLS standards (such as LER dual homing) and topology autodiscovery features using BGP signaling make VPLS scalable and easy to deploy. Because VPLS uses MPLS as its core, it provides low latency variation and statistically bound low convergence times within the MPLS network.

Ethernet VPN (EVPN), on the other hand, is a combined Layer 2 and Layer 3 VPN solution that is more scalable, resilient, and efficient than current technologies. It provides several benefits including greater network efficiency, reliability, scalability, virtual machine (VM) mobility, and policy control for service providers and enterprises.

Although VPLS is a widely deployed Layer 2 VPN technology, service provider networks migrate to EVPN because of the scaling benefits and ease of deployment. Some of the benefits of EVPN include:

- Control plane traffic is distributed with BGP and the broadcast and multicast traffic is sent using a shared multicast tree or with ingress replication.
- Control plane learning is used for MAC and IP addresses instead of data plane learning. MAC address learning requires the flooding of unknown unicast and ARP frames, whereas IP address learning does not require any flooding.
- Route reflector is used to reduce a full mesh of BGP sessions among PE devices to a single BGP session between a PE device and the route reflector.
- Autodiscovery with BGP is used to discover PE devices participating in a given VPN, PE devices participating in a given redundancy group, tunnel encapsulation types, multicast tunnel type, and multicast members.
- All-active multihoming is used. This allows a given CE device to have multiple links to multiple PE devices, and traffic traversing to-and-from that CE device fully utilizes all of these links (Ethernet segment).
- When a link between a CE device and a PE device fails, the PE devices for that EVPN instance (EVI) are notified of the failure with the withdrawal of a single EVPN route. This allows those PE devices to remove the withdrawing PE device as a next hop for every MAC address associated with the failed link (mass withdrawal).

FEC128 LDP-VPLS to EVPN Migration

Some service providers want to preserve their investments in VPLS. This leads to the need to connect the old VPLS networks to new networks that run EVPN. For this purpose, logical tunnel interfaces on the interconnection point of the VPLS and EVPN routing instances were used. However, all the other PE

devices belonged either to the VPLS network or to the EVPN network and were unaware of the other technology.

Starting in Junos OS Release 17.3, EVPN can be introduced into an existing VPLS network in a staged manner, with minimal impact to VPLS services. On a VPLS PE device, some customers can be moved to EVPN, while other customers continue to use VPLS pseudowires. Other PE devices can be entirely VPLS and switching customers on other PE devices to EVPN. This solution provides support for the seamless migration Internet draft (expires January ,2018), *(PBB-)EVPN Seamless Integration with (PBB-)VPLS*.

The seamless migration from FEC128 LDP-VPLS to EVPN solution supports the following functionality:

- Allow for staged migration toward EVPN on a site-by-site basis per VPN instance. For instance, new EVPN sites to be provisioned on EVPN PE devices.
- Allow for the coexistence of PE devices running both EVPN and VPLS for the same VPN instance and single-homed segments.

In the LDP-VPLS to EVPN migration, the PE device where some customers have been migrated to EVPN while other customers are being served using VPLS is called a super PE device. As super PE devices discover other super PE devices within a routing instance, they use EVPN forwarding to communicate with other super PE devices and VPLS pseudowires to PE devices running VPLS. The PE device with no EVPN awareness, and running only VPLS for all the customers, is called a VPLS PE device.

The CE device connected to a super PE can reach CE devices connected to EVPN-only PE devices or VPLS-only PE devices, but CE devices connected to EVPN-only PE devices cannot reach CE devices connected to VPLS-only PE devices.

Because the migration from LDP-VPLS to EVPN is supported on a per-routing instance basis, and if the routing instance is serving multiple customers on a PE device, all are migrated together. EVPN is responsible for setting up data forwarding between the PE devices upgraded to EVPN, while VPLS continues to set up data forwarding to PE devices that run VPLS. There should be zero impact for customers that still use VPLS pseudowire on all the PE devices.

NOTE: The following features are not supported with the LDP-VPLS to EVPN migration:

- Migration from FEC129 VPLS to EVPN.
- Migration from BGP-VPLS to EVPN.
- Migration of VPLS virtual switch to EVPN virtual switch.
- Migration of VPLS routing instance to EVPN virtual switch.
- Migration of VPLS routing instance or PBB-VPLS to PBB-EVPN.

- Seamless migration from EVPN back to VPLS.
- Enhancing EVPN to support the set of tools or statements and commands that VPLS supports.
- Active-active and active-standby multihoming. The migration to EVPN is supported only on single-homed deployments.
- Spanning all-active across EVPN and VPLS PE devices does not work, because the all-active multihoming feature is not supported on VPLS.
- Connecting EVPN-only PE devices with VPLS-only PE devices through super PE devices.
- IPv6, logical systems, multichassis support, and SNMP, because they are currently not supported on EVPN.

Sample Configuration for LDP-VPLS to EVPN Migration

The following sections provide the sample configuration required for performing the LDP-VPLS to EVPN migration.

LDP-VPLS Configuration

A typical static LDP-VPLS routing instance configuration is as follows:

```
user@host# show routing-instance foo
instance-type vpls;
vlan-id 100; (not needed for VLAN bundle service)
interface ge-2/0/0.590;
interface ae500.590;
routing-interface irb.0;
forwarding-options {
  family vpls {
    filter {
      input UNKNOWN-UNICAST;
    }
  }
}
protocols {
  vpls {
    control-word;
```

```

encapsulation-type ethernet-vlan;
enable-mac-move-action;
mac-table-size {
    100000;
    packet-action drop;
}
mac-table-aging-time ;
interface-mac-limit {
    100000;
    packet-action drop;
}
no-tunnel-services; (use label-switched interfaces)
vpls-id 245015;
mtu 1552;
ignore-mtu-mismatch;
mac-flush {
    any-spoke;
}
no-vlan-id-validate;
neighbor 192.168.252.64 {
    psn-tunnel-endpoint 10.0.0.31;
    pseudowire-status-tlv;
    revert-time 60;
    backup-neighbor 192.168.252.65 {
        psn-tunnel-endpoint 10.0.0.32;
        hot-standby;
    }
}
}
mesh-group Spoke { (access label-switched interface toward spoke)
local-switching;
neighbor 192.168.252.66 {
    psn-tunnel-endpoint 10.0.0.41;
    pseudowire-status-tlv;
}
neighbor 192.168.252.67 {
    psn-tunnel-endpoint 10.0.0.42;
    pseudowire-status-tlv;
}
}
}

```



```
connectivity-type permanent;
}
```

```
user@host# show interfaces ge-2/0/0.590
encapsulation vlan-vpls;
vlan-id 590;
output-vlan-map {
    swap;
    tag-protocol-id 0x8100;
    inner-vlan-id 590;
}
family vpls {
    filter {
        input-list [ listA ];
        output-list listB;
    }
}
```

EVPN Migration Configuration

To perform the FEC128 LDP-VPLS to EVPN migration, do the following:

1. On the backup Routing Engine, load Junos OS Release 17.3R1.
2. Perform in-service software upgrade (ISSU) to acquire primary role. Ensure that the VPLS unified ISSU does not have any impact on the VPLS forwarding.
3. Identify routing instances (customers) that need to be migrated to EVPN.
4. Enable EVPN in a single routing instance.
 - Change routing instance type to `evpn`, and include the `evpn` statement at the `[edit routing-instances routing-instance-name protocols]` hierarchy level, and also include the `vpls` statement at the same hierarchy to support VPLS commands.

For example:

```
[edit routing-instances routing-instance-name]
instance-type evpn;
interface ge-2/0/0.590;
interface ae500.590;
routing-interface irb.0;
```

```

route-distinguisher 1.1.1.1:50; (add for LDP-VPLS)
vrf-target target:100:100; (add for LDP-VPLS)
forwarding-options {
    family vpls {
        filter {
            input UNKNOWN-UNICAST;
        }
    }
}
protocols {
    vpls { (supports all existing VPLS commands)
}

```

5. Enable family EVPN signaling in BGP.

For example:

```

protocols {
    bgp {
        local-as 102;

        group 2mx {
            type internal;
            local-address 81.1.1.1;

            family evpn {
                signaling;
            }
            neighbor 81.2.2.2;
            neighbor 81.9.9.9;
        }
    }
}

```

After the configuration for the EVPN migration is committed, the routing protocol process and the Layer 2 address learning process start building the EVPN state to reflect interfaces, bridge domains, peers and routes. The locally learnt MAC addresses are synchronized by the Layer 2 address learning process in the instance.vpls.0 to the routing protocol process. When a local MAC ages out in the instance.vpls.0, the routing protocol process is informed by the Layer 2 address learning process.

When an EVPN peer is learnt, the routing protocol process sends a new message to the Layer 2 address learning process to remove the peer's label-switched interface or virtual tunnel logical interface from the VE mesh group and disables MAC-learning on it. The EVPN IM next-hop is then added to the VE mesh

group. The EVPN behavior in the routing protocol process of learning MAC addresses over BGP and informing Layer 2 address learning process of the MPLS next hop is maintained.

The VPLS statements and commands continue to apply to the VPLS pseudowires between the PE devices and the MAC addresses learnt over them. The EVPN statements and commands apply to PE devices running EVPN.

Reverting to VPLS

If the EVPN migration runs into issues, you can revert back to VPLS until the issue is understood. The routing instance is reverted from a super PE to a VPLS PE in a non-catastrophic manner by enabling the following configuration:

```
[edit routing-instances routing-instance-name]
user@host# set instance-type vpls
user@host# delete protocols evpn
user@host# delete route-distinguisher (if running LDP-VPLS)
user@host# delete vrf-target (if running LDP-VPLS)
```

On reverting the EVPN migration to VPLS, the following happens:

1. The EVPN state information is deleted.
2. There is a trigger for withdrawal of EVPN control plane routes.
3. The routing protocol process sends a new message to the Layer 2 address learning process with the label-switched interface or the virtual tunnel logical interface for the routing instance and peer.
4. The label-switched or virtual tunnel interface adds the new message to the flood group and MAC learning is enabled.
5. The egress IM next hop is deleted by the routing protocols process, prompting the Layer 2 address learning process to remove it from the flood group.
6. Remote MAC addresses are learned again over the label-switched interface or virtual tunnel logical interface.

LDP-VPLS to EVPN Migration and Other Features

[Table 40 on page 889](#) describes the functionality of some of the related features, such as multihoming and integrated routing and bridging (IRB) with the LDP-VPLS to EVPN migration.

Table 40: EVPN Migration and Other Features Support

Feature	Supported Functionality in EVPN Migration
MAC move	<p>MAC moves are supported between VPLS-only PE device and super PE devices.</p> <p>When a MAC address moves from a VPLS-only PE device to a super PE device, it is learned over BGP, and the routing protocol process informs the Layer 2 address learning process of the EVPN next hop to be updated in the foo.vpls.0 routing table.</p> <p>When a MAC address moves from a super PE device to a VPLS-only PE device, it is learned in the Packet Forwarding Engine on the label-switched interface or virtual tunnel interface. The Layer 3 address learning process updates it to VPLS or the label-switched interface next hop.</p> <p>When the type 2 route is withdrawn by EVPN BGP, the MAC address is not deleted from the forwarding table, so there is no loss of data.</p> <p>The forwarding MAC table is shared by VPLS and EVPN. Some attributes, such as <code>mac-table-size</code> and <code>mac-table-aging-time</code> could be configured under both EVPN and VPLS. When there is a conflict, the values under EVPN take precedence.</p>
IRB	<p>No changes needed in IRB.</p> <p>On a super PE device, EVPN populates the /32 host routes learned over MAC+IP type 2 routes from EVPN peers in a Layer 3 virtual routing and forwarding, while VPLS IRB forwarding using subnet routes works on sites still running VPLS.</p>

Table 40: EVPN Migration and Other Features Support *(Continued)*

Feature	Supported Functionality in EVPN Migration
Hierarchical VPLS	<p>In an H-VPLS network with hub-and-spoke PE devices, when the hub PE device is migrated to EVPN, local MAC addresses learned over the access label-switched or virtual tunnel interface need to be advertised to BGP, so that the other EVPN-only PE devices or super PE devices can reach them.</p> <p>Take the following into consideration when migrating an H-VPLS network to EVPN:</p> <ul style="list-style-type: none"> • Hubs typically have local switching enabled as interspoke traffic is forwarded through the hub. If spokes alone are migrated to EVPN and spokes have Layer 3 or MPLS reachability to each other, the label-switched or virtual tunnel interface to the hub and EVPN next hop (remote spoke) is present in the VPLS edge (VE) floodgroup. This results in two copies of broadcast, unknown unicast, and multicast (BUM) traffic received by the remote spoke. One option to avoid this behavior is to migrate the hubs to EVPN too. • EVPN is not aware of hierarchy. All peers are considered core-facing. Once hubs and spokes are migrated to EVPN, split horizon prevents the BUM traffic from being forwarded to other core-facing PE devices.
ESI configuration	Ethernet segment identifier (ESI) is configured at the physical interface or port level.

RELATED DOCUMENTATION

[EVPN Overview](#) | 874

Convergence in an EVPN MPLS Network

IN THIS CHAPTER

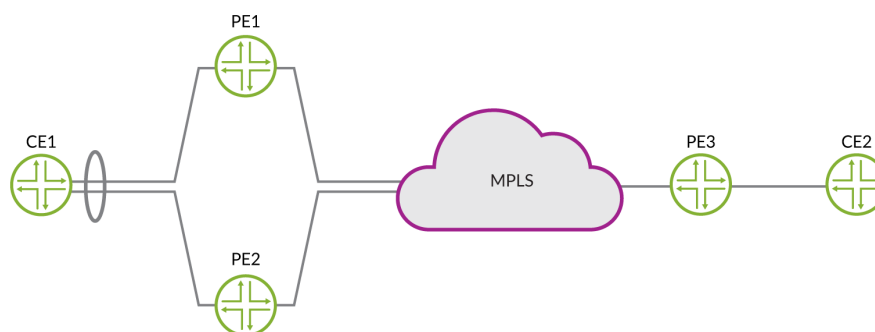
- [Convergence in a Multihomed EVPN MPLS Network | 891](#)

Convergence in a Multihomed EVPN MPLS Network

Multihoming a CE device to two or more PE devices on an EVPN MPLS network provides redundant connectivity and allows the network to continue providing services when one of the PE devices fail or if there is a PE-CE link failure. When the multihomed connection from a CE device to a PE device fails, the devices in the networks updates the EVPN routes, including any MAC routes and next hop entries in their forwarding table

[Figure 89 on page 891](#) shows a simple EVPN topology with CE1 multihomed to PE1 and PE2 with an Ethernet segment. If the connection from CE1 to PE1 fails and similarly when the connection from CE1 to PE1 is restored, PE1 will withdraw MAC routes from PE2 and PE3, while PE2 and PE3 will remove the MAC routes that they learned from PE1. EVPN egress link protection adds backup next hops on the multihomed PE devices and is used to support fast reroute (FRR) in the network. When you enable EVPN egress link protection, you can improve the convergence time in the EVPN MPLS network and reduce traffic loss when the link to a PE device fails.

Figure 89: Simple EVPN MPLS Topology



g301503

To enable EVPN egress link protection, include `evpn-egress-link-protection` at the `[edit routing-options forwarding-table]` hierarchy level on all multihomed PE devices in the EVPN network.

NOTE: In addition, you must also include `dynamic-list-next-hop` at the `[edit routing-options forwarding-table]` hierarchy level on the single-homed PE device (PE3).

The following example shows a sample configuration for evpn egress link protection.

```
routing-instances {
  blue {
    instance-type evpn;
    vlan-id 100;
    interface ae0.0;
    route-distinguisher 10.255.255.1:100;
    vrf-target target:100:100;
    protocols evpn;
  }
}
routing-options {
  forwarding-table {
    export lb;
    evpn-egress-link-protection;
  }
}
policy-options {
  policy-statement lb {
    term 1 {
      then {
        load-balance per-packet;
      }
    }
  }
}
```

RELATED DOCUMENTATION

[Configuring Dynamic List Next Hop | 332](#)

forwarding-table

Pseudowire Termination at an EVPN

IN THIS CHAPTER

- [Overview of Pseudowire Termination at an EVPN | 893](#)
- [Configuring Pseudowire Termination | 894](#)
- [Support for Redundant Logical Tunnel | 898](#)

Overview of Pseudowire Termination at an EVPN

IN THIS SECTION

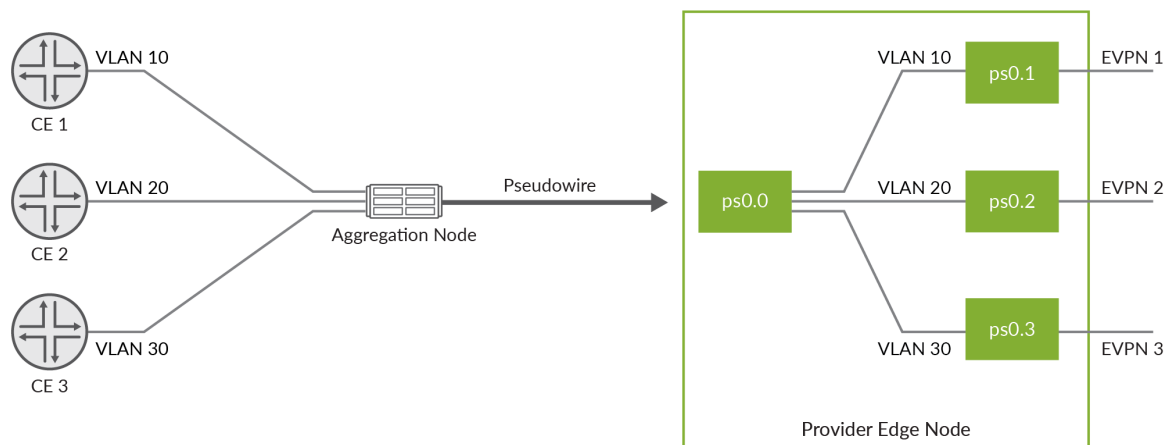
- [Benefits of Pseudowire Termination at an EVPN | 894](#)
- [Support for Junos Node Slicing | 894](#)

Pseudowires provide point-to-point service over an MPLS or Layer 2 circuit. They enable providers to extend their Layer 2 networks. A pseudowire tunnel terminates on a logical interface on the PE router with the transport logical interface configured for interoperability with the access or aggregation device on the other end of the pseudowire tunnel. It is identified as psn.0 and supports both port-based and VLAN-based encapsulation over pseudowire. The corresponding service logical interfaces are identified as psn.1 to psn.n and are configured to support EVPN in the EVPN routing instance.

[Figure 90 on page 894](#) shows a port-based pseudowire tunnel that originates on an aggregation node carrying VLAN traffic from several CE devices. The pseudowire terminates in a transport logical interface

(ps0.0) on a single-homed PE router, where the VLAN traffic is demultiplexed into different service logical interfaces(ps0.1, ps0.2, and ps0.3) along with their corresponding EVPN routing instances.

Figure 90: Pseudowire Terminating into Single-Homed PE Device



Benefits of Pseudowire Termination at an EVPN

- **Resiliency**—When a redundant logical tunnel is configured on a separate FPC, connectivity for the pseudowire automatically switches to another FPC.
- **Reduced cost**—Rather than using additional devices to terminate a pseudowire tunnels, you can configure pseudowire termination on the same device that also supports EVPN.

Support for Junos Node Slicing

Pseudowire termination is supported on multiple partitions in a single MX Series. Using Junos Node Slicing, you can create multiple partitions on a single MX router. These partitions are referred to as a guest network functions (GNFs). For more information on Junos Node slicing, see [Junos Node Slicing User Guide](#).

Configuring Pseudowire Termination

To configure a pseudowire logical interface on the PE device, perform these tasks:

- Configure a logical interface.

- Configure the transport logical interface
- Configure the service logical interface.
- Configure the EVPN routing instance to support the incoming VLAN services.

To configure the logical interface:

1. Specify that you want to configure the pseudowire logical interface device.

```
user@host# edit interfaces ps0
```

2. Specify the logical tunnel interface that is the anchor point for the pseudowire logical interface device. In this case, the logical tunnel interface and anchor point is in the format *lt-fpc/pic/port*.

NOTE: Tunnel services must be enabled in order to create the *lt* interface that is the anchor point or a member link in a redundant logical tunnel. You use the command, *set chassis fpc slot-number pic pic-number tunnel-services bandwidth bandwidth* to enable tunnel services.

```
[edit interfaces ps0]
user@host# set anchor-point lt-1/0/10
```

3. (Optional) Specify the MAC address for the pseudowire logical interface device.

NOTE: You should ensure that you change the MAC address prior to passing traffic or binding subscribers on the pseudowire port. Changing the MAC address when the pseudowire port is active (for example, while an upper layer protocol is negotiating) can negatively impact network performance until adjacencies learn of the new MAC address.

```
[edit interfaces ps0]
user@host# set mac 00:00:5E:00:53:55
```

4. (Optional) Specify the VLAN tagging method used for the pseudowire logical interface device. You can specify single tagging, dual (stacked) tagging, mixed (flexible) tagging, or no tagging.

```
[edit interfaces ps0]
user@host# set flexible-vlan-tagging
```

See [Enabling VLAN Tagging](#) for additional information about VLAN tagging.

To configure the logical interface:

1. Specify that you want to configure the pseudowire logical interface.

```
user@host# edit interfaces ps0
```

2. Specify that you want to configure unit 0 , which represents the transport logical interface.

```
[edit interfaces ps0]  
user@host# edit unit 0
```

3. Specify either the encapsulation method for the transport logical interface. You can specify either ethernet-ccc (port-based) or vlan-ccc (vlan-based) encapsulation.

```
[edit interfaces ps0 unit 0]  
user@host# set encapsulation ethernet-ccc
```

To configure the service logical interface:

1. Configure the unit for the service logical interface. Use a non-zero unit number.

```
user@host# edit interfaces ps0 unit1
```

2. Configure the encapsulation on the service interface.

```
[edit interfaces ps0 unit 1]  
user@host# set encapsulation vlan-bridge;
```

3. (Optional) Configure the VLAN IDs.

```
[edit interfaces ps0 unit 1]  
user@host# set vlan-id vlan-id;
```

4. (Optional) Configure the VLAN tag IDs to support dual-tagged VLANs.

```
[edit interfaces ps0 unit 1]
set vlan-tags inner vlan-id outer vlan-id;
```

The following is a sample configuration for a logical interface with services supporting single-tag and dual-tag VLANs and the L2 circuit associated with the interface:

```
ps0 {
  anchor-point {
    lt-0/0/0;
  }
  flexible-vlan-tagging;
  mac 00:00:5E:00:53:00;
  unit 0 {
    encapsulation ethernet-ccc;
  }
  unit 1 {
    encapsulation vlan-bridge;
    vlan-id 600;
  }
  unit 2 {
    encapsulation vlan-bridge;
    vlan-tag outer 200 inner 101;
  }
}
protocols {
  l2circuit {
    neighbor 192.168.100.1 {
      interface ps0.0 {
        virtual-circuit-id 700;
        encapsulation-type ethernet-vlan;
        ignore-mtu-mismatch;
        pseudowire-status-tlv;
      }
    }
  }
}
```

The following is a sample configuration for an EVPN routing instance that corresponds to a service logical interface:

```
routing-instances evpn-1 {
    instance-type evpn;
    vlan-id 600;
    interface ps0.1;
    route-distinguisher 3;3;
    vrf-target target:1:1;
    protocols {
        evpn;
    }
}
```

For more information on configuring logical interface termination, see [Pseudowire Subscriber Logical Interfaces Overview](#).

For more information on configuring EVPN routing instances, see ["Configuring EVPN Routing Instances" on page 10](#).

Support for Redundant Logical Tunnel

You can configure two logical tunnels on two different line cards to create a redundant logical tunnel (RLT). This provides redundancy when there is a failure in the FPC. Pseudowire over RLT supports two members in active-standby mode. Only one member link is active and carrying traffic at any given time.

To create the RLT, configure a pair of `redundant-logical-tunnel` interface at the `[edit chassis redundancy-group interface-type]` hierarchy and include the logical tunnel interface in the redundancy group by configuring `member-interface interface-name` at the `[edit interfaces interface-name redundancy-group]` hierarchy level.

The following is a sample configuration for pseudowire RLT.

```
chassis {
    pseudowire-service {
        device-count 500;
    }
    redundancy-group {
        interface-type {
            redundant-logical-tunnel {
                device-count 10;
            }
        }
    }
}
```

```
    }  
  }  
}  
interfaces {  
  rlt0 {  
    redundancy-group {  
      member-interface lt-0/1/0;  
      member-interface lt-0/1/1;  
    }  
  }  
  ps0 {  
    anchor-point {  
      rlt0;  
    }  
  }  
}
```

For more information on redundant logical tunnels, see [Redundant Logical Tunnels Overview](#)

Configuring the Distribution of Routes

IN THIS CHAPTER

- [Configuring an IGP on the PE and P Routers on EX9200 Switches | 900](#)
- [Configuring IBGP Sessions Between PE Routers in VPNs on EX9200 Switches | 901](#)
- [Configuring a Signaling Protocol and LSPs for VPNs on EX9200 Switches | 902](#)
- [Configuring Entropy Labels | 905](#)
- [Configuring Control Word for EVPN-MPLS | 906](#)
- [Understanding P2MPs LSP for the EVPN Inclusive Provider Tunnel | 908](#)
- [Configuring Bud Node Support | 910](#)

Configuring an IGP on the PE and P Routers on EX9200 Switches

For Layer 2 VPNs, Layer 3 VPNs, virtual-router routing instances, VPLS, EVPNs, and Layer 2 circuits to function properly, the service provider's PE and P routers must be able to exchange routing information. For this to happen, you must configure either an IGP (such as OSPF or IS-IS) or static routes on these routers. You configure the IGP on the master instance of the routing protocol process at the [edit protocols] hierarchy level, not within the routing instance used for the VPN—that is, not at the [edit routing-instances] hierarchy level.

When you configure the PE router, do not configure any summarization of the PE router's loopback addresses at the area boundary. Each PE router's loopback address should appear as a separate route.

RELATED DOCUMENTATION

[Understanding IS-IS Configuration](#)

[Example: Configuring IS-IS](#)

[OSPF User Guide](#)

Configuring IBGP Sessions Between PE Routers in VPNs on EX9200 Switches

You must configure an IBGP session between the PE routers to allow the PE routers to exchange information about routes originating and terminating in the VPN. The PE routers rely on this information to determine which labels to use for traffic destined for remote sites.

Configure an IBGP session for the VPN as follows:

```
[edit protocols]
bgp {
  group group-name {
    type internal;
    local-address ip-address;
    family evpn {
      signaling;
    }
    family (inet-vpn | inet6-vpn) {
      unicast;
    }
    family l2vpn {
      signaling;
    }
    neighbor ip-address;
  }
}
```

The IP address in the `local-address` statement is the address of the loopback interface on the local PE router. The IBGP session for the VPN runs through the loopback address. (You must also configure the loopback interface at the `[edit interfaces]` hierarchy level.)

The IP address in the `neighbor` statement is the loopback address of the neighboring PE router.

The `family` statement allows you to configure the IBGP session for Layer 2 VPNs, VPLS, EVPNs or for Layer 3 VPNs.

- To configure an IBGP session for Layer 2 VPNs and VPLS, include the `signaling` statement at the `[edit protocols bgp group group-name family l2vpn]` hierarchy level:

```
[edit protocols bgp group group-name family l2vpn]
signaling;
```


- To configure an IBGP session for EVPNs, include the signaling statement at the [edit protocols bgp group *group-name* family evpn] hierarchy level:

```
[edit protocols bgp group group-name family evpn]
signaling;
```

- To configure an IPv4 IBGP session for Layer 3 VPNs, configure the unicast statement at the [edit protocols bgp group *group-name* family inet-vpn] hierarchy level:

```
[edit protocols bgp group group-name family inet-vpn]
unicast;
```

- To configure an IPv6 IBGP session for Layer 3 VPNs, configure the unicast statement at the [edit protocols bgp group *group-name* family inet6-vpn] hierarchy level:

```
[edit protocols bgp group group-name family inet6-vpn]
unicast;
```

NOTE: You can configure both family inet and family inet-vpn or both family inet6 and family inet6-vpn within the same peer group. This allows you to enable support for both IPv4 and IPv4 VPN routes or both IPv6 and IPv6 VPN routes within the same peer group.

RELATED DOCUMENTATION

[Configuring a Signaling Protocol and LSPs for VPNs on EX9200 Switches](#) | 902

Configuring a Signaling Protocol and LSPs for VPNs on EX9200 Switches

IN THIS SECTION

- [Using LDP for VPN Signaling](#) | 903

For VPNs to function, you must enable the LDP signaling protocol on the provider edge (PE) routers and on the provider (P) routers.

To enable the LDP signaling protocol, perform the steps in the following section:

Using LDP for VPN Signaling

To use LDP for VPN signaling, perform the following steps on the PE and provider (P) routers:

1. Configure LDP on the interfaces in the core of the network by including the `ldp` statement at the `[edit protocols]` hierarchy level.

You need to configure LDP only on the interfaces between PE routers or between PE and P routers. You can think of these as the “core-facing” interfaces. You do not need to configure LDP on the interface between the PE and customer edge (CE) routers.

```
[edit]
protocols {
  ldp {
    interface type-fpc/pic/port;
  }
}
```

2. Configure the MPLS address family on the interfaces on which you enabled LDP (the interfaces you configured in Step ["1" on page 903](#)) by including the `family mpls` statement at the `[edit interfaces type-fpc/pic/port unit logical-unit-number]` hierarchy level.

```
[edit]
interfaces {
  type-fpc/pic/port {
    unit logical-unit-number {
      family mpls;
    }
  }
}
```

3. Configure OSPF or IS-IS on each PE and P router.

You configure these protocols at the master instance of the routing protocol, not within the routing instance used for the VPN.

- To configure OSPF, include the `ospf` statement at the [edit protocols] hierarchy level. At a minimum, you must configure a backbone area on at least one of the router's interfaces.

```
[edit]
protocols {
  ospf {
    area 0.0.0.0 {
      interface type-fpc/pic/port;
    }
  }
}
```

- To configure IS-IS, include the `isis` statement at the [edit protocols] hierarchy level and configure the loopback interface and International Organization for Standardization (ISO) family at the [edit interfaces] hierarchy level. At a minimum, you must enable IS-IS on the router, configure a network entity title (NET) on one of the router's interfaces (preferably the loopback interface, lo0), and configure the ISO family on all interfaces on which you want IS-IS to run. When you enable IS-IS, Level 1 and Level 2 are enabled by default. The following is the minimum IS-IS configuration. In the address statement, *address* is the NET.

```
[edit]
interfaces {
  lo0 {
    unit logical-unit-number {
      family iso {
        address address;
      }
    }
  }
  type-fpc/pic/port {
    unit logical-unit-number {
      family iso;
    }
  }
}
protocols {
  isis {
    interface all;
  }
}
```

For more information about configuring OSPF and IS-IS, see the [OSPF User Guide](#) and [IS-IS User Guide](#).

RELATED DOCUMENTATION

| [EVPN Overview for Switches](#) | 878

Configuring Entropy Labels

An entropy label is a special label that enhances a network device's ability to load-balance traffic across equal-cost multipath (ECMP) paths or link aggregation groups (LAGs). When you configure an entropy label at the ingress device, the transit device use the label to determine a path in place of performing a deep packet inspection (DPI). If the transit device does not support LDP Entropy Label Capability (ELC), the transit device ignores the entropy label and continues to propagate the packet using traditional DPI for load balancing. The entropy label is popped at the penultimate hop device.

An entropy label can be any label value between 16 to 1048575 (regular 20-bit label range). Since this range overlaps with the existing regular label range, a special label called entropy label indicator (ELI) is also inserted in the label stack before the entropy label. IANA assigned ELI label a value of 7. This distinguish entropy labels from other service labels.

To configure entropy labels, include the **entropy label** at the **[edit protocols ldp]** hierarchy on the ingress device and include the configured policy in the routing policy.

```
protocols {
  ldp {
    entropy-label {
      ingress-policy policy-name;
    }
  }
}
```

```
policy-options {
  policy-statement policy-name {
    term Add-Entropy-label {
      from {
        route-filter route exact;
      }
    }
  }
}
```

```

        then accept;
    }
}
}

```

To enable a penultimate hop device to pop the entropy label, include the **load-balance-label-capability** statement at the **[edit forwarding-option]** hierarchy.

```

forwarding-options {
    load-balance-label-capability;
}

```

RELATED DOCUMENTATION

[Configuring the Entropy Label for LSPs](#)

[entropy-label](#)

Configuring Control Word for EVPN-MPLS

SUMMARY

Transit devices in an EVPN-MPLS network are not aware of the type of payload it carries. While parsing an MPLS encapsulated packet for hashing, a transit device can incorrectly calculate an Ethernet payload as an IPv4 or IPv6 payload if the first nibble of the destination address MAC is 0x4 or 0x6, respectively, causing out-of-order packet delivery. To identify the payload as an Ethernet payload, we can insert a control word with a value of 0 in the first four bits between the label stack and the L2 header of the packet. This ensures that the packet is not identified as an IPv4 or IPv6 packet.

NOTE: You do not need to enable control word on the devices in your transit network when the transit devices consist of Juniper Networks EX 9200 switches, MX series routers, or PTX Series routers. These Juniper devices correctly identify the Ethernet payload as an IPv4/IPv6 payloads,

even when the Ethernet destination MAC address starts with 0x4 or 0x6 nibble. The Juniper devices perform hashing based on the IP header fields inside the Ethernet frame and will not send out-of-order packets. In this case, we recommend not using control word as there are no benefits..

To enable control word, set `control-word` for the `evpn` protocol for a specified routing instance. The following output shows a sample multihomed routing instance with control word configured

```
user@router1# show routing-instances
routing-instances EVPN-green
vlan-id 200;
interface ae0.1;
route-distinguisher 10.255.255.1:200;
vrf-target target:100:200;
protocols {
    evpn {
        control-word;
    }
}
```

To view routes where control word is supported, use the `show route table mpls.0 protocol evpn` operational command. Egress routes display an offset of 252.

```
show route table mpls.0 protocol evpn
303744          *[EVPN/7] 00:00:13, remote-pe 10.255.255.2, routing-instance blue, route-type
Egress-MAC
312             > to 5.0.0.1 via ge-0/0/2.0, Push 299984 Offset: 252
313    303760     *[EVPN/7] 00:00:13, remote-pe 10.255.255.2, routing-instance blue,
route-type Egress-MAC
314             > to 5.0.0.1 via ge-0/0/2.0, Push 299888 Offset: 252
315    303776     *[EVPN/7] 00:00:13, remote-pe 10.255.255.2, routing-instance blue,
route-type Egress-MAC
316             > to 5.0.0.1 via ge-0/0/2.0, Push 300032 Offset: 252
317    303792     *[EVPN/7] 00:00:13, remote-pe 10.255.255.2, routing-instance blue,
route-type Egress-IM, vlan-id 4
318             > to 5.0.0.1 via ge-0/0/2.0, Push 302000 Offset: 252
319    303808     *[EVPN/7] 00:00:13, remote-pe 10.255.255.2, routing-instance blue,
route-type Egress-IM, vlan-id 5
320             > to 5.0.0.1 via ge-0/0/2.0, Push 302016 Offset: 252
321    303824     *[EVPN/7] 00:00:13, remote-pe 10.255.255.2, routing-instance blue,
```

```
route-type Egress-IM, vlan-id 6
322 > to 5.0.0.1 via ge-0/0/2.0, Push 302032 Offset: 252
```

Understanding P2MPs LSP for the EVPN Inclusive Provider Tunnel

IN THIS SECTION

- [NSR and Unified ISSU Support on EVPN with P2MP | 909](#)
- [Benefits of EVPN P2MPs LSP for the EVPN Inclusive Provider Tunnel | 910](#)

An EVPN instance comprises Customer Edge devices (CEs) that are connected to Provider Edge devices (PEs) to form the edge of the MPLS infrastructure. A CE may be a host, a router, or a switch. The PEs provide virtual Layer 2 bridged connectivity between the CEs. There may be multiple EVPN instances in the provider's network. The PEs may be connected by an MPLS Label Switched Path (LSP) infrastructure, which provides the benefits of MPLS technology, such as fast reroute, resiliency, etc.

The PEs may also be connected by an IP infrastructure, where IP/GRE (Generic Routing Encapsulation) tunneling or other IP tunneling can be used between the PEs. Here we understand the MPLS LSPs as the tunneling technology designed to be extensible to IP tunneling as the Packet Switched Network (PSN) tunneling technology.

Starting in Junos OS Release 18.2R1 onwards, Junos OS provides the ability to configure and signal a P2MP LSP for the EVPN Inclusive Provider Tunnel for BUM traffic. P2MP LSPs can provide efficient core bandwidth utilization by using the multicast replication only at the required nodes instead of ingress replication at the ingress PE.

You can configure and signal a P2MP LSP for the EVPN Inclusive Provider Tunnel in the PMSI Attributes of the Inclusive Multicast Ethernet Tag Route. The P2MP tunnel is used for forwarding Broadcast, unknown Unicast, and Multicast (BUM) packets at the ingress PE. When representing the PMSI attributes in the Inclusive Multicast Ethernet Tag Route, a transport label is not represented for P2MP Provider Tunnels, except in the case where Aggregation is used. The transport label is used to identify the EVI at the egress PE.

When the P2MP Provider Tunnels are used, the ESI labels for Split Horizon are assigned upstream instead of downstream. The label that the egress PE receives as a Split Horizon label is allocated by the ingress PE. Since each upstream PE cannot allocate the same label, the label does not identify the ESI and must be referred in the context label space of the ingress PE. The transport label uniquely identifies the ingress PE and the split horizon label is identified by transport-label or SH-label tuple.

At the ingress PE, an upstream assigned ESI label is allocated and signaled for Split Horizon function. This label is added to the BUM traffic that originates from the given Ethernet Segment. By default, an egress may receive traffic with an ESI label that is not configured on this PE because of P2MP tunnels and upstream assigned labels.

NOTE: For IR tunnels, ingress includes the ESI label to only those PEs with the ES. In such case, the egress PE pops the ESI label and floods the packet normally.

For E-tree, the leaf label is assigned upstream and its function is similar to the SH label. The ingress PE allocates the leaf label and represents it in the E-tree extended community for the Ethernet A-D per ES route. When the Leaf PE acts as an ingress, it adds the leaf label for BUM traffic. An egress PE acting as a leaf discards the traffic which contains the leaf label. Egress PEs acting as a root pops the leaf label and forwards the traffic. Since leaf labels are upstream assigned, they may not be unique because multiple Leaf PEs may have allocated the same leaf label. Therefore, the leaf label must be identified by the (transport-label, Leaf-label) tuple.

NOTE: The EVPN P2MP functions without a `lsi` or `vt` interface. Instead, the EVPN allocates a label for use as the mLDP/RSVP transport label at the egress PE.

NSR and Unified ISSU Support on EVPN with P2MP

Nonstop active routing (NSR) and graceful Routing Engine switchover (GRES) minimize traffic loss when there is a Routing Engine switchover. When a Routing Engine fails, NSR and GRES enable a routing platform with redundant Routing Engines to switch over from a primary Routing Engine to a backup Routing Engine and continue forwarding packets.

Unified in-service software upgrade (ISSU) allows you to upgrade your Junos OS software on your MX Series router with no disruption on the control plane and with minimal disruption of traffic. Both GRES and NSR must be enabled to use unified ISSU. Nonstop active routing (NSR) and graceful Routing Engine switchover (GRES) minimize traffic loss when there is a Routing Engine switchover.

When a Routing Engine fails, NSR and GRES enable a routing platform with redundant Routing Engines to switch over from a primary Routing Engine to a backup Routing Engine and continue forwarding packets. Unified in-service software upgrade (ISSU) allows you to upgrade your Junos OS software on your MX Series router with no disruption on the control plane and with minimal disruption of traffic. Both GRES and NSR must be enabled to use unified ISSU.

Junos OS mirrors essential data when NSR is enabled. For EVPN with P2MP mLDP replication, the LDP/RSVP transport label will be mirrored on the standby Routing Engine. For information on other mirrored data and NSR data flow, see ["NSR and Unified ISSU Support for EVPN " on page 383](#).

Benefits of EVPN P2MPs LSP for the EVPN Inclusive Provider Tunnel

- Provides efficient core bandwidth utilization by using multicast replication only at the required nodes.
- Manages the ingress replication at the ingress PE to avoid an over-load.
- Supports P2MP LSPs for EVPN Inclusive P-Multicast Trees for EVPNoMPLS for both mLDP and RSVP-TE P2MP-E-tree.

RELATED DOCUMENTATION

[Example: Configuring Point-to-Multipoint LDP LSPs as the Data Plane for Intra-AS MBGP MVPNs](#)

Example: Configuring Ingress Replication for IP Multicast Using MBGP MVPNs

Flooding Unknown Traffic Using Point-to-Multipoint LSPs in VPLS

Configuring Point-to-Multipoint LSPs for an MBGP MVPN

Configuring Dynamic Point-to-Multipoint Flooding LSPs

[Configuring RSVP Automatic Mesh](#)

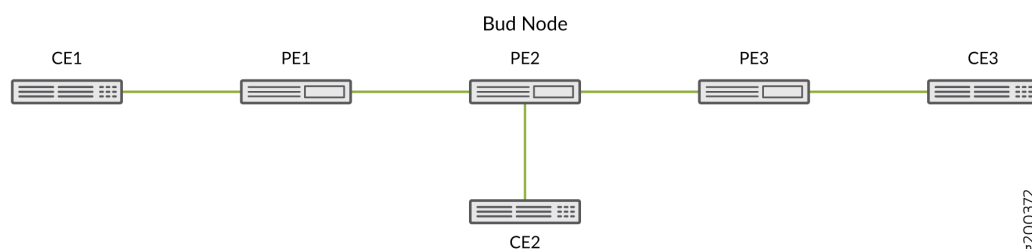
[container-label-switched-path](#)

Configuring Bud Node Support

Along a LSP, PE devices function as an ingress device, transit device or egress device. A bud node is a PE device that functions as both an egress and transit device. In a P2MP LSP, you can have devices that will function in the role of a both as a transit device and an egress device.

Figure 91 on page 911 illustrates a simple EVPN network with a bud node at PE2. When CE1 sends a multicast packet out, PE2 operates as a transit device and forwards the packet to PE3. It also functions as a egress device and pops the MPLS label and replicates multicast packets destined for CE2.

Figure 91: Bud Node in an EVPN Network



To enable a PE device to function as a bud-node, include `p2mp-bud-support` statement at the `[edit routing-instances routing-instance-name protocols evpn]` hierarchy. When `p2mp-bud-support` is enabled or disabled, you may observe dropped packets on the device. This occurs because changing bud node support affects the forwarding state in the routing instance, which results in the forwarding table being rebuilt.

NOTE: We recommend that all PE devices in the LSP that may potentially function as both a egress and transit device be enabled as a bud node.

Configuring VLAN Services and Virtual Switch Support

IN THIS CHAPTER

- [Overview of VLAN Services for EVPN | 912](#)
- [VLAN-Based Service for EVPN | 915](#)
- [VLAN Bundle Service for EVPN | 918](#)
- [Configuring EVPN VLAN Bundle Services | 920](#)
- [Virtual Switch Support for EVPN Overview | 923](#)
- [Configuring EVPN with Support for Virtual Switch | 925](#)
- [Example: Configuring EVPN with Support for Virtual Switch | 929](#)

Overview of VLAN Services for EVPN

A Data Center Service Provider (DCSP) hosts the data center for its multiple customers on a common physical network. To each customer (also called a tenant), the service looks like a full-fledged data center that can expand to 4094 VLANs and all private subnets. For disaster recovery, high availability, and optimization of resource utilization, it is common for the DCSP to span across the data center to more than one site. When deploying the data center services, a DCSP faces the following main challenges:

- Extending Layer 2 domains across more than one data center site. This requires optimal intra-subnet traffic forwarding.
- Supporting optimal inter-subnet traffic forwarding and optimal routing in the event of virtual machine (VM).
- Supporting multiple tenants with independent VLAN and subnet space.

The DCSP might require Ethernet VLAN services to be extended over a WAN with a single EVPN instance (EVI). Junos OS supports VLAN-based service, VLAN bundle service, and VLAN-aware bundle service, while maintaining data and control plane separation. The following figures illustrate the relationship between VLANs and EVIs.

- [Figure 92 on page 914](#) illustrates a VLAN-based service where you have a one-to-one mapping between a VLAN ID and a EVI.
- [Figure 93 on page 914](#) illustrates a VLAN bundles service where you can have one EVI mapped to many VLAN IDs in a single bridge domain. The bridge table is shared among the VLANs.

- [Figure 94 on page 914](#) illustrates the relationship between VLANs and EVIS in a VLAN-aware bundle service where you can have one EVI mapped to many VLAN IDs. Each VLAN has a different bridge table.

Figure 92: VLAN-Based Service

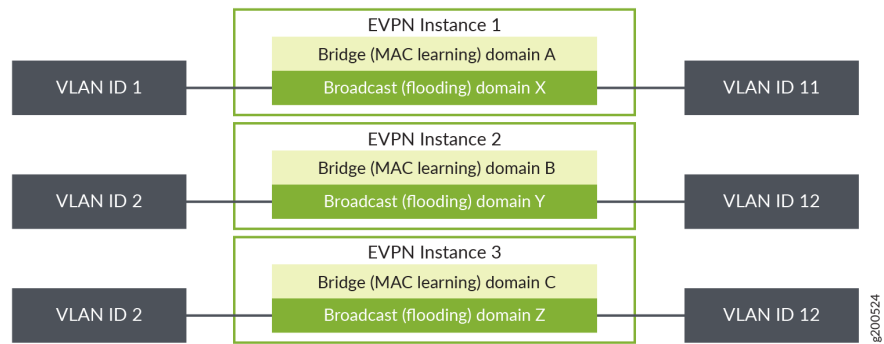


Figure 93: VLAN Bundle Service

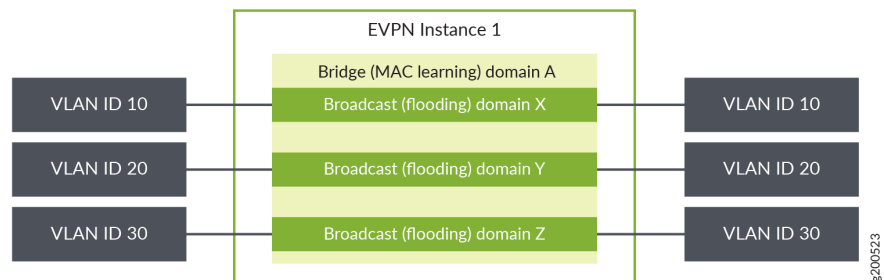
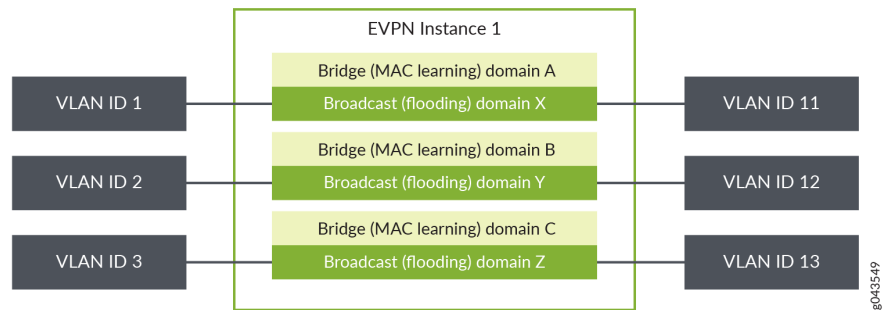


Figure 94: VLAN-Aware Bundle Service



RELATED DOCUMENTATION

[VLAN-Based Service for EVPN | 915](#)

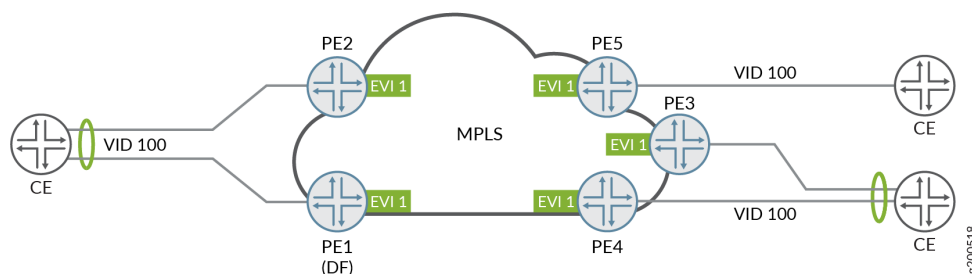
[VLAN Bundle Service for EVPN | 918](#)

[Virtual Switch Support for EVPN Overview | 504](#)

VLAN-Based Service for EVPN

VLAN-based service allows a one-to-one mapping of a single broadcast domain to a single bridge domain. Each VLAN is mapped to a single EVPN instance (EVI), resulting in a separate bridge table for each VLAN. Prior to Junos OS Release 17.3R1, VLAN translation was not supported. Without VLAN translation, the customer edge VLAN must use the same VLAN ID (VID). You can still send an MPLS encapsulated frames with the originating VID. [Figure 95 on page 915](#) illustrates a topology where all the CE devices use the same CE VID for a single VLAN-based EVI. VID translation is not needed.

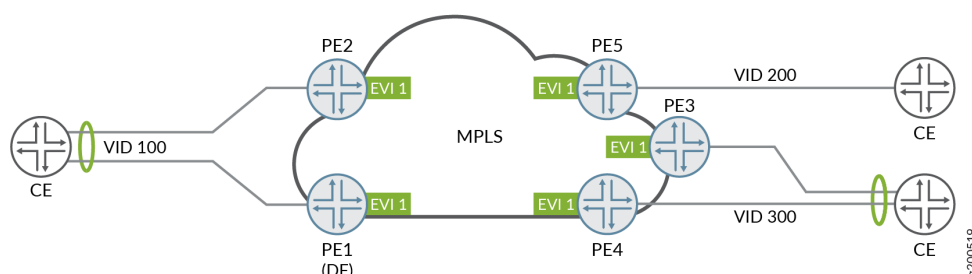
Figure 95: Single VID with no VLAN Translation



Starting with Junos OS Release 17.3R1, VLAN-based service with VID translation as described in RFC 7432 is supported. This means that Junos supports VID translation and the customer can have a different VID for each VLAN. As described in the RFC, the VID translation must be performed at the

egress PE device while the MPLS encapsulated frames should also retain the originating VID. [Figure 96 on page 916](#) illustrates a topology where CE devices use different CE-VIDs for single VLAN-based EVI.

Figure 96: Multiple VIDs with VLAN Translation



For more information on configuring VLAN-based service, see ["Configuring EVPN with VLAN-Based Service" on page 498](#).

The following is a sample configuration for a single VLAN-based EVI. The same VID is used on all the PE devices, so VLAN translation is not required. In this example, the `vlan-id=none` statement is included to remove the originating VID and to set the Ethernet tag ID to zero in the MPLS frame.

```
interfaces {
  xe-0/0/1 {
    unit 100 {
      encapsulation vlan-bridge;
      vlan-id 100;
    }
  }
}
routing-instances evpn-vlan-based-no-vid {
  instance-type evpn;
  vlan-id none;
  interface xe-0/0/1.100;
  route-distinguisher 10.0.0.1:100;
  vrf-target target:65303:101100;
  protocols evpn;
}
```

The following is a sample configuration for a single VLAN-based EVI. The same VID is used on all the PE devices, so VLAN translation is not required. In this example, the CE-VID is used and sent as part of the MPLS frame.

```

interfaces {
  xe-0/0/1 {
    unit 100 {
      encapsulation vlan-bridge;
      vlan-id 100;
    }
  }
}
routing-instances evpn-vlan-based-with-vid {
  instance-type evpn;
  interface xe-0/0/1.100;
  route-distinguisher 10.0.0.1:100;
  vrf-target target:65303:101100;
  protocols evpn;
}

```

Starting with Junos OS Release 17.3R1, Junos supports VLAN-based service with translation as described in RFC 7432. The following is a sample VLAN-based service configuration that adheres to a strict compliance of RFC 7432. Strict compliance to RFC 7432 requires that translation occurs at the egress PE device, the originating VID be carried in the MPLS frame, and the Ethernet tag ID be set to zero for all EVPN routes. Therefore, the `VLAN-id=none` and the `no-normalization` statements are included. This will set the Ethernet tag ID to zero, while ensuring that different VIDs can still be used.

```

interfaces {
  xe-0/0/1 {
    unit 100 {
      encapsulation vlan-bridge;
      vlan-id 100;
      output-vlan-map {
        swap;
      }
    }
  }
}
routing-instances evpn-vlan-based-normalization-strict-RFC-compliance {
  instance-type evpn;
  vlan-id none;
}

```



```

no-normalization;
interface xe-0/0/1.100;
route-distinguisher 10.0.0.1:100;
vrf-target target:65303:101100;
protocols evpn;
}

```

The following is a sample VLAN-based service configuration that adheres to RFC 7432 except for the condition that the originating VID be carried in the Ethernet frame. The originating VID is removed and the Ethernet tag ID is set to zero.

```

interfaces {
  xe-0/0/1 {
    unit 100 {
      encapsulation vlan-bridge;
      vlan-id 100;
    }
  }
}
routing-instances evpn-vlan-based-normalization-loose-RFC-compliance {
  instance-type evpn;
  vlan-id none;
  interface xe-0/0/1.100;
  route-distinguisher 10.0.0.1:100;
  vrf-target target:65303:101100;
  protocols evpn;
}

```

RELATED DOCUMENTATION

[Stacking and Rewriting Gigabit Ethernet VLAN Tags Overview](#)

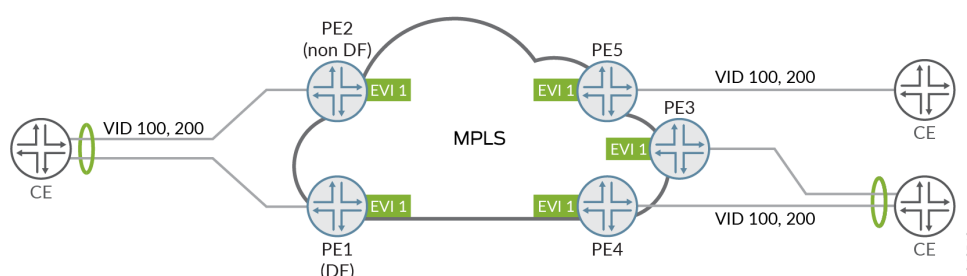
VLAN Bundle Service for EVPN

Starting with Junos OS Release 17.1, VLAN bundle service allows multiple broadcast domains to map to a single bridge domain. Multiple VLANs are mapped to a single EVPN instance (EVI) and share the same bridge table in the MAC-VRF table, thus reducing the number of routes and labels stored in the table. This lowers the control plane overhead on the router. Having a single bridge domain requires all CE

devices in the VLAN network to have unique MAC addresses. VLAN ID translation is not permitted as the MPLS encapsulated frames must retain the originating VLAN ID. As such the Ethernet Tag ID in all EVPN routes is set to zero. A VLAN range cannot be specified. The entire VLAN from 1 to 4095 must be bundled on the interface.

Figure 97 on page 919 illustrates a topology where VID 100 and VID 200 are bundled and assigned to EVI 1. The service provider creates a single broadcast domain for the customer and assigns a preconfigured number of CE-VIDs on the ingress PE routers (PE1 through PE5) to EVI 1. All CE devices use the same CE-VIDs for the EVI.

Figure 97: VLAN Bundle Network Topology



Using the VLAN bundle service reduces the number of routes and labels, which in turn, reduces the control plane overhead. The trade-off for the ease of provisioning in a customer network is that the service provider has no control over the customer broadcast domain since there is a single inclusive multicast tree and no CE-VID translation. .

NOTE: Junos OS also supports port-based VLAN bundle service where all of the VLANs on a port are part of the same service and are mapped to the same bundle.

NOTE: Integrated routing and bridging (IRB) is not supported for VLAN bundle service.

NOTE: Arp suppression is automatically disabled when you configure EVPN bundle services. As a result, you may observe ARP packets being sent by the device.

Release History Table

Release	Description
17.1	Starting with Junos OS Release 17.1, VLAN bundle service allows multiple broadcast domains to map to a single bridge domain.

RELATED DOCUMENTATION

[Configuring EVPN VLAN Bundle Services](#) | 920

Configuring EVPN VLAN Bundle Services

To configure EVPN VLAN bundle services , complete the following configuration on all PE routers within the EVPN service provider's network:

1. Configure the EVPN routing instance name using the `routing-instances` statement at the [edit] hierarchy level:

```
routing-instances routing-instance-name {...}
```

2. Configure the `evpn` option for the `instance-type` statement at the [edit `routing-instances routing-instance-name`] hierarchy level:

```
instance-type evpn;
```

3. Configure a route distinguisher on a PE router by including the `route-distinguisher` statement:

```
route-distinguisher (as-number: number | ip-address: number);
```

Each routing instance that you configure on a PE router must have a unique route distinguisher associated with it. VPN routing instances need a route distinguisher to help BGP to distinguish between potentially identical network layer reachability information (NLRI) messages received from different VPNs. If you configure different VPN routing instances with the same route distinguisher, the commit fails.

For a list of the hierarchy levels at which you can include the `route-distinguisher` statement, see "[route-distinguisher](#)" on page 1651.

The route distinguisher is a 6-byte value that you can specify in either of the following formats:

- *as-number:number*, where *as-number* is an autonomous system (AS) number (a 2-byte value) and *number* is any 4-byte value. The AS number can be in the range 1 through 65,535. We recommend that you use an Internet Assigned Numbers Authority (IANA)-assigned, nonprivate AS number, preferably the ISP's own or the customer's own AS number.

NOTE: The automatic derivation of the BGP route target (auto-RT) for advertised prefixes is supported on 2-byte AS numbers only.

- *ip-address:number*, where *ip-address* is an IP address (a 4-byte value) and *number* is any 2-byte value. The IP address can be any globally unique unicast address. We recommend that you use the address that you configure in the *router-id* statement, which is a nonprivate address in your assigned prefix range.
4. Configure either import and export policies for the EVPN routing table, or configure the default policies using the *vrf-target* statement at the [edit routing-instances *routing-instance-name*] hierarchy level.

See *Configuring Policies for the VRF Table on PE Routers in VPNs*.

5. Configure each EVPN interface for the EVPN routing instance:

NOTE: Adding a trunk port with dual tags to an EVPN and MPLS routing instance to an EVPN and VXLAN routing instance causes the CLI commit check configuration to fail and generate an error. To avoid configuration check errors, use sub-interface style interface configuration with dual tags for the routing instances.

- Configure each interface using the **"interface" on page 1599** statement at the [edit routing-instances *routing-instance-name* protocols *evpn*] hierarchy level.
- Configure interface encapsulation for the CE-facing interfaces at the [edit interfaces *interface-name* encapsulation] hierarchy level. Supported encapsulations are ethernet-bridge, vlan-bridge, and extended-vlan-bridge. .

NOTE: If you are configuring the port-based VLAN bundle service, you will need to also configure the interface encapsulation ethernet-bridge unit to 0.

- (Optional) Include the *ignore-encapsulation-mismatch* statement at the [edit routing-instances *routing-instance-name* protocols evpn interface *interface-name*] hierarchy level to allow the EVPN to

establish a connection to the CE device even if the CE device interface encapsulation and the EVPN interface encapsulations do not match.

- Specify a static MAC address for a logical interface in a bridge domain using the *static-mac* statement at the [edit routing-instances *routing-instance-name* protocols evpn interface *interface-name*] hierarchy level.
6. Specify the maximum number of media access control (MAC) addresses that can be learned by the EVPN routing instance by including the ["interface-mac-limit" on page 1602](#) statement.
You can configure the same limit for all interfaces configured for a routing instance by including the *interface-mac-limit* statement at the [edit routing-instances *routing-instance-name* protocols evpn] hierarchy level. You can also configure a limit for a specific interface by including this statement at the [edit routing-instances *routing-instance-name* protocols evpn interface *interface-name*] hierarchy level.

By default, packets with new source MAC addresses are forwarded after the MAC address limit is reached. You can alter this behavior by including the *packet-action* drop statement at either the [edit routing-instances *routing-instance-name* protocols evpn interface-mac-limit] or the [edit routing-instances *routing-instance-name* protocols evpn interface *interface-name*] hierarchy level. If you configure this statement, packets from new source MAC addresses are dropped after the configured MAC address limit is reached.
 7. Specify the MPLS label allocation setting for the EVPN by including the *label-allocation* statement with the per-instance option at the [edit routing-instances *routing-instance-name* protocols evpn] hierarchy level.

If you configure this statement, one MPLS label is allocated for the specified EVPN routing instance.
 8. Enable MAC accounting for the EVPN by including the *mac-statistics* statement at the [edit routing-instances *routing-instance-name* protocols evpn] hierarchy level.
 9. Disable MAC learning by including the *no-mac-learning* statement at either the [edit routing-instances *routing-instance-name* protocols evpn] hierarchy level to apply this behavior to all of the devices configured for an EVPN routing instance or at the [edit routing-instances *routing-instance-name* protocols evpn interface *interface-name*] hierarchy level to apply this behavior to only one of the CE devices.

NOTE: Arp suppression is automatically disabled when you configure EVPN bundle services. As a result, you may observe ARP packets being sent by the device.

The following output shows a sample EVPN VLAN bundle services configuration.

```
user@router1# show routing-instances evpn_1
  instance-type evpn;
    interface xe-2/3/3.300;
    route-distinguisher 7.7.7.7:3;
    vrf-target target:65221:3;
```

```

        protocols evpn;
        label-allocation per-instance;
    }
}

```

RELATED DOCUMENTATION

[VLAN Bundle Service for EVPN](#) | 918

Virtual Switch Support for EVPN Overview

Starting with Junos OS Release 14.1, VLAN-aware bundle service is introduced on MX Series routers. Starting with Junos OS Release 14.2, VLAN-aware bundle service is introduced on EX9200 switches. Starting with Junos OS Release 17.3, VLAN-aware bundle service is introduced on QFX Series switches. This feature allows Ethernet VLANs over a WAN to share a single EVPN instance while maintaining data-plane separation between the different VLANs.

Junos OS has a highly flexible and scalable virtual switch interface. With a virtual switch, a single router or switch can be divided into multiple logical switches. Layer 2 domains (also called *bridge-domains* or *vlangs*) can be defined independently in each virtual switch. To configure VLAN-aware bundle service, an EVPN must run in a virtual-switch routing instance.

On the EX Series and MX Series, a single EVPN instance can stretch up to 4094 bridge domains or VLANs defined in a virtual switch to remote sites. A virtual switch can have more than 4094 bridge domains or VLANs with a combination of none, single, and dual VLANs. However, because EVPN signaling deals only with single VLAN tags, a maximum of 4094 bridge domains or VLANs can be stretched. The EVPN virtual switch also provides support for trunk and access interfaces.

Starting with Junos OS Release 17.3R1, on the QFX10000 line of switches, you can configure both EVPN and virtual-switch routing instances. The EVPN routing instance supports VLAN-based service. With VLAN-based service, the EVPN instance includes only a single broadcast domain, and there is a one-to-one mapping between a VNI and MAC-VRF. Up to 100 EVPN routing instances are supported. The virtual-switch routing instance supports VLAN-aware service, and up to 10 virtual-switch routing instances with 2000 VLANs are supported.

If you create VLANs that are not part of a routing instance, they become part of the default switch routing instance.

NOTE:

- The `none` VLAN option is supported with bridge domains or VLANs under the virtual switch instance type for EVPNs.
- Access interfaces configured with single or dual VLAN tags are supported in EVPN. By default, only Ethernet frames with single or no VLAN tags are transported across the EVPN core. As a result, dual-tagged Ethernet frames received on the access interfaces must be normalized to Ethernet frames with single or no VLAN tags for proper transmission over the EVPN core.

You can enable transporting of dual-tagged frames across the EVPN core network by including both the `vlan-id none` and `no-normalization` configuration statements together.

There are two types of VLAN-aware bundle service:

- VLAN-aware bundle without translation

The service interface provides bundling of customer VLANs into a single Layer 2 VPN service instance with a guarantee end-to-end customer VLAN transparency. The data-plane separation between the customer VLANs is maintained by creating a dedicated bridge-domain for each VLAN.

- VLAN-aware bundle with translation

The service interface provides bundling of customer VLANs into a single Layer 2 VPN service instance. The data-plane separation between the customer VLANs is maintained by creating a dedicated bridge-domain for each VLAN. The service interface supports customer VLAN translation to handle the scenario where different VLAN Identifiers (VIDs) are used on different interfaces to designate the same customer VLAN.

EVPN with virtual switch provides support for VLAN-aware bundle with translation only.

Release History Table

Release	Description
17.3R1	Starting with Junos OS Release 17.3R1, on the QFX10000 line of switches, you can configure both EVPN and virtual-switch routing instances.
17.3	Starting with Junos OS Release 17.3, VLAN-aware bundle service is introduced on QFX Series switches.
14.2	Starting with Junos OS Release 14.2, VLAN-aware bundle service is introduced on EX9200 switches.
14.1	Starting with Junos OS Release 14.1, VLAN-aware bundle service is introduced on MX Series routers.

RELATED DOCUMENTATION

[Example: Configuring EVPN with Support for Virtual Switch](#) | 929

Configuring EVPN with Support for Virtual Switch

You can configure an Ethernet VPN (EVPN) with virtual switch support to enable multiple tenants with independent VLAN and subnet space within an EVPN instance. Virtual switch provides the ability to extend Ethernet VLANs over a WAN using a single EVPN instance while maintaining data-plane separation between the various VLANs associated with that instance. A single EVPN instance can stretch up to 4094 bridge domains defined in a virtual switch to remote sites.

When configuring virtual switch for EVPN, be aware of the following considerations:

- Due to default ARP policing, some of the ARP packets not destined for the device can be missed. This can lead to delayed ARP learning and synchronization.
- Clearing ARP for an EVPN can lead to inconsistency between the ARP table and the EVPN ARP table. To avoid this situation, clear both ARP and EVPN ARP tables.
- The `vlan-tag` can be configured for local switching. However, vlan-tagged VLANs should not be extended over the EVPN cloud.

Before you begin:

1. Configure the router interfaces.
2. Configure the router ID and autonomous system number for the device.
3. Enable chained composite next hop for EVPN.
4. Configure OSPF or any other IGP protocol.
5. Configure a BGP internal group.
6. Include the EVPN signaling network layer reachability information (NLRI) to the internal BGP group.
7. Configure RSVP or LDP.
8. Configure MPLS.
9. Create a label-switched path between the provider edge (PE) devices.

To configure the PE device:

1. Configure the virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance instance-type virtual-switch
```

2. Configure the interface names for the virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance interface interface-name
```

3. Configure the route distinguisher for the virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance route-distinguisher route-distinguisher-value
```

4. Configure the VPN routing and forwarding (VRF) target community for the virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance vrf-target vrf-target
```

5. List the VLAN identifiers that are to be EVPN extended.

```
[edit routing-instances]
user@PE1# set evpn-instance protocols evpn extended-vlan-list [vlan-id-range]
```

6. Configure the bridge domain for the first virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance bridge-domains first-bridge-domain-name domain-type bridge
```

7. Assign the VLAN ID for the first bridge domain.

```
[edit routing-instances]
user@PE1# set evpn-instance bridge-domains first-bridge-domain-name vlan-id 10
```

8. Configure the IRB interface as the routing interface for the first bridge domain.

```
[edit routing-instances]
user@PE1# set evpn-instance bridge-domains first-bridge-domain-name routing-interface irb.0
```

9. Configure the interface name for the first bridge domain.

```
[edit routing-instances]
user@PE1# set evpn-instance bridge-domains first-bridge-domain-name bridge-options
interface CE-facing-interface
```

10. Configure the bridge domain for the second virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance bridge-domains second-bridge-domain-name domain-type bridge
```

11. Assign the VLAN ID for the second bridge domain.

```
[edit routing-instances]
user@PE1# set evpn-instance bridge-domains second-bridge-domain-name vlan-id VLAN-ID
```

12. Configure the IRB interface as the routing interface for the second bridge domain.

```
[edit routing-instances]
user@PE1# set evpn-instance bridge-domains second-bridge-domain-name routing-interface irb.1
```

13. Configure the interface name for the second bridge domain.

```
[edit routing-instances]
user@PE1# set evpn-instance bridge-domains second-bridge-domain-name bridge-options
interface CE-facing-interface
```

14. Configure the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf-instance instance-type vrf
```

15. Configure the IRB interface as the routing interface for the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf-instance interface irb.0
user@PE1# set vrf-instance interface irb.1
```

16. Configure the route distinguisher for the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf-instance route-distinguisher route-distinguisher-value
```

17. Configure the VRF target community for the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf-instance vrf-target vrf-target
```

18. Configure the VRF label for the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf-instance vrf-table-label
```

19. Verify and commit the configuration.

For example:

```
[edit routing-instances]
user@PE1# set evpna instance-type virtual-switch
user@PE1# set evpna interface ge-0/1/4.0
user@PE1# set evpna interface ge-0/1/4.1
user@PE1# set evpna route-distinguisher 10.255.169.37:1
user@PE1# set evpna vrf-target target:100:1
user@PE1# set evpna protocols evpn extended-vlan-list [ 10 20 ]
user@PE1# set evpna bridge-domains bda domain-type bridge
user@PE1# set evpna bridge-domains bda vlan-id 10
user@PE1# set evpna bridge-domains bda routing-interface irb.0
user@PE1# set evpna bridge-domains bda bridge-options interface ge-0/1/4.0
user@PE1# set evpna bridge-domains bdb domain-type bridge
user@PE1# set evpna bridge-domains bdb vlan-id 20
user@PE1# set evpna bridge-domains bdb routing-interface irb.1
user@PE1# set evpna bridge-domains bdb bridge-options interface ge-0/1/4.1
user@PE1# set vrf instance-type vrf
```

```

user@PE1# set vrf interface irb.0
user@PE1# set vrf interface irb.1
user@PE1# set vrf route-distinguisher 192.0.2.1:2
user@PE1# set vrf vrf-target target:100:2
user@PE1# set vrf vrf-table-label

```

```

[edit]
user@PE1# commit
commit complete

```

RELATED DOCUMENTATION

[Example: Configuring EVPN with Support for Virtual Switch | 929](#)

Example: Configuring EVPN with Support for Virtual Switch

IN THIS SECTION

- [Example: Configuring EVPN with Support for Virtual Switch | 929](#)

Example: Configuring EVPN with Support for Virtual Switch

IN THIS SECTION

- [Requirements | 930](#)
- [Overview | 930](#)
- [Configuration | 931](#)
- [Verification | 942](#)

This example shows how to configure a virtual switch in an Ethernet VPN (EVPN) deployment.

Requirements

This example uses the following hardware and software components:

- Two MX Series 5G Universal Routing Platforms containing MPC FPCs.
- Two customer edge (CE) routers.
- Junos OS Release 14.1 or later.

Before you begin:

1. Configure the router interfaces.
2. Configure OSPF or any other IGP protocol.
3. Configure BGP.
4. Configure RSVP or LDP.
5. Configure MPLS.

Overview

IN THIS SECTION

- [Topology | 931](#)

Starting with Junos OS Release 14.1, the Ethernet VPN (EVPN) solution on MX Series routers with MPC interfaces is extended to provide virtual switch support that enables multiple tenants with independent VLAN and subnet space within an EVPN instance. Virtual switch provides the ability to extend Ethernet VLANs over a WAN using a single EVPN instance while maintaining data-plane separation between the various VLANs associated with that instance. A single EVPN instance can stretch up to 4094 bridge domains defined in a virtual switch to remote sites.

When configuring virtual switch for EVPN, be aware of the following considerations:

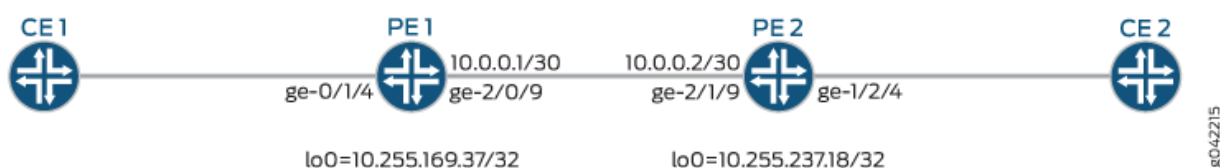
- Due to default ARP policing, some of the ARP packets not destined for the device can be missed. This can lead to delayed ARP learning and synchronization.
- Clearing ARP for an EVPN can lead to inconsistency between the ARP table and the EVPN ARP table. To avoid this situation, clear both ARP and EVPN ARP tables.

- The `vlan-tag` can be configured for local switching. However, `vlan-tagged` VLANs should not be extended over the EVPN cloud.

Topology

Figure 98 on page 931 illustrates a simple EVPN topology with virtual switch support. Routers PE1 and PE2 are the provider edge (PE) routers that connect to one customer edge (CE) router each – CE1 and CE2.

Figure 98: EVPN with Virtual Switch Support



Configuration

IN THIS SECTION

- Procedure | 931
- Results | 939

Procedure

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the `[edit]` hierarchy level.

PE1

```
set interfaces ge-2/0/9 unit 0 family inet address 10.0.0.1/30
set interfaces ge-2/0/9 unit 0 family mpls
set interfaces ge-0/1/4 flexible-vlan-tagging
set interfaces ge-0/1/4 encapsulation flexible-ethernet-services
```

```

set interfaces ge-0/1/4 unit 0 family bridge interface-mode trunk
set interfaces ge-0/1/4 unit 0 vlan-id-list 10
set interfaces ge-0/1/4 unit 1 family bridge interface-mode trunk
set interfaces ge-0/1/4 unit 1 vlan-id-list 20
set interfaces irb unit 0 family inet address 192.168.1.1/16
set interfaces irb unit 1 family inet address 192.168.2.1/16
set interfaces lo0 unit 0 family inet address 10.255.169.37/32
set routing-options router-id 10.255.169.37
set routing-options autonomous-system 100
set routing-options forwarding-table chained-composite-next-hop ingress evpn
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path PE1-to-PE2 from 10.255.169.37
set protocols mpls label-switched-path PE1-to-PE2 to 10.255.237.18
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 10.255.169.37
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 10.255.237.18
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set routing-instances evpna instance-type virtual-switch
set routing-instances evpna interface ge-0/1/4.0
set routing-instances evpna interface ge-0/1/4.1
set routing-instances evpna route-distinguisher 10.255.169.37:1
set routing-instances evpna vrf-target target:100:1
set routing-instances evpna protocols evpn extended-vlan-list [ 10 20 ]
set routing-instances evpna bridge-domains bda domain-type bridge
set routing-instances evpna bridge-domains bda vlan-id 10
set routing-instances evpna bridge-domains bda routing-interface irb.0
set routing-instances evpna bridge-domains bda bridge-options interface ge-0/1/4.0
set routing-instances evpna bridge-domains bdb domain-type bridge
set routing-instances evpna bridge-domains bdb vlan-id 20
set routing-instances evpna bridge-domains bdb routing-interface irb.1
set routing-instances evpna bridge-domains bdb bridge-options interface ge-0/1/4.1
set routing-instances vrf instance-type vrf
set routing-instances vrf interface irb.0
set routing-instances vrf interface irb.1
set routing-instances vrf route-distinguisher 198.51.100.1:2
set routing-instances vrf vrf-target target:100:2
set routing-instances vrf vrf-table-label

```

PE2

```

set interfaces ge-2/1/9 unit 0 family inet address 10.0.0.2/30
set interfaces ge-2/1/9 unit 0 family mpls
set interfaces ge-1/2/4 flexible-vlan-tagging
set interfaces ge-1/2/4 encapsulation flexible-ethernet-services
set interfaces ge-1/2/4 unit 0 family bridge interface-mode trunk
set interfaces ge-1/2/4 unit 0 vlan-id-list 10
set interfaces ge-1/2/4 unit 1 family bridge interface-mode trunk
set interfaces ge-1/2/4 unit 1 vlan-id-list 20
set interfaces irb unit 0 family inet address 192.168.2.2/16
set interfaces irb unit 1 family inet address 192.168.2.3/16
set interfaces lo0 unit 0 family inet address 10.255.237.18/32
set routing-options router-id 10.255.237.18
set routing-options autonomous-system 100
set routing-options forwarding-table chained-composite-next-hop ingress evpn
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path PE2-to-PE1 from 10.255.237.18
set protocols mpls label-switched-path PE2-to-PE1 to 10.255.169.37
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 10.255.237.18
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 10.255.169.37
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set routing-instances evpna instance-type virtual-switch
set routing-instances evpna interface ge-1/2/4.0
set routing-instances evpna interface ge-1/2/4.1
set routing-instances evpna route-distinguisher 10.255.237.18:1
set routing-instances evpna vrf-target target:100:1
set routing-instances evpna protocols evpn extended-vlan-list [ 10 20 ]
set routing-instances evpna bridge-domains bda domain-type bridge
set routing-instances evpna bridge-domains bda vlan-id 10
set routing-instances evpna bridge-domains bda routing-interface irb.0
set routing-instances evpna bridge-domains bda bridge-options interface ge-1/2/4.0
set routing-instances evpna bridge-domains bdb domain-type bridge
set routing-instances evpna bridge-domains bdb vlan-id 20
set routing-instances evpna bridge-domains bdb routing-interface irb.1
set routing-instances evpna bridge-domains bdb bridge-options interface ge-1/2/4.1

```



```

set routing-instances vrf instance-type vrf
set routing-instances vrf interface irb.0
set routing-instances vrf interface irb.1
set routing-instances vrf route-distinguisher 198.51.100.2:2
set routing-instances vrf vrf-target target:100:2
set routing-instances vrf vrf-table-label

```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure Router PE1:

NOTE: Repeat this procedure for Router PE2, after modifying the appropriate interface names, addresses, and other parameters.

1. Configure the PE1 interfaces.

```

[edit interfaces]
user@PE1# set ge-2/0/9 unit 0 family inet address 10.0.0.1/30
user@PE1# set ge-2/0/9 unit 0 family mpls
user@PE1# set ge-0/1/4 flexible-vlan-tagging
user@PE1# set ge-0/1/4 encapsulation flexible-ethernet-services
user@PE1# set ge-0/1/4 unit 0 family bridge interface-mode trunk
user@PE1# set ge-0/1/4 unit 0 vlan-id-list 10
user@PE1# set ge-0/1/4 unit 1 family bridge interface-mode trunk
user@PE1# set ge-0/1/4 unit 1 vlan-id-list 20
user@PE1# set irb unit 0 family inet address 192.168.1.1/16
user@PE1# set irb unit 1 family inet address 192.168.2.1/16
user@PE1# set lo0 unit 0 family inet address 10.255.169.37/32

```

2. Set the router ID and autonomous system number for Router PE1.

```

[edit routing-options]
user@PE1# set router-id 10.255.169.37
user@PE1# set autonomous-system 100

```

3. Configure the chained composite next hop for EVPN.

```
[edit routing-options]
user@PE1# set forwarding-table chained-composite-next-hop ingress evpn
```

4. Enable RSVP on all the interfaces of Router PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set rsvp interface all
user@PE1# set rsvp interface fxp0.0 disable
```

5. Create label-switched paths for PE1 to reach PE2.

```
[edit protocols]
user@PE1# set mpls label-switched-path PE1-to-PE2 from 10.255.169.37
user@PE1# set mpls label-switched-path PE1-to-PE2 to 10.255.237.18
```

6. Enable MPLS on all the interfaces of Router PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set mpls interface all
user@PE1# set mpls interface fxp0.0 disable
```

7. Configure the BGP group for Router PE1.

```
[edit protocols]
user@PE1# set bgp group ibgp type internal
```

8. Assign local and neighbor addresses to the ibgp BGP group for Router PE1 to peer with Router PE2.

```
[edit protocols]
user@PE1# set bgp group ibgp local-address 10.255.169.37
user@PE1# set bgp group ibgp neighbor 10.255.237.18
```

9. Include the EVPN signaling Network Layer Reachability Information (NLRI) to the ibgp BGP group.

```
[edit protocols]
user@PE1# set bgp group ibgp family evpn signaling
```

10. Configure OSPF on all the interfaces of Router PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set ospf area 0.0.0.0 interface all
user@PE1# set ospf area 0.0.0.0 interface fxp0.0 disable
```

11. Configure the virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set evpna instance-type virtual-switch
```

12. Configure the interface name for the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna interface ge-0/1/4.0
user@PE1# set evpna interface ge-0/1/4.1
```

13. Configure the route distinguisher for the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna route-distinguisher 10.255.169.37:1
```

14. Configure the VPN routing and forwarding (VRF) target community for the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna vrf-target target:100:1
```

15. List the VLAN identifiers that are to be EVPN extended.

```
[edit routing-instances]
user@PE1# set evpna protocols evpn extended-vlan-list [ 10 20 ]
```

16. Configure the bridge domains for the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna bridge-domains bda domain-type bridge
```

17. Assign the VLAN ID for the bda bridge domain.

```
[edit routing-instances]
user@PE1# set evpna bridge-domains bda vlan-id 10
```

18. Configure the IRB interface as the routing interface for the bda bridge domain.

```
[edit routing-instances]
user@PE1# set evpna bridge-domains bda routing-interface irb.0
```

19. Configure the interface name for the bda bridge domain.

```
[edit routing-instances]
user@PE1# set evpna bridge-domains bda bridge-options interface ge-0/1/4.0
```

20. Configure the bridge domains for the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna bridge-domains bdb domain-type bridge
```

21. Assign the VLAN ID for the bdb bridge domain.

```
[edit routing-instances]
user@PE1# set evpna bridge-domains bdb vlan-id 20
```

22. Configure the IRB interface as the routing interface for the bda bridge domain.

```
[edit routing-instances]
user@PE1# set evpna bridge-domains bdb routing-interface irb.1
```

23. Configure the interface name for bdb bridge domain.

```
[edit routing-instances]
user@PE1# set evpna bridge-domains bdb bridge-options interface ge-0/1/4.1
```

24. Configure the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf instance-type vrf
```

25. Configure the IRB interface as the routing interface for the vrf routing instance.

```
[edit routing-instances]
user@PE1# set vrf interface irb.0
user@PE1# set vrf interface irb.1
```

26. Configure the route distinguisher for the vrf routing instance.

```
[edit routing-instances]
user@PE1# set vrf route-distinguisher 198.51.100.1:2
```

27. Configure the VRF target community for the vrf routing instance.

```
[edit routing-instances]
user@PE1# set vrf vrf-target target:100:2
```

28. Configure VRF label for the vrf routing instance.

```
[edit routing-instances]
user@PE1# set vrf vrf-table-label
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show routing-options`, `show protocols`, and `show routing-instances` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```

user@PE1# show interfaces
ge-2/0/9 {
  unit 0 {
    family inet {
      address 10.0.0.1/30;
    }
    family mpls;
  }
}
ge-0/1/4 {
  flexible-vlan-tagging;
  encapsulation flexible-ethernet-services;
  unit 0 {
    family bridge {
      interface-mode trunk;
      vlan-id-list 10;
    }
  }
  unit 1 {
    family bridge {
      interface-mode trunk;
      vlan-id-list 20;
    }
  }
}
irb {
  unit 0 {
    family inet {
      address 192.168.1.1/16;
    }
  }
  unit 1 {
    family inet {
      address 192.168.2.1/16;
    }
  }
}

```

```

}
lo0 {
    unit 0 {
        family inet {
            address 10.255.169.37/32;
        }
    }
}

```

```

user@PE1# show routing-options
router-id 10.255.169.37;
autonomous-system 100;
forwarding-table {
    chained-composite-next-hop {
        ingress {
            evpn;
        }
    }
}

```

```

user@PE1# show protocols
rsvp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
mpls {
    label-switched-path PE1-to-PE2 {
        from 10.255.169.37;
        to 10.255.237.18;
    }
    interface all;
    interface fxp0.0 {
        disable;
    }
}
bgp {
    group ibgp {
        type internal;
    }
}

```

```

        local-address 10.255.169.37;
        family evpn {
            signaling;
        }
        neighbor 10.255.237.18;
    }
}
ospf {
    area 0.0.0.0 {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
}
}

```

```

user@PE1# show routing-instances
evpna {
    instance-type virtual-switch;
    interface ge-0/1/4.0;
    interface ge-0/1/4.1;
    route-distinguisher 10.255.169.37:1;
    vrf-target target:100:1;
    protocols {
        evpn {
            extended-vlan-list [ 10 20 ];
        }
    }
    bridge-domains {
        bda {
            domain-type bridge;
            vlan-id 10;
            routing-interface irb.0;
            bridge-options {
                interface ge-0/1/4.0;
            }
        }
        bdb {
            domain-type bridge;
            vlan-id 20;
            routing-interface irb.1;
        }
    }
}

```



```

        bridge-options {
            interface ge-0/1/4.1;
        }
    }
}
vrf {
    instance-type vrf;
    interface irb.0;
    interface irb.1;
    route-distinguisher 10.255.169.37:2;
    vrf-target target:100:2;
    vrf-table-label;
}

```

Verification

IN THIS SECTION

- [Verifying the Bridge Domain Configuration | 942](#)
- [Verifying MAC Table Routes | 943](#)
- [Verifying the Bridge EVPN Peer Gateway MAC | 944](#)

Confirm that the configuration is working properly.

Verifying the Bridge Domain Configuration

Purpose

Verify the bridge domain configuration for the evpna routing instance.

Action

From operational mode, run the `show bridge domain extensive` command.

```

user@PE1> show bridge domain extensive
Routing instance: evpna

```

```

Bridge domain: bda                                State: Active
Bridge VLAN ID: 10                                EVPN extended: Yes
Interfaces:
    ge-0/1/4.0
    pip-10.000010000000
    pip-10.feff0f000000
Total MAC count: 2

Bridge domain: bdb                                State: Active
Bridge VLAN ID: 20                                EVPN extended: Yes
Interfaces:
    ge-0/1/4.1
    pip-11.010010000000
    pip-11.ffff0f000000
Total MAC count: 2

```

Meaning

The configured bridge domains bda and bdb and their associated VLAN IDs and interfaces are displayed. The bridge domains are also extended with EVPN.

Verifying MAC Table Routes

Purpose

Verify the MACs learned in the data plane and control plane.

Action

From operational mode, run the show bridge mac-table command.

```

user@PE1> show bridge mac-table
MAC flags (S -static MAC, D -dynamic MAC, L -locally learned, C -Control MAC
          SE -Statistics enabled, NM -Non configured MAC, R -Remote PE MAC)

Routing instance : evpna
Bridging domain : bda, VLAN : 10

```

MAC address	MAC flags	Logical interface	NH Index	RTR ID
00:00:00:aa:01:01	S	ge-0/1/4.0		
00:00:00:bb:01:01	DC		1048574	1048574

```

00:00:00:cc:01:01  DC                               1048576 1048576

Bridging domain : bdb, VLAN : 20
MAC              MAC      Logical      NH      RTR
address          flags    interface  Index   ID
00:00:00:aa:02:01 S      ge-0/1/4.1
00:00:00:bb:02:01 DC                               1048575 1048575
00:00:00:cc:02:01 DC                               1048577 1048577

```

Meaning

The configured static MACs for the bridge domains are displayed.

Verifying the Bridge EVPN Peer Gateway MAC

Purpose

Verify the bridge EVPN peer gateway MAC for the evpna routing instance.

Action

From operational mode, run the `show bridge evpn peer-gateway-macs` command.

```

user@PE1> show bridge evpn peer-gateway-macs
Routing instance : evpna
Bridging domain : bda, VLAN : 10
  Installed GW MAC addresses:
    00:23:9c:96:af:f0
    a8:d0:e5:5b:02:08

Bridging domain : bdb, VLAN : 20
  Installed GW MAC addresses:
    00:23:9c:96:af:f0
    a8:d0:e5:5b:02:08

```

Meaning

The gateway MACs of the EVPN peers for the evpna routing instance are displayed.

SEE ALSO

| [Virtual Switch Support for EVPN Overview](#) | 504

Configuring Integrated Bridging and Routing

IN THIS CHAPTER

- [EVPN with IRB Solution Overview | 946](#)
- [An EVPN with IRB Solution on EX9200 Switches Overview | 952](#)
- [Anycast Gateways | 958](#)
- [Configuring EVPN with IRB Solution | 962](#)
- [Configuring an EVPN with IRB Solution on EX9200 Switches | 965](#)
- [Example: Configuring EVPN with IRB Solution | 968](#)
- [Example: Configuring an EVPN with IRB Solution on EX9200 Switches | 995](#)

EVPN with IRB Solution Overview

IN THIS SECTION

- [Need for an EVPN IRB Solution | 947](#)
- [Implementing the EVPN IRB Solution | 948](#)
- [Benefits of Implementing the EVPN IRB Solution | 950](#)

A Data Center Service Provider (DCSP) hosts the data center for its multiple customers on a common physical network. To each customer (also called a tenant), the service looks like a full-fledged data center that can expand to 4094 VLANs and all private subnets. For disaster recovery, high availability, and optimization of resource utilization, it is common for the DCSP to span across the data center to more than one site. To deploy the data center services, a DCSP faces the following main challenges:

- Extending Layer 2 domains across more than one data center site. This requires optimal intra-subnet traffic forwarding.

- Supporting optimal inter-subnet traffic forwarding and optimal routing in the event of virtual machine (VM).
- Supporting multiple tenants with independent VLAN and subnet space.

Ethernet VPN (EVPN) is targeted to handle all of the above mentioned challenges, wherein:

- The basic EVPN functionality enables optimal intra-subnet traffic forwarding
- Implementing the integrated routing and bridging (IRB) solution in an EVPN deployment enables optimal inter-subnet traffic forwarding
- Configuring EVPN with virtual switch support enables multiple tenants with independent VLAN and subnet space

The following sections describe the IRB solution for EVPNs:

Need for an EVPN IRB Solution

EVPN is a technology used to provide Layer 2 extension and interconnection across an IP/MPLS core network to different physical sites belonging to a single Layer 2 domain. In a data center environment with EVPN, there is a need for both Layer 2 (intra-subnet traffic) and Layer 3 (inter-subnet traffic) forwarding and potentially interoperation with tenant Layer 3 VPNs.

With only a Layer 2 solution, there is no optimum forwarding of inter-subnet traffic, even when the traffic is local, for instance, when both the subnets are on the same server.

With only a Layer 3 solution, the following issues for intra-subnet traffic can arise:

- MAC address aliasing issue where duplicate MAC addresses are not detected.
- TTL issue for applications that use TTL 1 to confine traffic within a subnet.
- IPv6 link-local addressing and duplicate address detection that relies on Layer 2 connectivity.
- Layer 3 forwarding does not support the forwarding semantics of a subnet broadcast.
- Support of non-IP applications that require Layer 2 forwarding.

Because of the above mentioned shortcomings of a pure Layer 2 and Layer 3 solution, there is a need for a solution incorporating optimal forwarding of both Layer 2 and Layer 3 traffic in the data center environment when faced with operational considerations such as Layer 3 VPN interoperability and virtual machine (VM) mobility.

An EVPN-based integrated routing and bridging (IRB) solution provides optimum unicast and multicast forwarding for both intra-subnets and inter-subnets within and across data centers.

The EVPN IRB feature is useful for service providers operating in an IP/MPLS network that provides both Layer 2 VPN or VPLS services and Layer 3 VPN services who want to extend their service to provide cloud computation and storage services to their existing customers.

Implementing the EVPN IRB Solution

An EVPN IRB solution provides the following:

- Optimal forwarding for intra-subnet (Layer 2) traffic.
- Optimal forwarding for inter-subnet (Layer 3) traffic.
- Support for ingress replication for multicast traffic.
- Support for network-based as well as host-based overlay models.
- Support for consistent policy-based forwarding for both Layer 2 and Layer 3 traffic.
- Support for the following routing protocols on the IRB interface:
 - BFD
 - BGP
 - IS-IS
 - OSPF and OSPF version 3
- Support for single-active and all-active multihoming

Junos OS supports several models of EVPN configuration to satisfy the individual needs of EVPN and data center cloud services customers. To provide flexibility and scalability, multiple bridge domains can be defined within a particular EVPN instance. Likewise, one or more EVPN instances can be associated with a single Layer 3 VPN virtual routing and forwarding (VRF). In general, each data center tenant is assigned a unique Layer 3 VPN VRF, while a tenant could comprise one or more EVPN instances and one or more bridge domains per EVPN instance. To support this model, each configured bridge domain (including the default bridge domain for an EVPN instance) requires an IRB interface to perform the Layer 2 and Layer 3 functions. Each bridge domain or IRB interface maps to a unique IP subnet in the VRF.

NOTE: You can associate an IRB interface with the primary instance inet.0 table instead of a VRF in an EVPN IRB solution.

There are two major functions that are supported for IRB in EVPN.

- Host MAC-IP synchronization

This includes:

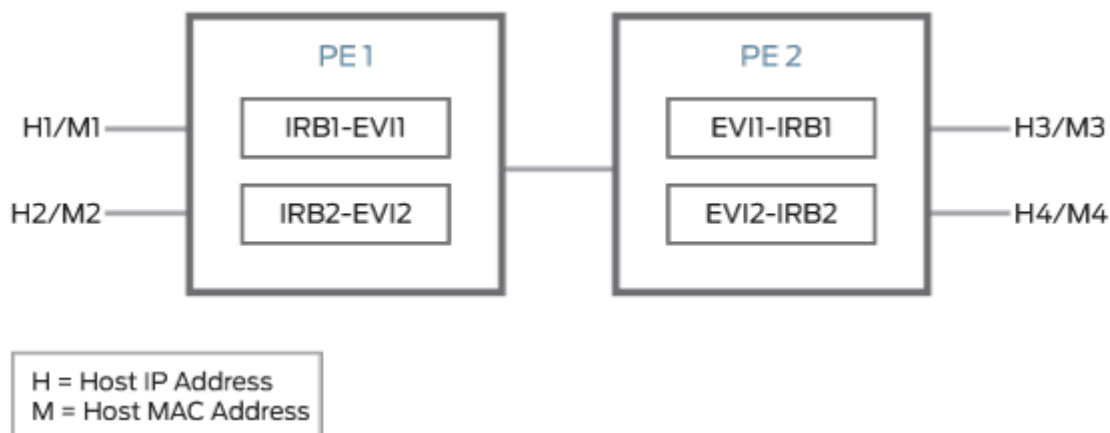
- Advertising the IP address along with the MAC advertisement route in EVPN. This is done by using the IP field in the EVPN MAC advertisement route.
- The receiving PE router installs MAC into the EVPN instance (EVI) table and installs IP into the associated VRF.
- Gateway MAC-IP synchronization

This includes:

- Advertising all local IRB MAC and IP addresses in an EVPN. This is achieved by including the default gateway extended community in the EVPN MAC advertisement route.
- The receiving PE creates a forwarding state to route packets destined for the gateway MAC, and a proxy ARP is done for the gateway IP with the MAC advertised in the route.

Figure 99 on page 949 illustrates the inter-subnet traffic forwarding between two provider edge (PE) devices – PE1 and PE2. The IRB1 and IRB2 interfaces on each PE device belong to a different subnet, but they share a common VRF.

Figure 99: Inter-Subnet Traffic Forwarding



The inter-subnet traffic forwarding is performed as follows:

1. PE2 advertises H3-M3 and H4-M4 binding to PE1. Similarly PE1 advertises H1-M1 and H2-M2 binding to PE2.
2. PE1 and PE2 install the MAC address in the corresponding EVI MAC table, whereas the IP routes are installed in the shared VRF.
3. The advertising PE device is set as the next hop for the IP routes.

4. If H1 sends packets to H4, the packets are sent to IRB1 on PE1.
5. IP lookup for H4 happens in the shared VRF on PE1. Because the next hop for the H4 IP is PE2 (the advertising PE), an IP unicast packet is sent to PE2.
6. PE1 rewrites the MAC header based on the information in the VRF route, and PE2 performs a MAC lookup to forward the packet to H4.

Benefits of Implementing the EVPN IRB Solution

The main goal of the EVPN IRB solution is to provide optimal Layer 2 and Layer 3 forwarding. The solution is required to efficiently handle inter-subnet forwarding as well as virtual machine (VM) mobility. VM mobility refers to the ability of a VM to migrate from one server to another within the same or a different data center while retaining its existing MAC and IP address. Providing optimal forwarding for inter-subnet traffic and effective VM mobility involves solving two problems – the default gateway problem and the triangular routing problem.

Starting in Junos OS Release 17.1R1, IPv6 addresses are supported on IRB interfaces with EVPN using the Neighbor Discovery Protocol (NDP). The following capabilities are introduced for IPv6 support with EVPN:

- IPv6 addresses on IRB interfaces in primary routing instances
- Learning IPv6 neighborhood from solicited NA message
- NS and NA packets on the IRB interfaces are disabled from network core
- Virtual gateway addresses are used as Layer 3 addresses
- Host MAC-IP synchronization for IPv6

You can configure the IPv6 addresses in the IRB interface at the `[edit interfaces irb]` hierarchy level.

Gateway MAC and IP Synchronization

In an EVPN IRB deployment, the IP default gateway for a VM is the IP address configured on the IRB interface of the provider edge (PE) router corresponding to the bridge domain or VLAN of which the VM is a member. The default gateway problem arises because a VM does not flush its ARP table when relocating from one server to another and continues sending packets with the destination MAC address set to that of the original gateway. If the old and new servers are not part of the same Layer 2 domain (the new Layer 2 domain could be within the current data center or a new data center), the gateway previously identified is no longer the optimal or local gateway. The new gateway needs to identify packets containing the MAC addresses of other gateways on remote PE routers and forward the traffic as if the packets were destined to the local gateway itself. At the minimum, this functionality requires each PE router to advertise its gateway or IRB MAC and IP addresses to all other PE routers in the

network. The gateway address exchange can be accomplished using the standard MAC route advertisement message (including the IP address parameter) and tagging that route with the default gateway extended community so that the remote PE routers can distinguish the gateway MAC advertisement routes from normal MAC advertisement routes.

Layer 3 VPN Interworking

The inter-data center aspect of the EVPN IRB solution involves routing between VMs that are present in different data centers or routing between a host site completely outside of the data center environment and a VM within a data center. This solution relies on the ability of EVPN MAC route advertisements to carry both MAC address and IP address information. The local MAC learning functionality of the PE router is extended to also capture IP address information associated with MAC addresses learned locally. That IP-MAC address mapping information is then distributed to each PE router through normal EVPN procedures. When a PE router receives such MAC and IP information, it installs the MAC route in the EVPN instance as well as a host route for the associated IP address in the Layer 3 VPN VRF corresponding to that EVPN instance. When a VM moves from one data center to another, normal EVPN procedures result in the MAC and IP address being advertised from the new PE router which the VM resides behind. The host route installed in the VRF associated with an EVPN solicits Layer 3 traffic destined to that VM to the new PE router and avoids triangular routing between the source, the former PE router the VM resided behind, and the new PE router.

BGP scalability is a potential concern with the inter-data center triangular routing avoidance solution because of the potential for injection of many host routes into Layer 3 VPN. With the method previously described, in the worst case there is an IP host route for each MAC address learned through the local EVPN MAC learning procedures or through a MAC advertisement message received from a remote PE router. BGP route target filtering can be used to limit distribution of such routes.

The following functional elements are required to implement the inter-data center triangular routing avoidance using Layer 3 inter-subnet forwarding procedures:

1. The source host sends an IP packet using its own source MAC and IP address with the destination MAC of the IRB interface of the local PE router and the IP address of the destination host.
2. When the IRB interface receives the frame with its MAC as the destination, it performs a Layer 3 lookup in the VRF associated with the EVPN instance to determine where to route the packet.
3. In the VRF, the PE router finds the Layer 3 route derived from a MAC plus an IP EVPN route received from the remote PE router earlier. The destination MAC address is then changed to the destination MAC address corresponding to the destination IP.
4. The packet is then forwarded to the remote PE router serving the destination host using MPLS, using the label corresponding to the EVPN instance of which the destination host is a member.

5. The egress PE router receiving the packet performs a Layer 2 lookup for the destination host's MAC and sends the packet to the destination host on the attached subnet via the egress PE router's IRB interface.
6. Because the ingress PE router is performing Layer 3 routing, the IP TTL is decremented.

RELATED DOCUMENTATION

[EVPN Multihoming Overview | 96](#)

[Understanding VXLANs | 414](#)

[Example: Configuring EVPN with IRB Solution | 974](#)

An EVPN with IRB Solution on EX9200 Switches Overview

IN THIS SECTION

- [Need for an EVPN IRB Solution | 953](#)
- [Implementing the EVPN IRB Solution | 953](#)
- [Benefits of Implementing the EVPN IRB Solution | 955](#)
- [IPv6 Support for IRB Interfaces with EVPN Using Neighborhood Discovery Protocol \(NDP\) | 957](#)

A data center service provider (DCSP) hosts the data center for its multiple customers on a common physical network. To each customer (also called a tenant), the service looks like a full-fledged data center that can expand to 4094 VLANs and all private subnets. For disaster recovery, high availability, and optimization of resource utilization, it is common for the DCSP to span the data center over multiple sites. To deploy data center services, a DCSP faces the following main challenges:

- Extending Layer 2 domains across more than one data center site. This requires optimal intra-subnet traffic forwarding.
- Supporting optimal inter-subnet traffic forwarding and optimal routing in the event of virtual machine (VM) motion.
- Supporting multiple tenants with independent VLAN and subnet space.

Ethernet VPN (EVPN) is targeted to handle all the preceding challenges, wherein:

- The basic EVPN functionality enables optimal intra-subnet traffic forwarding
- Implementing the integrated routing and bridging (IRB) solution in an EVPN deployment enables optimal inter-subnet traffic forwarding
- Configuring EVPN with virtual switch support enables multiple tenants with independent VLAN and subnet space

The following sections describe the integrated routing and bridging (IRB) solution for EVPNs:

Need for an EVPN IRB Solution

EVPN is a technology used to provide Layer 2 extension and interconnection across an IP/MPLS core network to different physical sites belonging to a single Layer 2 domain. In a data center environment with EVPN, there is a need for both Layer 2 (intra-subnet traffic) and Layer 3 (inter-subnet traffic) forwarding and potentially interoperability with tenant Layer 3 VPNs.

With only a Layer 2 solution, there is no optimum forwarding of inter-subnet traffic, even when the traffic is local, for instance, when both the subnets are on the same server.

With only a Layer 3 solution, the following issues for intra-subnet traffic can arise:

- MAC address aliasing issue where duplicate MAC addresses are not detected.
- TTL issue for applications that use TTL 1 to confine traffic within a subnet.
- IPv6 link-local addressing and duplicate address detection that relies on Layer 2 connectivity.
- Layer 3 forwarding does not support the forwarding semantics of a subnet broadcast.
- Support of non-IP applications that require Layer 2 forwarding.

Because of the above mentioned shortcomings of a pure Layer 2 and Layer 3 solution, there is a need for a solution incorporating optimal forwarding of both Layer 2 and Layer 3 traffic in the data center environment when faced with operational considerations such as Layer 3 VPN interoperability and virtual machine (VM) mobility.

An EVPN-based integrated routing and bridging (IRB) solution provides optimum unicast and multicast forwarding for both intra-subnets and inter-subnets within and across data centers.

The EVPN IRB feature is useful for service providers operating in an IP/MPLS network that provides both Layer 2 VPN or VPLS services and Layer 3 VPN services who want to extend their service to provide cloud computation and storage services to their existing customers.

Implementing the EVPN IRB Solution

An EVPN IRB solution provides the following:

- Optimal forwarding for intra-subnet (Layer 2) traffic.
- Optimal forwarding for inter-subnet (Layer 3) traffic.
- Support for ingress replication for multicast traffic.
- Support for network-based as well as host-based overlay models.
- Support for consistent policy-based forwarding for both Layer 2 and Layer 3 traffic.

Junos OS supports several models of EVPN configuration to satisfy the individual needs of EVPN and data center cloud services customers. To provide flexibility and scalability, multiple VLANs can be defined within a particular EVPN instance. Likewise, one or more EVPN instances can be associated with a single Layer 3 VPN virtual routing and forwarding (VRF). In general, each data center tenant is assigned a unique Layer 3 VPN VRF, while a tenant could comprise one or more EVPN instances and one or more VLANs per EVPN instance. To support this model, each configured VLAN (including the default VLAN for an EVPN instance) requires an IRB interface to perform the Layer 2 and Layer 3 functions. Each VLAN or IRB interface maps to a unique IP subnet in the VRF.

There are two major functions that are supported for IRB in EVPN.

- Host MAC-IP synchronization

This includes:

- Advertising the IP address along with the MAC advertisement route in EVPN. This is done by using the IP field in the EVPN MAC advertisement route.
- The receiving PE router installs MAC into the EVPN instance (EVI) table and installs IP into the associated VRF.

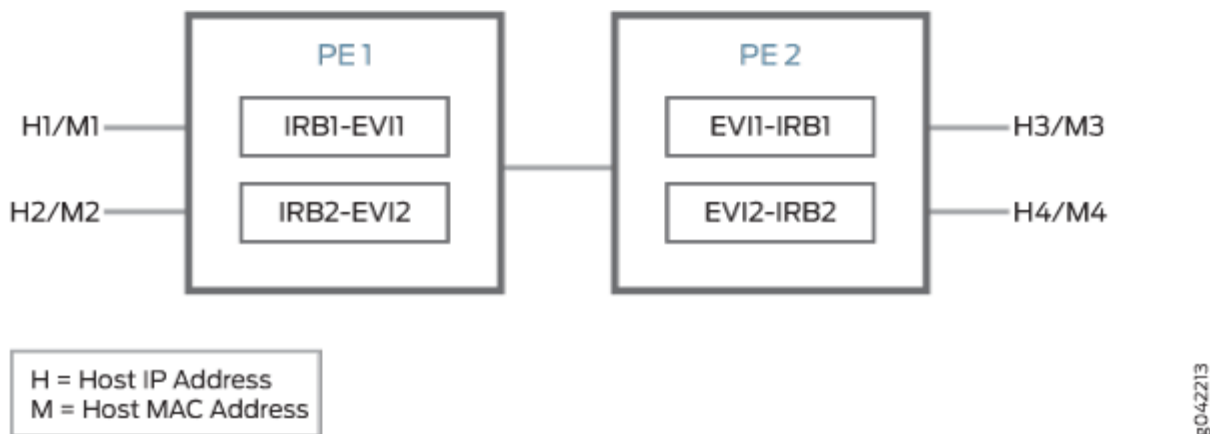
- Gateway MAC-IP synchronization

This includes:

- Advertising all local IRB MAC and IP addresses in an EVPN. This is achieved by including the default gateway extended community in the EVPN MAC advertisement route.
- The receiving PE creates a forwarding state to route packets destined for the gateway MAC, and a proxy ARP is done for the gateway IP with the MAC advertised in the route.

Figure 100 on page 955 illustrates the inter-subnet traffic forwarding between two provider edge (PE) devices—PE1 and PE2. The IRB1 and IRB2 interfaces on each PE device belong to a different subnet, but they share a common VRF.

Figure 100: Inter-Subnet Traffic Forwarding



The inter-subnet traffic forwarding is performed as follows:

1. PE2 advertises the H3-M3 and H4-M4 binding to PE1. Similarly, PE1 advertises the H1-M1 and H2-M2 binding to PE2.
2. PE1 and PE2 install the MAC address in the corresponding EVI MAC table, whereas the IP routes are installed in the shared VRF.
3. The advertising PE device is set as the next hop for the IP routes.
4. If H1 sends packets to H4, the packets are sent to IRB1 on PE1.
5. IP lookup for H4 happens in the shared VRF on PE1. Because the next hop for the H4 IP is PE2 (the advertising PE), an IP unicast packet is sent to PE2.
6. PE1 rewrites the MAC header based on the information in the VRF route, and PE2 performs a MAC lookup to forward the packet to H4.

Benefits of Implementing the EVPN IRB Solution

The main goal of the EVPN IRB solution is to provide optimal Layer 2 and Layer 3 forwarding. The solution is required to efficiently handle inter-subnet forwarding as well as virtual machine (VM) mobility. VM mobility refers to the ability of a VM to migrate from one server to another within the same or a different data center while retaining its existing MAC and IP address. Providing optimal forwarding for inter-subnet traffic and effective VM mobility involves solving two problems – the default gateway problem and the triangular routing problem.

Gateway MAC and IP Synchronization

In an EVPN IRB deployment, the IP default gateway for a VM is the IP address configured on the IRB interface of the provider edge (PE) router corresponding to the VLAN of which the VM is a member. The default gateway problem arises because a VM does not flush its ARP table when relocating from one server to another and continues sending packets with the destination MAC address set to that of the original gateway. If the old and new servers are not part of the same Layer 2 domain (the new Layer 2 domain could be within the current data center or a new data center), the gateway previously identified is no longer the optimal or local gateway. The new gateway needs to identify packets containing the MAC addresses of other gateways on remote PE routers and forward the traffic as if the packets were destined to the local gateway itself. At minimum, this functionality requires each PE router to advertise its gateway or IRB MAC and IP addresses to all other PE routers in the network. The gateway address exchange can be accomplished using the standard MAC route advertisement message (including the IP address parameter) and tagging that route with the default gateway extended community so that the remote PE routers can distinguish the gateway MAC advertisement routes from normal MAC advertisement routes.

Layer 3 VPN Interworking

The inter-data center aspect of the EVPN IRB solution involves routing between VMs that are present in different data centers or routing between a host site completely outside of the data center environment and a VM within a data center. This solution relies on the ability of EVPN MAC route advertisements to carry both MAC address and IP address information. The local MAC learning functionality of the PE router is extended to also capture IP address information associated with MAC addresses learned locally. That IP-MAC address mapping information is then distributed to each PE router through normal EVPN procedures. When a PE router receives such MAC and IP information, it installs the MAC route in the EVPN instance as well as a host route for the associated IP address in the Layer 3 VPN VRF corresponding to that EVPN instance. When a VM moves from one data center to another, normal EVPN procedures result in the MAC and IP address being advertised from the new PE router which the VM resides behind. The host route installed in the VRF associated with an EVPN solicits Layer 3 traffic destined to that VM to the new PE router and avoids triangular routing between the source, the former PE router the VM resided behind, and the new PE router.

BGP scalability is a potential concern with the inter-data center triangular routing avoidance solution because of the potential for injection of many host routes into Layer 3 VPN. With the method previously described, in the worst case there is an IP host route for each MAC address learned through the local EVPN MAC learning procedures or through a MAC advertisement message received from a remote PE router. BGP route target filtering can be used to limit distribution of such routes.

The following functional elements are required to implement the inter-data center triangular routing avoidance using Layer 3 inter-subnet forwarding procedures:

1. The source host sends an IP packet using its own source MAC and IP address with the destination MAC of the IRB interface of the local PE router and the IP address of the destination host.

2. When the IRB interface receives the frame with its MAC as the destination, it performs a Layer 3 lookup in the VRF associated with the EVPN instance to determine where to route the packet.
3. In the VRF, the PE router finds the Layer 3 route derived from a MAC plus an IP EVPN route received from the remote PE router earlier. The destination MAC address is then changed to the destination MAC address corresponding to the destination IP.
4. The packet is then forwarded to the remote PE router serving the destination host using MPLS, using the label corresponding to the EVPN instance of which the destination host is a member.
5. The egress PE router receiving the packet performs a Layer 2 lookup for the destination host's MAC and sends the packet to the destination host on the attached subnet via the egress PE router's IRB interface.
6. Because the ingress PE router is performing Layer 3 routing, the IP TTL is decremented.

IPv6 Support for IRB Interfaces with EVPN Using Neighborhood Discovery Protocol (NDP)

Starting in Junos OS Release 17.3R1, IPv6 addresses are supported on IRB interfaces with EVPN using NDP. The following capabilities are introduced for IPv6 support with EVPN:

- IPv6 addresses on IRB interfaces in primary routing instances
- Learning IPv6 neighbors from solicited neighbor advertisement (NA) messages
- Neighbor solicitation (NS) and neighbor advertisement (NA) packets on the IRB interfaces are disabled from network core
- Virtual gateway addresses are used as Layer 3 addresses
- Host MAC-IP synchronization for IPv6

You can configure the IPv6 addresses in the IRB interface at the `[edit interfaces irb]` hierarchy level.

Release History Table

Release	Description
17.3R1	Starting in Junos OS Release 17.3R1, IPv6 addresses are supported on IRB interfaces with EVPN

RELATED DOCUMENTATION

- [Example: Configuring an EVPN with IRB Solution on EX9200 Switches | 995](#)
- [EVPN Overview for Switches | 878](#)

Anycast Gateways

IN THIS SECTION

- [Benefits of Anycast Gateways | 959](#)
- [Anycast Gateway Configuration Guidelines | 959](#)
- [Anycast Gateway Configuration Limitations | 961](#)

In an EVPN-MPLS or MC-LAG environment with two Juniper Networks devices multihomed in all-active mode, you can configure IRB interfaces on the devices. With the IRB interfaces in place, the multihomed devices function as gateways that handle inter-subnet routing. To set up an IRB interface on a Juniper Networks device, you can configure the following:

- An IRB interface with:
 - An IPv4 or an IPv6 address

```
set interface irb unit logical-unit-number family inet ipv4-address
set interface irb unit logical-unit-number family inet6 ipv6-address
```

- A media access control (MAC) address

```
set interface irb unit logical-unit-number mac mac-address
```

NOTE: In addition to explicitly configuring a MAC address using the above command syntax, you can use the MAC address that the Juniper Networks device automatically generates (chassis MAC).

- A virtual gateway address (VGA) with:
 - An IPv4 or an IPv6 address

```
set interfaces irb unit logical-unit-number family inet address primary-ipv4-address/8
virtual-gateway-address gateway-ipv4-address
```

```
set interfaces irb unit logical-unit-number family inet6 address primary-ipv6-address/104
virtual-gateway-address gateway-ipv6-address
```

- A MAC address

```
set interface irb unit logical-unit-number virtual-gateway-v4-mac mac-address
set interface irb unit logical-unit-number virtual-gateway-v6-mac mac-address
```

NOTE: In addition to explicitly configuring a MAC address using the above command syntax, you can use the MAC address that the Juniper Networks device automatically generates (chassis MAC).

When specifying an IP or MAC address for an IRB interface or VGA on the multihomed devices, you can now use an anycast address. This support of anycast addresses enables you to configure the same addresses for the IRB interface or VGA on each of the multihomed devices, thereby establishing the devices as anycast gateways.

Your IP address subnet scheme will determine whether you use the IRB interface command syntax or the VGA command syntax to set up your anycast gateway.

In an Ethernet VPN–Multiprotocol Label Switching (EVPN–MPLS) or multichassis link aggregation (MC–LAG) environment, you can configure two Juniper Networks devices multihomed in all-active mode as anycast gateways.

The following sections provide more information about anycast gateways.

Benefits of Anycast Gateways

- With the two multihomed Juniper Networks devices acting as anycast gateways in an EVPN–MPLS or MC–LAG network, a host in the same network that generates Layer 3 packets with destinations in other networks can now send the packets to the local anycast gateway. Upon receipt of these Layer 3 packets, the anycast gateway routes the packets in the core network based on destination IP lookup.

Anycast Gateway Configuration Guidelines

- In general, when configuring addresses for an anycast gateway:
 - For IPv4 or IPv6 addresses, you can specify any subnet.

- For MAC addresses, you can use the MAC address that the Juniper Networks device automatically generates (chassis MAC), or you can explicitly configure a MAC address using the CLI.
- Your IP address subnet scheme will determine whether you use the IRB interface command syntax or the VGA command syntax to set up your anycast gateway.

To set up your multihomed devices as anycast gateways, we provide the following configuration guidelines:

- Guideline 1—If the IP address for the anycast gateways is in the /30 (for IPv4) or /126 (for IPv6) subnet:
 - You must configure the same IP address for the IRB interface on each of the multihomed devices using one of the following commands.

```
set interface irb unit logical-unit-number family inet ipv4-address
set interface irb unit logical-unit-number family inet6 ipv6-address
```

- You must explicitly configure the MAC address using the following command:

```
set interface irb unit logical-unit-number mac mac-address
```

- You must not configure a VGA (IP and MAC addresses).
- Guideline 2—If the IP address for the anycast gateways is in the /31 (for IPv4) or /127 (for IPv6) subnet:
 - You must configure the same IP address for the IRB interface on each of the multihomed devices using one of the following commands.

```
set interface irb unit logical-unit-number family inet ipv4-address
set interface irb unit logical-unit-number family inet6 ipv6-address
```

- You must explicitly configure the MAC address using the following command:

```
set interface irb unit logical-unit-number mac mac-address
```

- You must not configure a VGA (IP and MAC addresses).

- We do not recommend configuring the Virtual Router Redundancy Protocol (VRRP) protocol on the IRB interface.
- Guideline 3—If the IP address for the anycast gateways is a subnet other than the ones described in the previous bullets:
 - You must configure the same IP address for the VGA on each of the multihomed devices using one of the following commands.

```
set interfaces irb unit logical-unit-number family inet address primary-ipv4-address/8
virtual-gateway-address gateway-ipv4-address
set interfaces irb unit logical-unit-number family inet6 address primary-ipv6-address/104
virtual-gateway-address gateway-ipv6-address
```

- You must explicitly configure the MAC address using one of the following commands:

```
set interface irb unit logical-unit-number virtual-gateway-v4-mac mac-address
set interface irb unit logical-unit-number virtual-gateway-v6-mac mac-address
```

- When specifying a MAC address for the VGA, we do not recommend using the same MAC address used for VRRP.

Anycast Gateway Configuration Limitations

When configuring the anycast gateway using guidelines described earlier in this topic, keep the following in mind:

- In general, we do not recommend reusing a VRRP MAC address as a MAC address for an IRB interface. However, if you must do so, as is the general practice when configuring VRRP on Juniper Networks devices, you must use a VRRP IPv4 MAC address for the IPv4 family and a VRRP IPv6 MAC address for the IPv6 family.

Given these parameters, the only configuration guideline with which this limitation will work is configuration guideline 3.

- When configuring anycast gateway addresses using guidelines 1 and 2 in an EVPN-MPLS environment, you must also specify the `default-gateway do-not-advertise` configuration statements within a routing instance. For example:

```
set routing-instance routing-instance-name protocols evpn default-gateway do-not-advertise
```

- In an EVPN-MPLS environment, if your anycast gateway IP addresses are in different subnets and you specify the addresses within multiple routing instances:
- If you configured an anycast gateway IP address using configuration guidelines 1 or 2 in one routing instance, and another anycast gateway IP address using configuration guideline 3 in a different routing instance, you must also specify the `default-gateway no-gateway-community` configuration statements within the routing instance:

```
set routing-instance routing-instance-name protocols evpn default-gateway no-gateway-community
```

This additional configuration applies only to the routing instance that includes anycast gateway IP addresses configuring using guidelines 1 or 2.

- For each routing instance in which you specified the anycast gateway IP address using configuration guidelines 1 and 2, we recommend specifying a single non-VRRP MAC address.

Configuring EVPN with IRB Solution

You can configure an Ethernet VPN (EVPN) with IRB solution to enable Layer 2 switching and Layer 3 routing operations within a single node, thus avoiding extra hops for inter-subnet traffic. The EVPN IRB solution eliminates the default gateway problem using the gateway MAC and IP synchronization, and avoids the triangular routing problem with Layer 3 interworking by creating IP host routes for virtual machines (VMs) in the tenant virtual routing and forwarding (VRF) routing instances.

Before you begin:

1. Configure the router interfaces.
2. Configure the router ID and autonomous system number for the device.
3. Enable chained composite next hop for EVPN.
4. Configure OSPF or any other IGP protocol.
5. Configure a BGP internal group.
6. Include the EVPN signaling network layer reachability information (NLRI) to the internal BGP group.
7. Configure RSVP or LDP.
8. Configure MPLS.
9. Create a label-switched path between the provider edge (PE) devices.

To configure the PE device:

1. Configure the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance instance-type evpn
```

2. Set the VLAN identifier for the bridging domain in the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance vlan-id VLAN-ID
```

3. Configure the interface name for the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance interface CE-facing-interface
```

4. Configure the IRB interface as the routing interface for the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance routing-interface irb.0
```

5. Configure the route distinguisher for the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance route-distinguisher route-distinguisher-value
```

6. Configure the VPN routing and forwarding (VRF) target community for the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance vrf-target vrf-target-value
```

7. Assign the interface name that connects the PE device site to the VPN.

```
[edit routing-instances]
user@PE1# set evpn-instance protocols evpn interface CE-facing-interface
```

8. Configure the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf-instance instance-type vrf
```

9. Configure the IRB interface as the routing interface for the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf-instance interface irb.0
```

10. Configure the route distinguisher for the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf-instance route-distinguisher route-distinguisher-value
```

11. Configure the VRF label for the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf-instance vrf-table-label
```

12. Verify and commit the configuration.

For example:

```
[edit routing-instances]
user@PE1# set evpna instance-type evpn
user@PE1# set evpna vlan-id 10
user@PE1# set evpna interface ge-1/1/8.0
user@PE1# set evpna routing-interface irb.0
user@PE1# set evpna route-distinguisher 192.0.2.1:100
user@PE1# set evpna vrf-target target:100:100
user@PE1# set evpna protocols evpn interface ge-1/1/8.0
user@PE1# set vrf instance-type vrf
user@PE1# set vrf interface irb.0
user@PE1# set vrf route-distinguisher 192.0.2.1:300
```

```
user@PE1# set vrf vrf-target target:100:300
user@PE1# set vrf vrf-table-label
```

```
[edit]
user@PE1# commit
commit complete
```

RELATED DOCUMENTATION

[Example: Configuring EVPN with IRB Solution | 968](#)

Configuring an EVPN with IRB Solution on EX9200 Switches

You can configure an Ethernet VPN (EVPN) with IRB solution to enable Layer 2 switching and Layer 3 routing operations within a single node, thus avoiding extra hops for inter-subnet traffic. The EVPN IRB solution eliminates the default gateway problem using the gateway MAC and IP synchronization, and avoids the triangular routing problem with Layer 3 interworking by creating IP host routes for virtual machines (VMs) in the tenant virtual routing and forwarding (VRF) routing instances.

Before you begin:

1. Configure the switch interfaces.
2. Configure the router ID and autonomous system number for the device.
3. Enable the chained composite next hop for EVPN.
4. Configure OSPF or any other IGP protocol.
5. Configure a BGP internal group.
6. Include the EVPN signaling network layer reachability information (NLRI) to the internal BGP group.
7. Configure LDP.
8. Configure MPLS.

To configure the PE device:

1. Configure the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance instance-type evpn
```

2. Set the VLAN identifier for the bridging domain in the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance vlan-id VLAN-ID
```

3. Configure the interface name for the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance interface CE-facing-interface
```

4. Configure the IRB interface as the routing interface for the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance l3-interface irb.0
```

5. Configure the route distinguisher for the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance route-distinguisher route-distinguisher-value
```

6. Configure the VPN routing and forwarding (VRF) target community for the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance vrf-target vrf-target-value
```

7. Assign the interface name that connects the PE device site to the VPN.

```
[edit routing-instances]
user@PE1# set evpn-instance protocols evpn interface CE-facing-interface
```

8. Configure the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf-instance instance-type vrf
```

9. Configure the IRB interface as the routing interface for the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf-instance interface irb.0
```

10. Configure the route distinguisher for the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf-instance route-distinguisher route-distinguisher-value
```

11. Configure the VRF label for the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf-instance vrf-table-label
```

12. Verify and commit the configuration.

For example:

```
[edit routing-instances]
user@PE1# set evpna instance-type evpn
user@PE1# set evpna vlan-id 10
user@PE1# set evpna interface ge-1/1/8.0
user@PE1# set evpna l3-interface irb.0
user@PE1# set evpna route-distinguisher 100.255.0.1:100
user@PE1# set evpna vrf-target target:100:100
user@PE1# set evpna protocols evpn interface ge-1/1/8.0
user@PE1# set vrf instance-type vrf
user@PE1# set vrf interface irb.0
user@PE1# set vrf route-distinguisher 100.255.0.1:300
```

```
user@PE1# set vrf vrf-target target:100:300
user@PE1# set vrf vrf-table-label
```

```
[edit]
user@PE1# commit
commit complete
```

RELATED DOCUMENTATION

[Example: Configuring an EVPN with IRB Solution on EX9200 Switches | 995](#)

Example: Configuring EVPN with IRB Solution

IN THIS SECTION

- [EVPN with IRB Solution Overview | 968](#)
- [Example: Configuring EVPN with IRB Solution | 974](#)

EVPN with IRB Solution Overview

IN THIS SECTION

- [Need for an EVPN IRB Solution | 969](#)
- [Implementing the EVPN IRB Solution | 970](#)
- [Benefits of Implementing the EVPN IRB Solution | 972](#)

A Data Center Service Provider (DCSP) hosts the data center for its multiple customers on a common physical network. To each customer (also called a tenant), the service looks like a full-fledged data center that can expand to 4094 VLANs and all private subnets. For disaster recovery, high availability, and

optimization of resource utilization, it is common for the DCSP to span across the data center to more than one site. To deploy the data center services, a DCSP faces the following main challenges:

- Extending Layer 2 domains across more than one data center site. This requires optimal intra-subnet traffic forwarding.
- Supporting optimal inter-subnet traffic forwarding and optimal routing in the event of virtual machine (VM).
- Supporting multiple tenants with independent VLAN and subnet space.

Ethernet VPN (EVPN) is targeted to handle all of the above mentioned challenges, wherein:

- The basic EVPN functionality enables optimal intra-subnet traffic forwarding
- Implementing the integrated routing and bridging (IRB) solution in an EVPN deployment enables optimal inter-subnet traffic forwarding
- Configuring EVPN with virtual switch support enables multiple tenants with independent VLAN and subnet space

The following sections describe the IRB solution for EVPNs:

Need for an EVPN IRB Solution

EVPN is a technology used to provide Layer 2 extension and interconnection across an IP/MPLS core network to different physical sites belonging to a single Layer 2 domain. In a data center environment with EVPN, there is a need for both Layer 2 (intra-subnet traffic) and Layer 3 (inter-subnet traffic) forwarding and potentially interoperation with tenant Layer 3 VPNs.

With only a Layer 2 solution, there is no optimum forwarding of inter-subnet traffic, even when the traffic is local, for instance, when both the subnets are on the same server.

With only a Layer 3 solution, the following issues for intra-subnet traffic can arise:

- MAC address aliasing issue where duplicate MAC addresses are not detected.
- TTL issue for applications that use TTL 1 to confine traffic within a subnet.
- IPv6 link-local addressing and duplicate address detection that relies on Layer 2 connectivity.
- Layer 3 forwarding does not support the forwarding semantics of a subnet broadcast.
- Support of non-IP applications that require Layer 2 forwarding.

Because of the above mentioned shortcomings of a pure Layer 2 and Layer 3 solution, there is a need for a solution incorporating optimal forwarding of both Layer 2 and Layer 3 traffic in the data center

environment when faced with operational considerations such as Layer 3 VPN interoperability and virtual machine (VM) mobility.

An EVPN-based integrated routing and bridging (IRB) solution provides optimum unicast and multicast forwarding for both intra-subnets and inter-subnets within and across data centers.

The EVPN IRB feature is useful for service providers operating in an IP/MPLS network that provides both Layer 2 VPN or VPLS services and Layer 3 VPN services who want to extend their service to provide cloud computation and storage services to their existing customers.

Implementing the EVPN IRB Solution

An EVPN IRB solution provides the following:

- Optimal forwarding for intra-subnet (Layer 2) traffic.
- Optimal forwarding for inter-subnet (Layer 3) traffic.
- Support for ingress replication for multicast traffic.
- Support for network-based as well as host-based overlay models.
- Support for consistent policy-based forwarding for both Layer 2 and Layer 3 traffic.
- Support for the following routing protocols on the IRB interface:
 - BFD
 - BGP
 - IS-IS
 - OSPF and OSPF version 3
- Support for single-active and all-active multihoming

Junos OS supports several models of EVPN configuration to satisfy the individual needs of EVPN and data center cloud services customers. To provide flexibility and scalability, multiple bridge domains can be defined within a particular EVPN instance. Likewise, one or more EVPN instances can be associated with a single Layer 3 VPN virtual routing and forwarding (VRF). In general, each data center tenant is assigned a unique Layer 3 VPN VRF, while a tenant could comprise one or more EVPN instances and one or more bridge domains per EVPN instance. To support this model, each configured bridge domain (including the default bridge domain for an EVPN instance) requires an IRB interface to perform the Layer 2 and Layer 3 functions. Each bridge domain or IRB interface maps to a unique IP subnet in the VRF.

NOTE: You can associate an IRB interface with the primary instance inet.0 table instead of a VRF in an EVPN IRB solution.

There are two major functions that are supported for IRB in EVPN.

- Host MAC-IP synchronization

This includes:

- Advertising the IP address along with the MAC advertisement route in EVPN. This is done by using the IP field in the EVPN MAC advertisement route.
- The receiving PE router installs MAC into the EVPN instance (EVI) table and installs IP into the associated VRF.

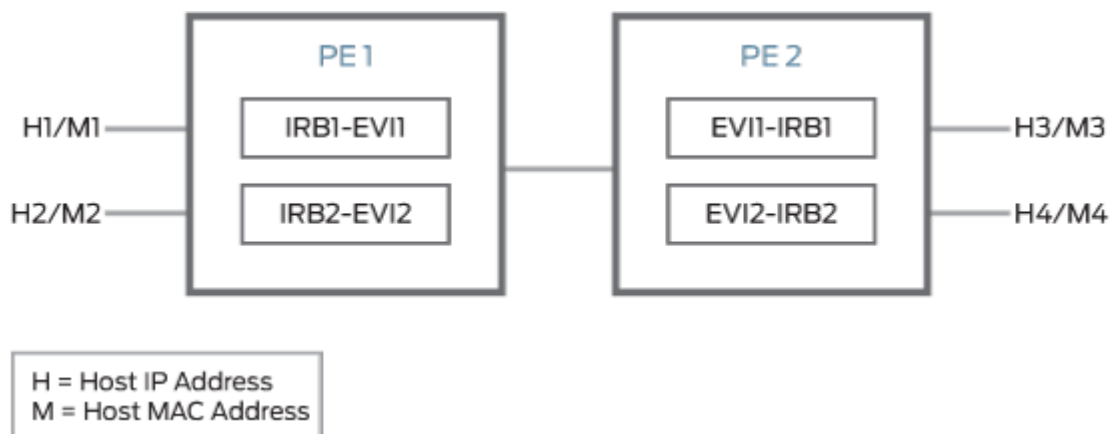
- Gateway MAC-IP synchronization

This includes:

- Advertising all local IRB MAC and IP addresses in an EVPN. This is achieved by including the default gateway extended community in the EVPN MAC advertisement route.
- The receiving PE creates a forwarding state to route packets destined for the gateway MAC, and a proxy ARP is done for the gateway IP with the MAC advertised in the route.

Figure 101 on page 971 illustrates the inter-subnet traffic forwarding between two provider edge (PE) devices – PE1 and PE2. The IRB1 and IRB2 interfaces on each PE device belong to a different subnet, but they share a common VRF.

Figure 101: Inter-Subnet Traffic Forwarding



The inter-subnet traffic forwarding is performed as follows:

1. PE2 advertises H3-M3 and H4-M4 binding to PE1. Similarly PE1 advertises H1-M1 and H2-M2 binding to PE2.
2. PE1 and PE2 install the MAC address in the corresponding EVI MAC table, whereas the IP routes are installed in the shared VRF.
3. The advertising PE device is set as the next hop for the IP routes.
4. If H1 sends packets to H4, the packets are sent to IRB1 on PE1.
5. IP lookup for H4 happens in the shared VRF on PE1. Because the next hop for the H4 IP is PE2 (the advertising PE), an IP unicast packet is sent to PE2.
6. PE1 rewrites the MAC header based on the information in the VRF route, and PE2 performs a MAC lookup to forward the packet to H4.

Benefits of Implementing the EVPN IRB Solution

The main goal of the EVPN IRB solution is to provide optimal Layer 2 and Layer 3 forwarding. The solution is required to efficiently handle inter-subnet forwarding as well as virtual machine (VM) mobility. VM mobility refers to the ability of a VM to migrate from one server to another within the same or a different data center while retaining its existing MAC and IP address. Providing optimal forwarding for inter-subnet traffic and effective VM mobility involves solving two problems – the default gateway problem and the triangular routing problem.

Starting in Junos OS Release 17.1R1, IPv6 addresses are supported on IRB interfaces with EVPN using the Neighbor Discovery Protocol (NDP). The following capabilities are introduced for IPv6 support with EVPN:

- IPv6 addresses on IRB interfaces in primary routing instances
- Learning IPv6 neighborhood from solicited NA message
- NS and NA packets on the IRB interfaces are disabled from network core
- Virtual gateway addresses are used as Layer 3 addresses
- Host MAC-IP synchronization for IPv6

You can configure the IPv6 addresses in the IRB interface at the `[edit interfaces irb]` hierarchy level.

Gateway MAC and IP Synchronization

In an EVPN IRB deployment, the IP default gateway for a VM is the IP address configured on the IRB interface of the provider edge (PE) router corresponding to the bridge domain or VLAN of which the VM

is a member. The default gateway problem arises because a VM does not flush its ARP table when relocating from one server to another and continues sending packets with the destination MAC address set to that of the original gateway. If the old and new servers are not part of the same Layer 2 domain (the new Layer 2 domain could be within the current data center or a new data center), the gateway previously identified is no longer the optimal or local gateway. The new gateway needs to identify packets containing the MAC addresses of other gateways on remote PE routers and forward the traffic as if the packets were destined to the local gateway itself. At the minimum, this functionality requires each PE router to advertise its gateway or IRB MAC and IP addresses to all other PE routers in the network. The gateway address exchange can be accomplished using the standard MAC route advertisement message (including the IP address parameter) and tagging that route with the default gateway extended community so that the remote PE routers can distinguish the gateway MAC advertisement routes from normal MAC advertisement routes.

Layer 3 VPN Interworking

The inter-data center aspect of the EVPN IRB solution involves routing between VMs that are present in different data centers or routing between a host site completely outside of the data center environment and a VM within a data center. This solution relies on the ability of EVPN MAC route advertisements to carry both MAC address and IP address information. The local MAC learning functionality of the PE router is extended to also capture IP address information associated with MAC addresses learned locally. That IP-MAC address mapping information is then distributed to each PE router through normal EVPN procedures. When a PE router receives such MAC and IP information, it installs the MAC route in the EVPN instance as well as a host route for the associated IP address in the Layer 3 VPN VRF corresponding to that EVPN instance. When a VM moves from one data center to another, normal EVPN procedures result in the MAC and IP address being advertised from the new PE router which the VM resides behind. The host route installed in the VRF associated with an EVPN solicits Layer 3 traffic destined to that VM to the new PE router and avoids triangular routing between the source, the former PE router the VM resided behind, and the new PE router.

BGP scalability is a potential concern with the inter-data center triangular routing avoidance solution because of the potential for injection of many host routes into Layer 3 VPN. With the method previously described, in the worst case there is an IP host route for each MAC address learned through the local EVPN MAC learning procedures or through a MAC advertisement message received from a remote PE router. BGP route target filtering can be used to limit distribution of such routes.

The following functional elements are required to implement the inter-data center triangular routing avoidance using Layer 3 inter-subnet forwarding procedures:

1. The source host sends an IP packet using its own source MAC and IP address with the destination MAC of the IRB interface of the local PE router and the IP address of the destination host.
2. When the IRB interface receives the frame with its MAC as the destination, it performs a Layer 3 lookup in the VRF associated with the EVPN instance to determine where to route the packet.

3. In the VRF, the PE router finds the Layer 3 route derived from a MAC plus an IP EVPN route received from the remote PE router earlier. The destination MAC address is then changed to the destination MAC address corresponding to the destination IP.
4. The packet is then forwarded to the remote PE router serving the destination host using MPLS, using the label corresponding to the EVPN instance of which the destination host is a member.
5. The egress PE router receiving the packet performs a Layer 2 lookup for the destination host's MAC and sends the packet to the destination host on the attached subnet via the egress PE router's IRB interface.
6. Because the ingress PE router is performing Layer 3 routing, the IP TTL is decremented.

SEE ALSO

[EVPN Multihoming Overview | 96](#)

[Understanding VXLANs | 414](#)

[Example: Configuring EVPN with IRB Solution | 974](#)

Example: Configuring EVPN with IRB Solution

IN THIS SECTION

- [Requirements | 974](#)
- [Overview | 975](#)
- [Configuration | 976](#)
- [Verification | 985](#)

This example shows how to configure an integrated routing and bridging (IRB) solution in an Ethernet VPN (EVPN) deployment.

Requirements

This example uses the following hardware and software components:

- Two MX Series 5G Universal Routing Platforms containing MPC FPCs configured as PE routers.
- Two customer edge (CE) routers, each connected to the PE routers.

- Junos OS Release 14.1 or later running on all the PE routers.

Before you begin:

1. Configure the router interfaces.
2. Configure OSPF or any other IGP protocol.
3. Configure BGP.
4. Configure RSVP or LDP.
5. Configure MPLS.

Overview

IN THIS SECTION

- [Topology | 976](#)

In an EVPN solution, multiple bridge domains can be defined within a particular EVPN instance, and one or more EVPN instances can be associated with a single Layer 3 VPN VRF. In general, each data center tenant is assigned a unique Layer 3 VPN virtual route forwarding (VRF), although the tenant can be comprised of one or more EVPN instances or bridge domains per EVPN instance.

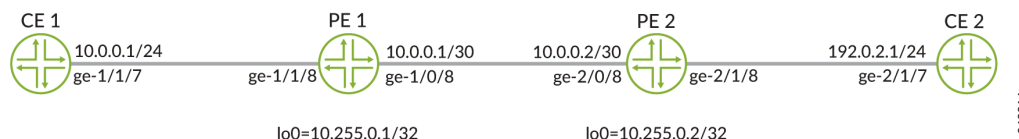
To support this flexibility and scalability factor, the EVPN solution provides support for the IRB interfaces on MX Series routers containing MPC FPCs to facilitate optimal Layer 2 and Layer 3 forwarding along with virtual machine mobility. The IRB interfaces are configured on each configured bridge domain including the default bridge domain for an EVPN instance.

IRB is the ability to do Layer 2 switching and Layer 3 routing within a single node, thus avoiding extra hops for inter-subnet traffic. The EVPN IRB solution eliminates the default gateway problem using the gateway MAC and IP synchronization, and avoids the triangular routing problem with Layer 3 interworking by creating IP host routes for virtual machines (VMs) in the tenant VRFs.

Topology

Figure 102 on page 976 illustrates a simple EVPN topology with IRB solution. Routers PE1 and PE2 are the provider edge routers that connect to two customer edge (CE) routers each – CE1 and CE2.

Figure 102: EVPN with IRB Solution



Configuration

IN THIS SECTION

- Procedure | 976
- Results | 983

Procedure

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

CE1

```
set interfaces ge-1/1/7 vlan-tagging
set interfaces ge-1/1/7 unit 0 vlan-id 10
set interfaces ge-1/1/7 unit 0 family inet address 10.0.0.1/24
set routing-options static route 192.0.2.0/24 next-hop 10.0.0.251
```

PE1

```

set interfaces ge-1/0/8 unit 0 family inet address 10.0.0.1/30
set interfaces ge-1/0/8 unit 0 family mpls
set interfaces ge-1/1/8 flexible-vlan-tagging
set interfaces ge-1/1/8 encapsulation flexible-ethernet-services
set interfaces ge-1/1/8 unit 0 encapsulation vlan-bridge
set interfaces ge-1/1/8 unit 0 vlan-id 10
set interfaces irb unit 0 family inet address 10.0.0.251/24
set interfaces lo0 unit 0 family inet address 10.255.0.1/32
set routing-options router-id 10.255.0.1
set routing-options autonomous-system 100
set routing-options forwarding-table chained-composite-next-hop ingress evpn
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path PE1-to-PE2 from 10.255.0.1
set protocols mpls label-switched-path PE1-to-PE2 to 10.255.0.2
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 10.255.0.1
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 10.255.0.2
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set routing-instances evpna instance-type evpn
set routing-instances evpna vlan-id 10
set routing-instances evpna interface ge-1/1/8.0
set routing-instances evpna routing-interface irb.0
set routing-instances evpna route-distinguisher 10.255.0.1:100
set routing-instances evpna vrf-target target:100:100
set routing-instances evpna protocols evpn interface ge-1/1/8.0
set routing-instances vrf instance-type vrf
set routing-instances vrf interface irb.0
set routing-instances vrf route-distinguisher 10.255.0.1:300
set routing-instances vrf vrf-target target:100:300
set routing-instances vrf vrf-table-label

```

PE2

```

set interfaces ge-2/0/8 unit 0 family inet address 10.0.0.2/30
set interfaces ge-2/0/8 unit 0 family mpls
set interfaces ge-2/1/8 flexible-vlan-tagging
set interfaces ge-2/1/8 encapsulation flexible-ethernet-services
set interfaces ge-2/1/8 unit 0 encapsulation vlan-bridge
set interfaces ge-2/1/8 unit 0 vlan-id 20
set interfaces irb unit 0 family inet address 192.0.2.1/24
set interfaces lo0 unit 0 family inet address 10.255.0.2/32
set routing-options router-id 10.255.0.2
set routing-options autonomous-system 100
set routing-options forwarding-table chained-composite-next-hop ingress evpn
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path PE2-to-PE1 from 10.255.0.2
set protocols mpls label-switched-path PE2-to-PE1 to 10.255.0.1
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 10.255.0.2
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 10.255.0.1
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf traffic-engineering
set routing-instances evpna instance-type evpn
set routing-instances evpna vlan-id 20
set routing-instances evpna interface ge-2/1/8.0
set routing-instances evpna routing-interface irb.0
set routing-instances evpna route-distinguisher 10.255.0.2:100
set routing-instances evpna vrf-target target:100:100
set routing-instances evpna protocols evpn interface ge-2/1/8.0
set routing-instances vrf instance-type vrf
set routing-instances vrf interface irb.0
set routing-instances vrf route-distinguisher 10.255.0.2:300
set routing-instances vrf vrf-target target:100:300
set routing-instances vrf vrf-table-label

```

CE2

```

set interfaces ge-2/1/7 vlan-tagging
set interfaces ge-2/1/7 unit 0 vlan-id 20
set interfaces ge-2/1/7 unit 0 family inet address 192.0.2.1/24
set routing-options static route 10.0.0.0/24 next-hop 192.0.2.5

```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure Router PE1:

NOTE: Repeat this procedure for Router PE2, after modifying the appropriate interface names, addresses, and other parameters.

1. Configure Router PE1 interfaces.

```

[edit interfaces]
user@PE1# set ge-1/0/8 unit 0 family inet address 10.0.0.1/30
user@PE1# set ge-1/0/8 unit 0 family mpls
user@PE1# set ge-1/1/8 flexible-vlan-tagging
user@PE1# set ge-1/1/8 encapsulation flexible-ethernet-services
user@PE1# set ge-1/1/8 unit 0 encapsulation vlan-bridge
user@PE1# set ge-1/1/8 unit 0 vlan-id 10
user@PE1# set irb unit 0 family inet address 10.0.0.251/24
user@PE1# set lo0 unit 0 family inet address 10.255.0.1/32

```

2. Set the router ID and autonomous system number for Router PE1.

```

[edit routing-options]
user@PE1# set router-id 10.255.0.1
user@PE1# set autonomous-system 100

```

3. Configure the chained composite next hop for EVPN.

```
[edit routing-options]
user@PE1# set forwarding-table chained-composite-next-hop ingress evpn
```

4. Enable RSVP on all the interfaces of Router PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set rsvp interface all
user@PE1# set rsvp interface fxp0.0 disable
```

5. Create label-switched path for Router PE1 to reach Router PE2.

```
[edit protocols]
user@PE1# set mpls label-switched-path PE1-to-PE2 from 10.255.0.1
user@PE1# set mpls label-switched-path PE1-to-PE2 to 10.255.0.2
```

6. Enable MPLS on all the interfaces of Router PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set mpls interface all
user@PE1# set mpls mpls interface fxp0.0 disable
```

7. Configure the BGP group for Router PE1.

```
[edit protocols]
user@PE1# set bgp group ibgp type internal
```

8. Assign local and neighbor addresses to the ibgp BGP group for Router PE1 to peer with Router PE2.

```
[edit protocols]
user@PE1# set bgp group ibgp local-address 10.255.0.1
user@PE1# set bgp group ibgp neighbor 10.255.0.2
```

9. Include the EVPN signaling Network Layer Reachability Information (NLRI) to the ibgp BGP group.

```
[edit protocols]
user@PE1# set bgp group ibgp family evpn signaling
```

10. Configure OSPF on all the interfaces of Router PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set ospf area 0.0.0.0 interface all
user@PE1# set ospf area 0.0.0.0 interface fxp0.0 disable
```

11. Enable traffic-engineering for OSPF. For RSVP-signaled LSPs with OSPF as the IGP, traffic-engineering must be enabled for the LSPs to come up.

```
[edit protocols]
user@PE1# set ospf traffic-engineering
```

12. Configure the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpna instance-type evpn
```

13. Set the VLAN identifier for the bridging domain in the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna vlan-id 10
```

14. Configure the interface name for the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna interface ge-1/1/8.0
```


15. Configure the IRB interface as the routing interface for the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna routing-interface irb.0
```

16. Configure the route distinguisher for the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna route-distinguisher 10.255.0.1:100
```

17. Configure the VPN routing and forwarding (VRF) target community for the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna vrf-target target:100:100
```

18. Assign the interface name that connects the PE1 site to the VPN.

```
[edit routing-instances]
user@PE1# set evpna protocols evpn interface ge-1/1/8.0
```

19. Configure the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf instance-type vrf
```

20. Configure the IRB interface as the routing interface for the vrf routing instance.

```
[edit routing-instances]
user@PE1# set vrf interface irb.0
```

21. Configure the route distinguisher for the vrf routing instance.

```
[edit routing-instances]
user@PE1# set vrf route-distinguisher 10.255.0.1:300
```

22. Configure the VRF label for the vrf routing instance.

```
[edit routing-instances]
user@PE1# set vrf vrf-table-label
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show routing-options`, `show protocols`, and `show routing-instances` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show interfaces
ge-1/0/8 {
  unit 0 {
    family inet {
      address 10.0.0.1/30;
    }
    family mpls;
  }
}
ge-1/1/8 {
  flexible-vlan-tagging;
  encapsulation flexible-ethernet-services;
  unit 0 {
    encapsulation vlan-bridge;
    vlan-id 10;
  }
}
irb {
  unit 0 {
    family inet {
      address 10.0.0.251/24;
    }
  }
}
lo0 {
  unit 0 {
    family inet {
      address 10.255.0.1/32 {

```

```
    }
}
```

```
user@PE1# show routing-options
router-id 10.255.0.1;
autonomous-system 100;
forwarding-table {
    chained-composite-next-hop {
        ingress {
            evpn;
        }
    }
}
```

```
user@PE1# show protocols
rsvp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
mpls {
    label-switched-path PE1-to-PE2 {
        from 10.255.0.1;
        to 10.255.0.2;
    }
    interface all;
    interface fxp0.0 {
        disable;
    }
}
bgp {
    group ibgp {
        type internal;
        local-address 10.255.0.1;
        family evpn {
            signaling;
        }
        neighbor 10.255.0.2;
    }
}
```

```

}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
}
}

```

```

user@PE1# show routing-instances
evpna {
    instance-type evpn;
    vlan-id 10;
    interface ge-1/1/8.0;
    routing-interface irb.0;
    route-distinguisher 10.255.0.1:100;
    vrf-target target:100:100;
    protocols {
        evpn {
            interface ge-1/1/8.0;
        }
    }
}
vrf {
    instance-type vrf;
    interface irb.0;
    route-distinguisher 10.255.0.1:300;
    vrf-target target:100:300;
    vrf-table-label;
}

```

Verification

IN THIS SECTION

● [Verifying Local IRB MACs | 986](#)

- [Verifying Remote IRB MACs | 987](#)
- [Verifying Local IRB IPs | 989](#)
- [Verifying Remote IRB IPs | 990](#)
- [Verifying CE-CE Inter-Subnet Forwarding | 992](#)
- [Verifying CE-PE Inter-Subnet Forwarding | 993](#)
- [Verifying PE-PE Inter-Subnet Forwarding | 994](#)

Confirm that the configuration is working properly.

Verifying Local IRB MACs

Purpose

Verify that the local IRB MACs are learned from L2ALD.

Action

On Router PE1, determine the MAC address of the local IRB interface.

From operational mode, run the `show interfaces irb extensive | match "Current address"` command.

```
user@PE1> show interfaces irb extensive | match
"Current address"
Current address: a8:d0:e5:54:0d:10, Hardware address: a8:d0:e5:54:0d:10
```

From operational mode, run the `show route table evpna.evpn.0 extensive | find "a8:d0:e5:54:0d:10"` command.

```
user@PE1> show route table evpna.evpn.0 extensive
| find "a8:d0:e5:54:0d:10"
2:10.255.0.1:100::0::100::a8:d0:e5:54:0d:10/384 (1 entry, 1 announced)
TSI:
Page 0 idx 0, (group PE type Internal) Type 1 val 0x2736568 (adv_entry)
  Advertised metrics:
    Flags: Nexthop Change
    Nexthop: Self
    Localpref: 100
    AS path: [100] I
```

```

Communities: target:100:100 evpn-default-gateway
Path 2:10.255.0.1:100::0::100::a8:d0:e5:54:0d:10 Vector len 4. Val: 0
  *EVPN  Preference: 170
    Next hop type: Indirect
    Address: 0x26f8354
    Next-hop reference count: 6
    Protocol next hop: 10.255.0.1
    Indirect next hop: 0x0 - INH Session ID: 0x0
    State: <Active Int Ext>
    Age: 23:29:08
    Validation State: unverified
    Task: evpna-evpn
    Announcement bits (1): 1-BGP_RT_Background
    AS path: I
    Communities: evpn-default-gateway
    Route Label: 299776

```

Meaning

The MAC-only route for the local IRB interface appears in the EVPN instance route table on Router PE1 and is learned from EVPN and tagged with the default gateway extended community.

Verifying Remote IRB MACs

Purpose

Verify that the remote IRB MACs are learned from BGP.

Action

On Router PE1, determine the MAC address of the local IRB interface.

From operational mode, run the `show interfaces irb extensive | match "Current address"` command.

```

user@PE1> show interfaces irb extensive | match
"Current address"
Current address: a8:d0:e5:54:0d:10, Hardware address: a8:d0:e5:54:0d:10

```

On Router PE2, verify that the remote IRB MACs are learned.

Meaning

The MAC-only route for the remote IRB interface appears in the EVPN instance route table on Router PE2 and is learned from BGP and tagged with the default gateway extended community.

Verifying Local IRB IPs

Purpose

Verify that the local IRB IPs are learned locally by RPD.

Action

On Router PE1, determine the MAC and IP addresses of the local IRB interface.

From operational mode, run the `show interfaces irb extensive | match "Current address"` command.

```
user@PE1> show interfaces irb extensive | match
"Current address"
Current address: a8:d0:e5:54:0d:10, Hardware address: a8:d0:e5:54:0d:10
```

From operational mode, run the `show interfaces irb.0 terse | match inet` command.

```
user@PE1> show interfaces irb.0 terse | match
inet
irb.0                up    up    inet    10.0.0.251/24
```

From operational mode, run the `show route table evpna.evpn.0 extensive | find "a8:d0:e5:54:0d:10::10.0.0.251"` command.

```
user@PE2> show route table evpna.evpn.0 extensive
| find "a8:d0:e5:54:0d:10::10.0.0.251"
2:10.255.0.1:100::0::100::a8:d0:e5:54:0d:10::10.0.0.251/384 (1 entry, 1 announced)
TSI:
Page 0 idx 0, (group PE type Internal) Type 1 val 0x27365a0 (adv_entry)
  Advertised metrics:
    Flags: Nexthop Change
    Nexthop: Self
    Localpref: 100
    AS path: [100] I
```



```

Communities: target:100:100 evpn-default-gateway
Path 2:10.255.0.1:100::0::100::a8:d0:e5:54:0d:10::10.0.0.251 Vector len 4. Val: 0
  *EVPN   Preference: 170 <<<<<
        Next hop type: Indirect
        Address: 0x26f8354
        Next-hop reference count: 6
        Protocol next hop: 10.255.0.1
        Indirect next hop: 0x0 - INH Session ID: 0x0
        State: <Active Int Ext>
        Age: 23:48:46
        Validation State: unverified
        Task: evpna-evpn
        Announcement bits (1): 1-BGP_RT_Background
        AS path: I
        Communities: evpn-default-gateway
        Route Label: 299776

```

Meaning

The MAC plus IP route for the local IRB interface appears in the EVPN instance route table on Router PE1 and is learned from EVPN and tagged with the default gateway extended community.

Verifying Remote IRB IPs

Purpose

Verify that the remote IRB IPs are learned from BGP.

Action

On Router PE1, determine the MAC and IP addresses of the local IRB interface.

From operational mode, run the `show interfaces irb extensive | match "Current address"` command.

```

user@PE1> show interfaces irb extensive | match
"Current address"
Current address: a8:d0:e5:54:0d:10, Hardware address: a8:d0:e5:54:0d:10

```

From operational mode, run the `show interfaces irb.0 terse | match inet` command.

```
user@PE1> show interfaces irb.0 terse | match
inet
irb.0          up    up    inet    10.0.0.251/24
```

On Router PE2, verify that the remote IRB IPs are learnt.

From operational mode, run the `show route table evpn.evpn.0 extensive | find "a8:d0:e5:54:0d:10::10.0.0.251"` command.

```
user@PE2> show route table evpn.evpn.0 extensive
| find "a8:d0:e5:54:0d:10::10.0.0.251"
2:10.255.0.1:100::0::100::a8:d0:e5:54:0d:10::10.0.0.251/384 (1 entry, 1 announced)
    *BGP    Preference: 170/-101
        Route Distinguisher: 203.0.113.2:100
        Next hop type: Indirect
        Address: 0x26f8d6c
        Next-hop reference count: 10
        Source: 10.255.0.1
        Protocol next hop: 10.255.0.1
        Indirect next hop: 0x2 no-forward INH Session ID: 0x0
        State: <Secondary Active Int Ext>
        Local AS: 100 Peer AS: 100
        Age: 23:56:36 Metric2: 1
        Validation State: unverified
        Task: BGP_100.10.255.0.1
        Announcement bits (1): 0-evpn-evpn
        AS path: I
        Communities: target:100:100 evpn-default-gateway
        Import Accepted
        Route Label: 299776
        Localpref: 100
        Router ID: 10.255.0.1
        Primary Routing Table bgp.evpn.0
        Indirect next hops: 1
            Protocol next hop: 10.255.0.1 Metric: 1
            Indirect next hop: 0x2 no-forward INH Session ID: 0x0
            Indirect path forwarding next hops: 1
                Next hop type: Router
                Next hop: 203.0.113.2 via ge-1/0/8.0
                Session Id: 0x1
```

```

10.255.0.1/32 Originating RIB: inet.3
Metric: 1                      Node path count: 1
Forwarding nexthops: 1
Nexthop: 203.0.113.2 via ge-1/0/8.0

```

Meaning

The MAC plus IP route for the remote IRB interface appears in the EVPN instance route table on Router PE2 and is tagged with the default gateway extended community.

Verifying CE-CE Inter-Subnet Forwarding

Purpose

Verify inter-subnet forwarding between Routers CE1 and CE2.

Action

From operational mode, run the `show route table inet.0` command.

```

user@CE1> show route table inet.0
inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[Static/5] 00:15:09
                   > to 10.0.0.251 via ge-1/1/7.0
10.0.0.0/24        *[Direct/0] 1d 23:24:30
                   > via ge-1/1/7.0
10.0.0.1/32        *[Local/0] 1d 23:24:38
                   Local via ge-1/1/7.0

```

From operational mode, run the `ping` command.

```

user@CE1> ping 192.0.2.0 interval 0.1 count 10
PING 192.0.2.0 (192.0.2.0): 56 data bytes
64 bytes from 192.0.2.0: icmp_seq=0 ttl=63 time=0.919 ms
64 bytes from 192.0.2.0: icmp_seq=1 ttl=63 time=0.727 ms
64 bytes from 192.0.2.0: icmp_seq=2 ttl=63 time=0.671 ms
64 bytes from 192.0.2.0: icmp_seq=3 ttl=63 time=0.671 ms
64 bytes from 192.0.2.0: icmp_seq=4 ttl=63 time=0.666 ms

```

```

64 bytes from 192.0.2.0: icmp_seq=5 ttl=63 time=0.704 ms
64 bytes from 192.0.2.0: icmp_seq=6 ttl=63 time=0.763 ms
64 bytes from 192.0.2.0: icmp_seq=7 ttl=63 time=0.750 ms
64 bytes from 192.0.2.0: icmp_seq=8 ttl=63 time=12.967 ms
64 bytes from 192.0.2.0: icmp_seq=9 ttl=63 time=0.752 ms

--- 192.0.2.0 ping statistics ---
10 packets transmitted, 10 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.666/1.959/12.967/3.670 ms

```

Meaning

Ping from Router CE1 to Router CE2 is successful.

Verifying CE-PE Inter-Subnet Forwarding

Purpose

Verify inter-subnet forwarding between Routers CE1 and PE2.

Action

From operational mode, run the `show route table inet.0` command.

```

user@CE1> show route table inet.0
inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[Static/5] 00:15:09
                   > to 10.0.0.251 via ge-1/1/7.0
10.0.0.0/24        *[Direct/0] 1d 23:24:30
                   > via ge-1/1/7.0
10.0.0.1/32        *[Local/0] 1d 23:24:38
                   Local via ge-1/1/7.0

```

From operational mode, run the `ping` command.

```

user@CE1> ping 192.0.2.5 interval 0.1 count 10
PING 192.0.2.5 (192.0.2.5): 56 data bytes
64 bytes from 192.0.2.5: icmp_seq=0 ttl=64 time=0.959 ms

```

```

64 bytes from 192.0.2.5: icmp_seq=1 ttl=64 time=0.710 ms
64 bytes from 192.0.2.5: icmp_seq=2 ttl=64 time=0.832 ms
64 bytes from 192.0.2.5: icmp_seq=3 ttl=64 time=0.754 ms
64 bytes from 192.0.2.5: icmp_seq=4 ttl=64 time=3.642 ms
64 bytes from 192.0.2.5: icmp_seq=5 ttl=64 time=0.660 ms
64 bytes from 192.0.2.5: icmp_seq=6 ttl=64 time=0.728 ms
64 bytes from 192.0.2.5: icmp_seq=7 ttl=64 time=0.725 ms
64 bytes from 192.0.2.5: icmp_seq=8 ttl=64 time=0.674 ms
64 bytes from 192.0.2.5: icmp_seq=9 ttl=64 time=0.760 ms

--- 192.0.2.5 ping statistics ---
10 packets transmitted, 10 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.666/1.959/12.967/3.670 ms

```

Meaning

Ping from Router CE1 to Router PE2 is successful.

Verifying PE-PE Inter-Subnet Forwarding

Purpose

Verify inter-subnet forwarding between Routers PE1 and PE2.

Action

From operational mode, run the `show route table vrf.inet.0 192.0.2.5` command.

```

user@PE1> show route table vrf.inet.0 192.0.2.5
vrf.inet.0: 8 destinations, 8 routes (8 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.0.2.5/24      *[EVPN/7] 02:24:32, metric2 1
                  > to 1.0.0.2 via ge-1/0/8.0, Push 299776, Push 299776(top)

```

From operational mode, run the `ping` command.

```

user@CE1> ping routing-instance evpna
192.0.2.5 interval 0.1 count 10
PING 192.0.2.5 (192.0.2.5): 56 data bytes

```

```

64 bytes from 192.0.2.5: icmp_seq=0 ttl=64 time=9.613 ms
64 bytes from 192.0.2.5: icmp_seq=1 ttl=64 time=0.789 ms
64 bytes from 192.0.2.5: icmp_seq=2 ttl=64 time=0.803 ms
64 bytes from 192.0.2.5: icmp_seq=3 ttl=64 time=0.695 ms
64 bytes from 192.0.2.5: icmp_seq=4 ttl=64 time=0.742 ms
64 bytes from 192.0.2.5: icmp_seq=5 ttl=64 time=0.702 ms
64 bytes from 192.0.2.5: icmp_seq=6 ttl=64 time=0.725 ms
64 bytes from 192.0.2.5: icmp_seq=7 ttl=64 time=0.730 ms
64 bytes from 192.0.2.5: icmp_seq=8 ttl=64 time=0.713 ms
64 bytes from 192.0.2.5: icmp_seq=9 ttl=64 time=7.370 ms

--- 192.0.2.5 ping statistics ---
10 packets transmitted, 10 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.695/2.288/9.613/3.142 ms

```

Meaning

Ping from Router PE1 to Router PE2's IRB interface is successful.

SEE ALSO

[EVPN with IRB Solution Overview](#) | 946

Example: Configuring an EVPN with IRB Solution on EX9200 Switches

IN THIS SECTION

- [Requirements](#) | 996
- [Overview](#) | 996
- [Configuration](#) | 996
- [Verification](#) | 1005

This example shows how to configure an integrated routing and bridging (IRB) solution in an Ethernet VPN (EVPN) deployment.

Requirements

This example uses the following hardware and software components:

- Two EX9200 switches configured as PE routers
- Junos OS Release 14.2 or later running on all the PE routers

Before you begin:

1. Configure the switch interfaces.
2. Configure OSPF or any other IGP protocol.
3. Configure BGP.
4. Configure LDP.
5. Configure MPLS.

Overview

In an EVPN solution, multiple VLANs can be defined within a particular EVPN instance, and one or more EVPN instances can be associated with a single Layer 3 VPN VRF. In general, each data center tenant is assigned a unique Layer 3 VPN virtual route forwarding (VRF), although the tenant can comprise one or more EVPN instances or VLANs per EVPN instance.

To support this flexibility and scalability factor, the EVPN solution provides support for the IRB interfaces on EX9200 switches to facilitate optimal Layer 2 and Layer 3 forwarding along with virtual machine mobility. The IRB interfaces are configured on each configured VLAN including the default VLAN for an EVPN instance.

IRB is the ability to do Layer 2 switching and Layer 3 routing within a single node, thus avoiding extra hops for inter-subnet traffic. The EVPN IRB solution eliminates the default gateway problem using the gateway MAC and IP synchronization, and avoids the triangular routing problem with Layer 3 interworking by creating IP host routes for virtual machines (VMs) in the tenant VRFs.

Configuration

IN THIS SECTION

- [Procedure | 997](#)
- [Results | 1002](#)

Procedure

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

CE1

```
set interfaces ge-1/1/7 vlan-tagging
set interfaces ge-1/1/7 unit 0 vlan-id 10
set interfaces ge-1/1/7 unit 0 family inet address 10.0.0.1/24
set routing-options static route 198.51.100.0/24 next-hop 10.0.0.251
```

PE1

```
set interfaces ge-1/0/8 unit 0 family inet address 192.0.2.1/24
set interfaces ge-1/0/8 unit 0 family mpls
set interfaces ge-1/1/8 flexible-vlan-tagging
set interfaces ge-1/1/8 encapsulation flexible-ethernet-services
set interfaces ge-1/1/8 unit 0 encapsulation vlan-bridge
set interfaces ge-1/1/8 unit 0 vlan-id 10
set interfaces irb unit 0 family inet address 10.0.0.251/24
set interfaces lo0 unit 0 family inet address 203.0.113.1/32
set routing-options router-id 203.0.113.1
set routing-options autonomous-system 100
set routing-options forwarding-table chained-composite-next-hop ingress evpn
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 203.0.113.1
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 203.0.113.2
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set routing-instances evpna instance-type evpn
set routing-instances evpna vlan-id 10
set routing-instances evpna interface ge-1/1/8.0
set routing-instances evpna l3-interface irb.0
```



```

set routing-instances evpna route-distinguisher 203.0.113.1:100
set routing-instances evpna vrf-target target:100:100
set routing-instances evpna protocols evpn interface ge-1/1/8.0
set routing-instances vrf instance-type vrf
set routing-instances vrf interface irb.0
set routing-instances vrf route-distinguisher 203.0.113.1:300
set routing-instances vrf vrf-target target:100:300
set routing-instances vrf vrf-table-label

```

PE2

```

set interfaces ge-2/0/8 unit 0 family inet address 192.0.2.2/24
set interfaces ge-2/0/8 unit 0 family mpls
set interfaces ge-2/1/8 flexible-vlan-tagging
set interfaces ge-2/1/8 encapsulation flexible-ethernet-services
set interfaces ge-2/1/8 unit 0 encapsulation vlan-bridge
set interfaces ge-2/1/8 unit 0 vlan-id 20
set interfaces irb unit 0 family inet address 198.51.100.251/24
set interfaces lo0 unit 0 family inet address 203.0.113.2/32
set routing-options router-id 203.0.113.2
set routing-options autonomous-system 100
set routing-options forwarding-table chained-composite-next-hop ingress evpn
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 203.0.113.2
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 203.0.113.1
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set routing-instances evpna instance-type evpn
set routing-instances evpna vlan-id 20
set routing-instances evpna interface ge-2/1/8.0
set routing-instances evpna l3-interface irb.0
set routing-instances evpna route-distinguisher 203.0.113.2:100
set routing-instances evpna vrf-target target:200:100
set routing-instances evpna protocols evpn interface ge-2/1/8.0
set routing-instances vrf instance-type vrf
set routing-instances vrf interface irb.0
set routing-instances vrf route-distinguisher 203.0.113.2:300

```

```
set routing-instances vrf vrf-target target:200:300
set routing-instances vrf vrf-table-label
```

CE2

```
set interfaces ge-2/1/7 unit 0 vlan-id 20
set interfaces ge-2/1/7 unit 0 family inet address 198.51.100.2/24
set routing-options static route 10.0.0.0/24 next-hop 198.51.100.251
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure Router PE1:

NOTE: Repeat this procedure for Router PE2, after modifying the appropriate interface names, addresses, and other parameters.

1. Configure Router PE1 interfaces.

```
[edit interfaces]
user@PE1# set ge-1/0/8 unit 0 family inet address 192.0.2.1/24
user@PE1# set ge-1/0/8 unit 0 family mpls
user@PE1# set ge-1/1/8 flexible-vlan-tagging
user@PE1# set ge-1/1/8 encapsulation flexible-ethernet-services
user@PE1# set ge-1/1/8 unit 0 encapsulation vlan-bridge
user@PE1# set ge-1/1/8 unit 0 vlan-id 10
user@PE1# set irb unit 0 family inet address 10.0.0.251/24
user@PE1# set lo0 unit 0 family inet address 203.0.113.1/32
```

2. Set the router ID and autonomous system number for Router PE1.

```
[edit routing-options]
user@PE1# set router-id 203.0.113.1
user@PE1# set autonomous-system 100
```

3. Configure the chained composite next hop for EVPN.

```
[edit routing-options]
user@PE1# set forwarding-table chained-composite-next-hop ingress evpn
```

4. Enable LDP on all interfaces of Router PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set ldp interface all
user@PE1# set ldp interface fxp0.0 disable
```

5. Enable MPLS on all the interfaces of Router PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set mpls interface all
user@PE1# set mpls mpls interface fxp0.0 disable
```

6. Configure the BGP group for Router PE1.

```
[edit protocols]
user@PE1# set bgp group ibgp type internal
```

7. Assign local and neighbor addresses to the ibgp BGP group for Router PE1 to peer with Router PE2.

```
[edit protocols]
user@PE1# set bgp group ibgp local-address 203.0.113.1
user@PE1# set bgp group ibgp neighbor 203.0.113.2
```

8. Include the EVPN signaling Network Layer Reachability Information (NLRI) to the ibgp BGP group.

```
[edit protocols]
user@PE1# set bgp group ibgp family evpn signaling
```

9. Configure OSPF on all the interfaces of Router PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set ospf area 0.0.0.0 interface all
user@PE1# set ospf area 0.0.0.0 interface fxp0.0 disable
```

10. Configure the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpna instance-type evpn
```

11. Set the VLAN identifier for the bridging domain in the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna vlan-id 10
```

12. Configure the interface name for the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna interface ge-1/1/8.0
```

13. Configure the IRB interface as the routing interface for the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna l3-interface irb.0
```

14. Configure the route distinguisher for the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna route-distinguisher 203.0.113.1:100
```

15. Configure the VPN routing and forwarding (VRF) target community for the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna vrf-target target:100:100
```

16. Assign the interface name that connects the PE1 site to the VPN.

```
[edit routing-instances]
user@PE1# set evpna protocols evpn interface ge-1/1/8.0
```

17. Configure the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf instance-type vrf
```

18. Configure the IRB interface as the routing interface for the vrf routing instance.

```
[edit routing-instances]
user@PE1# set vrf interface irb.0
```

19. Configure the route distinguisher for the vrf routing instance.

```
[edit routing-instances]
user@PE1# set vrf route-distinguisher 203.0.113.1:300
```

20. Configure the VRF label for the vrf routing instance.

```
[edit routing-instances]
user@PE1# set vrf vrf-table-label
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show routing-options`, `show protocols`, and `show routing-instances` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show interfaces
ge-1/0/8 {
  unit 0 {
    family inet {
      address 192.0.2.1/24;
    }
  }
}
```

```

        family mpls;
    }
}
ge-1/1/8 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 0 {
        encapsulation vlan-bridge;
        vlan-id 10;
    }
}
irb {
    unit 0 {
        family inet {
            address 10.0.0.251/24;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 203.0.113.1/32 {
            }
        }
    }
}
}

```

```

user@PE1# show routing-options
router-id 203.0.113.1;
autonomous-system 100;
forwarding-table {
    chained-composite-next-hop {
        ingress {
            evpn;
        }
    }
}

```

```

user@PE1# show protocols
ldp {

```

```

interface all;
interface fxp0.0 {
    disable;
}
}
mpls {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
bgp {
    group ibgp {
        type internal;
        local-address 203.0.113.1;
        family evpn {
            signaling;
        }
        neighbor 203.0.113.2;
    }
}
ospf {
    area 0.0.0.0 {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
}
}

```

```

user@PE1# show routing-instances
evpna {
    instance-type evpn;
    vlan-id 10;
    interface ge-1/1/8.0;
    l3-interface irb.0;
    route-distinguisher 203.0.113.1:100;
    vrf-target target:100:100;
    protocols {
        evpn {
            interface ge-1/1/8.0;

```

```
    }  
  }  
}  
vrf {  
  instance-type vrf;  
  interface irb.0;  
  route-distinguisher 203.0.113.1:300;  
  vrf-target target:100:300;  
  vrf-table-label;  
}
```

Verification

IN THIS SECTION

- [Verifying Local IRB MACs | 1005](#)
- [Verifying Remote IRB MACs | 1007](#)
- [Verifying Local IRB IPs | 1008](#)
- [Verifying Remote IRB IPs | 1009](#)
- [Verifying CE-CE Inter-Subnet Forwarding | 1011](#)

Confirm that the configuration is working properly.

Verifying Local IRB MACs

Purpose

Verify that the local IRB MACs are learned from L2ALD.

Action

On Router PE1, determine the MAC address of the local IRB interface.

From operational mode, run the `show interfaces irb extensive | match "Current address"` command.

```
user@PE1> show interfaces irb extensive | match
"Current address"
Current address: a8:d0:e5:54:0d:10, Hardware address: a8:d0:e5:54:0d:10
```

From operational mode, run the `show route table evpn.evpn.0 extensive | find "a8:d0:e5:54:0d:10"` command.

```
user@PE1> show route table evpn.evpn.0 extensive
| find "a8:d0:e5:54:0d:10"
2:10.255.0.1:100::0::100::a8:d0:e5:54:0d:10/384 (1 entry, 1 announced)
TSI:
Page 0 idx 0, (group PE type Internal) Type 1 val 0x2736568 (adv_entry)
  Advertised metrics:
    Flags: Nexthop Change
    Nexthop: Self
    Localpref: 100
    AS path: [100] I
    Communities: target:100:100 evpn-default-gateway
Path 2:10.255.0.1:100::0::100::a8:d0:e5:54:0d:10 Vector len 4. Val: 0
  *EVPN Preference: 170
    Next hop type: Indirect
    Address: 0x26f8354
    Next-hop reference count: 6
    Protocol next hop: 10.255.0.1
    Indirect next hop: 0x0 - INH Session ID: 0x0
    State: <Active Int Ext>
    Age: 23:29:08
    Validation State: unverified
    Task: evpn-evpn
    Announcement bits (1): 1-BGP_RT_Background
    AS path: I
    Communities: evpn-default-gateway
    Route Label: 299776
```

Meaning

The MAC-only route for the local IRB interface appears in the EVPN instance route table on Router PE1 and is learned from EVPN and tagged with the default gateway extended community.

Verifying Remote IRB MACs

Purpose

Verify that the remote IRB MACs are learned from BGP.

Action

On Router PE1, determine the MAC address of the local IRB interface.

From operational mode, run the `show interfaces irb extensive | match "Current address"` command.

```
user@PE1> show interfaces irb extensive | match
"Current address"
Current address: a8:d0:e5:54:0d:10, Hardware address: a8:d0:e5:54:0d:10
```

On Router PE2, verify that the remote IRB MACs are learned.

From operational mode, run the `show route table evpna.evpn.0 extensive | find "a8:d0:e5:54:0d:10"` command.

```
user@PE2> show route table evpna.evpn.0 extensive
| find "a8:d0:e5:54:0d:10"
2:10.255.0.1:100::0::100::a8:d0:e5:54:0d:10/384 (1 entry, 1 announced)
  *BGP    Preference: 170/-101
          Route Distinguisher: 2.91.223.24:100
          Next hop type: Indirect
          Address: 0x26f8d6c
          Next-hop reference count: 10
          Source: 10.255.0.1
          Protocol next hop: 10.255.0.1
          Indirect next hop: 0x2 no-forward INH Session ID: 0x0
          State: <Secondary Active Int Ext>
          Local AS: 100 Peer AS: 100
          Age: 23:22:17 Metric2: 1
          Validation State: unverified
          Task: BGP_100.10.255.0.1
          Announcement bits (1): 0-evpna-evpn
          AS path: I
          Communities: target:100:100 evpn-default-gateway
          Import Accepted
          Route Label: 299776
          Localpref: 100
```

```

Router ID: 10.255.0.1
Primary Routing Table bgp.evpn.0
Indirect next hops: 1
    Protocol next hop: 10.255.0.1 Metric: 1
    Indirect next hop: 0x2 no-forward INH Session ID: 0x0
    Indirect path forwarding next hops: 1
        Next hop type: Router
        Next hop: 1.0.0.1 via ge-1/0/8.0
        Session Id: 0x1
    10.255.0.1/32 Originating RIB: inet.3
        Metric: 1                      Node path count: 1
        Forwarding nexthops: 1
            Nexthop: 1.0.0.1 via ge-1/0/8.0

```

Meaning

The MAC-only route for the remote IRB interface appears in the EVPN instance route table on Router PE2 and is learned from BGP and tagged with the default gateway extended community.

Verifying Local IRB IPs

Purpose

Verify that the local IRB IPs are learned locally by RPD.

Action

On Router PE1, determine the MAC and IP addresses of the local IRB interface.

From operational mode, run the `show interfaces irb extensive | match "Current address"` command.

```

user@PE1> show interfaces irb extensive | match
"Current address"
Current address: a8:d0:e5:54:0d:10, Hardware address: a8:d0:e5:54:0d:10

```

From operational mode, run the `show interfaces irb.0 terse | match inet` command.

```

user@PE1> show interfaces irb.0 terse | match
inet
irb.0          up    up    inet    10.0.0.251/24

```

From operational mode, run the `show route table evpn.evpn.0 extensive | find "a8:d0:e5:54:0d:10::10.0.0.251"` command.

```
user@PE2> show route table evpn.evpn.0 extensive
| find "a8:d0:e5:54:0d:10::10.0.0.251"
2:10.255.0.1:100::0::100::a8:d0:e5:54:0d:10::10.0.0.251/384 (1 entry, 1 announced)
TSI:
Page 0 idx 0, (group PE type Internal) Type 1 val 0x27365a0 (adv_entry)
  Advertised metrics:
    Flags: Nexthop Change
    Nexthop: Self
    Localpref: 100
    AS path: [100] I
    Communities: target:100:100 evpn-default-gateway
Path 2:10.255.0.1:100::0::100::a8:d0:e5:54:0d:10::10.0.0.251 Vector len 4. Val: 0
  *EVPN Preference: 170 <<<<<
    Next hop type: Indirect
    Address: 0x26f8354
    Next-hop reference count: 6
    Protocol next hop: 10.255.0.1
    Indirect next hop: 0x0 - INH Session ID: 0x0
    State: <Active Int Ext>
    Age: 23:48:46
    Validation State: unverified
    Task: evpn-evpn
    Announcement bits (1): 1-BGP_RT_Background
    AS path: I
    Communities: evpn-default-gateway
    Route Label: 299776
```

Meaning

The MAC plus IP route for the local IRB interface appears in the EVPN instance route table on Router PE1 and is learned from EVPN and tagged with the default gateway extended community.

Verifying Remote IRB IPs

Purpose

Verify that the remote IRB IPs are learned from BGP.

Action

On Router PE1, determine the MAC and IP addresses of the local IRB interface.

From operational mode, run the `show interfaces irb extensive | match "Current address"` command.

```
user@PE1> show interfaces irb extensive | match
"Current address"
Current address: a8:d0:e5:54:0d:10, Hardware address: a8:d0:e5:54:0d:10
```

From operational mode, run the `show interfaces irb.0 terse | match inet` command.

```
user@PE1> show interfaces irb.0 terse | match
inet
irb.0          up    up    inet    10.0.0.251/24
```

On Router PE2, verify that the remote IRB IPs are learnt.

From operational mode, run the `show route table evpna.evpn.0 extensive | find "a8:d0:e5:54:0d:10::10.0.0.251"` command.

```
user@PE2> show route table evpna.evpn.0 extensive
| find "a8:d0:e5:54:0d:10::10.0.0.251"
2:10.255.0.1:100::0::100::a8:d0:e5:54:0d:10::10.0.0.251/384 (1 entry, 1 announced)
    *BGP    Preference: 170/-101
        Route Distinguisher: 2.91.223.216:100
        Next hop type: Indirect
        Address: 0x26f8d6c
        Next-hop reference count: 10
        Source: 10.255.0.1
        Protocol next hop: 10.255.0.1
        Indirect next hop: 0x2 no-forward INH Session ID: 0x0
        State: <Secondary Active Int Ext>
        Local AS: 100 Peer AS: 100
        Age: 23:56:36 Metric2: 1
        Validation State: unverified
        Task: BGP_100.10.255.0.1
        Announcement bits (1): 0-evpna-evpn
        AS path: I
        Communities: target:100:100 evpn-default-gateway
        Import Accepted
```

```

Route Label: 299776
Localpref: 100
Router ID: 10.255.0.1
Primary Routing Table bgp.evpn.0
Indirect next hops: 1
    Protocol next hop: 10.255.0.1 Metric: 1
    Indirect next hop: 0x2 no-forward INH Session ID: 0x0
    Indirect path forwarding next hops: 1
        Next hop type: Router
        Next hop: 1.0.0.1 via ge-1/0/8.0
        Session Id: 0x1
    10.255.0.1/32 Originating RIB: inet.3
    Metric: 1                      Node path count: 1
    Forwarding nexthops: 1
        Nexthop: 1.0.0.1 via ge-1/0/8.0

```

Meaning

The MAC plus IP route for the remote IRB interface appears in the EVPN instance route table on Router PE2 and is tagged with the default gateway extended community.

Verifying CE-CE Inter-Subnet Forwarding

Purpose

Verify inter-subnet forwarding between Routers CE1 and CE2.

Action

From operational mode, run the `show route table inet.0` command.

```

user@CE1> show route table inet.0
inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[Static/5] 00:15:09
                   > to 10.0.0.251 via ge-1/1/7.0
10.0.0.0/24       *[Direct/0] 1d 23:24:30
                   > via ge-1/1/7.0

```

```
10.0.0.1/32      *[Local/0] 1d 23:24:38
                  Local via ge-1/1/7.0
```

From operational mode, run the ping command.

```
user@CE1> ping 198.51.100.2 interval 0.1 count 10
PING 198.51.100.2 (20.0.0.2): 56 data bytes
64 bytes from 198.51.100.2: icmp_seq=0 ttl=63 time=0.919 ms
64 bytes from 198.51.100.2: icmp_seq=1 ttl=63 time=0.727 ms
64 bytes from 198.51.100.2: icmp_seq=2 ttl=63 time=0.671 ms
64 bytes from 198.51.100.2: icmp_seq=3 ttl=63 time=0.671 ms
64 bytes from 198.51.100.2: icmp_seq=4 ttl=63 time=0.666 ms
64 bytes from 198.51.100.2: icmp_seq=5 ttl=63 time=0.704 ms
64 bytes from 198.51.100.2: icmp_seq=6 ttl=63 time=0.763 ms
64 bytes from 198.51.100.2: icmp_seq=7 ttl=63 time=0.750 ms
64 bytes from 198.51.100.2: icmp_seq=8 ttl=63 time=12.967 ms
64 bytes from 198.51.100.2: icmp_seq=9 ttl=63 time=0.752 ms

--- 198.51.100.2 ping statistics ---
10 packets transmitted, 10 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.666/1.959/12.967/3.670 ms
```

Meaning

Ping from Router CE1 to Router CE2 is successful.

RELATED DOCUMENTATION

| [An EVPN with IRB Solution on EX9200 Switches Overview](#) | 952

Configuring IGMP or MLD Snooping with EVPN-MPLS

IN THIS CHAPTER

- Overview of Multicast Forwarding with IGMP or MLD Snooping in an EVPN-MPLS Environment | 1013
- Configure Multicast Forwarding with IGMP Snooping in an EVPN-MPLS Environment | 1028
- Configure Multicast Forwarding with MLD Snooping in an EVPN-MPLS Environment | 1038

Overview of Multicast Forwarding with IGMP or MLD Snooping in an EVPN-MPLS Environment

IN THIS SECTION

- Benefits of Multicast Forwarding with IGMP or MLD Snooping in an EVPN-MPLS Environment | 1014
- Multicast Forwarding with IGMP or MLD Snooping in Single-homed or Multihomed Ethernet Segments | 1014
- IGMP or MLD Snooping with Multicast Forwarding Between Bridge Domains or VLANs Using IRB Interfaces | 1025

To help optimize multicast traffic flow in an Ethernet VPN (EVPN) over MPLS environment, you can enable IGMP snooping for IPv4 multicast traffic or MLD snooping for IPv6 multicast traffic. In this environment, multicast receiver hosts in the EVPN instance (EVI) can be single-homed to one provider edge (PE) device or multihomed in all-active mode to multiple provider edge (PE) devices. For receivers that are multihomed to multiple PE devices, the peer PE devices synchronize IGMP or MLD state information. Receivers can be attached to PE devices in the EVI at the same site or at different sites.

Source hosts can be single-homed or multihomed to PE devices within the EVI in some supported use cases. For other use cases, the sources must reside outside the EVPN network in an external PIM

domain, and each PE device that needs to receive multicast traffic connects through a Layer 3 interface to the PIM gateway router.

You can enable IGMP and MLD snooping for:

- Multiple EVPN instances
- Specific bridge domains or VLANs in an EVPN instance
- All bridge domains or VLANs within an EVPN virtual switch instance (on supporting devices)

Devices in this environment can forward multicast traffic within a bridge domain or VLAN, and route multicast traffic across bridge domains or VLANs at Layer 3 using IRB interfaces.

Benefits of Multicast Forwarding with IGMP or MLD Snooping in an EVPN-MPLS Environment

- In an environment with significant multicast traffic, using IGMP or MLD snooping constrains multicast traffic in a broadcast domain or VLAN to interested receivers and multicast devices, which conserves network bandwidth.
- Synchronizing IGMP or MLD state among all EVPN PE devices for multihomed receivers ensures that all subscribed listeners receive multicast traffic, even in cases such as the following:
 - IGMP or MLD membership reports for a multicast group might arrive on a PE device that is not the Ethernet segment's designated forwarder (DF).
 - An IGMP or MLD message to leave a multicast group arrives at a different PE device than the PE device where the corresponding join message for the group was received.

Multicast Forwarding with IGMP or MLD Snooping in Single-homed or Multihomed Ethernet Segments

Hosts in the network send IGMP or MLD reports expressing interest in particular multicast groups from IPv4 multicast sources (using IGMP) or IPv6 multicast sources (using MLD). PE devices with IGMP or MLD snooping enabled listen to (snoop) IGMP or MLD packets. The PEs then use the snooped information to establish multicast routes that only forward traffic to interested receivers for a multicast group.

For redundancy, your network can include multicast receivers multihomed to a set of peer PE devices in all-active mode. In some use cases, sources can be multihomed to peer PE devices within the EVI. If you enable snooping on all PE devices in the EVI, the multihomed peer PE devices synchronize the IGMP or MLD state among themselves so multicast traffic can successfully reach all listeners.

The next sections describe the IGMP and MLD group membership report processing and subsequent multicast traffic flow in this environment.

IGMP or MLD State Synchronization Among Multihoming Peer PE Devices

In an EVI with receivers that are multihomed to multiple PE devices, corresponding IGMP or MLD join and leave messages for multicast group management might not be sent to the same PE device. As a result, all the PE devices must synchronize and share IGMP and MLD state.

PE devices with snooping enabled in this environment exchange BGP EVPN Type 7 (Join Sync Route) and Type 8 (Leave Sync Route) network layer reachability information (NLRI) to synchronize IGMP or MLD membership reports received on multihomed interfaces. The network must use multihoming in all-active mode.

The advertised EVPN Type 7 and Type 8 routes also carry EVI route target extended community attributes associated with multihomed EVIs to support multiple EVPN routing instances simultaneously. Only PE devices that share the same Ethernet segment ID import these routes.

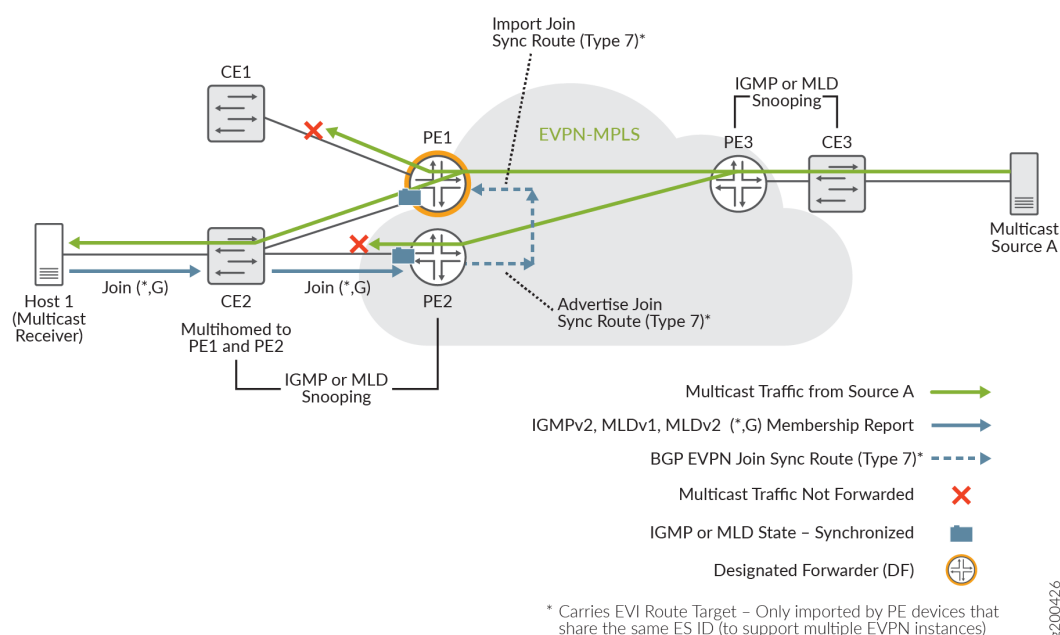
PE devices serving single-homed hosts do not need to advertise or import EVPN Type 7 and Type 8 routes. However, to save bandwidth on the access side by only forwarding multicast traffic to interested receivers, you should also configure IGMP or MLD snooping in this case.

Due to platform-specific differences among the devices that support multicast with IGMP or MLD snooping in an EVPN-MPLS multihomed environment, we offer slightly different solutions on the supporting platforms in certain use cases.

EX9200 Switches, MX Series Routers, and vMX Routers—Multicast State Synchronization for Multihoming in EVPN-MPLS

EX9200 switches, MX Series routers, and vMX routers support EVPN-MPLS multihoming with IGMP and MLD snooping and synchronize the IGMP or MLD state as shown in Figure 103 on page 1016.

Figure 103: Multicast Forwarding for Multihomed Receivers in EVPN-MPLS on EX9200 Switches, MX Series Routers, and VMX Routers



This example topology includes a multicast source (Multicast Source A) behind PE3 by way of customer edge device CE3, and a multicast receiver Host 1 behind CE2 that is multihomed to PE1 and PE2.

Multihoming peer PE2 receives a membership request from Host 1, and advertises the EVPN Type 7 Join Sync Route to the EVPN core. PE1 imports the Type 7 route to synchronize the IGMP or MLD state among the PE devices to which the receiver is multihomed (PE1 and PE2). When both PE1 and PE2 receive multicast traffic from the EVPN-MPLS network, only PE1, the designated forwarder (DF), forwards the traffic to CE2 to reach the interested listener on Host 1. PE1 doesn't forward the traffic to CE1, which has no interested listeners.

EX9200 switches, MX Series routers and vMX routers support IGMP and MLD snooping with EVPN-MPLS multihoming under the following constraints:

- All multihoming peer PE devices must support BGP EVPN Type 7 Join Sync Routes and Type 8 Leave Sync Routes.

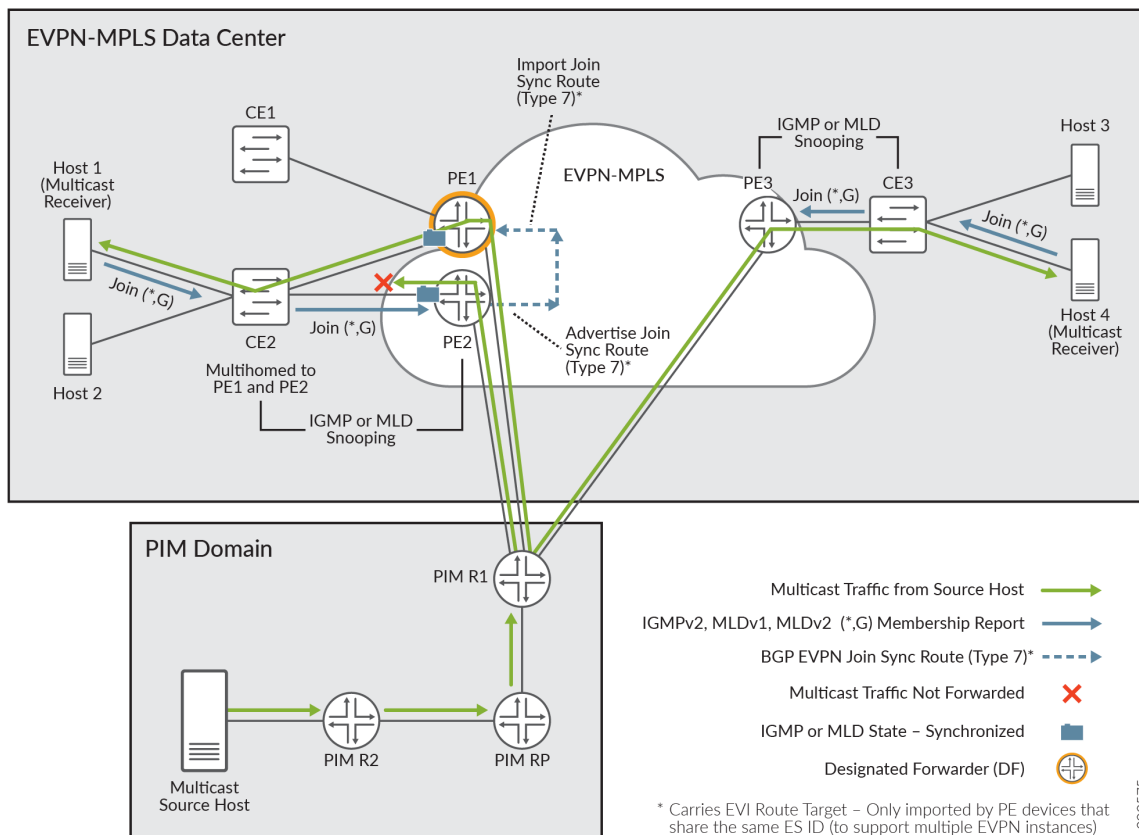
- You can configure IGMP snooping or MLD snooping for multiple routing instances of type evpn or type virtual-switch.
- You must configure all bridge domains or VLANs with multicast sources and receivers on all PE devices in the EVI.
- You must configure MX Series PE devices in enhanced IP Network Services mode. (See [Network Services Mode Overview](#).)
- Routing multicast traffic between bridge domains or VLANs is through IRB interfaces only. You can't have a topology that uses an external multicast router for intersubnet multicast routing in this environment. (Using an external multicast router instead of IRB interfaces is an available solution with EVPN-VXLAN networks.)
- Multicast sources and receivers must be within the EVI. Receiving or forwarding multicast traffic from or to devices outside of this environment using a PIM gateway is not supported.
- Selective multicast forwarding (advertising EVPN Type 6 Selective Multicast Ethernet Tag (SMET) routes for forwarding only to interested receivers) in the EVPN core is not supported.
- IGMP and MLD snooping are not supported with point-to-multipoint (P2MP) multipoint LDP/RSVP replication; only ingress replication is supported.

ACX Series Routers—Multicast State Synchronization for Multihoming in EVPN-MPLS

On ACX Series routers with EVPN-MPLS multihoming and IGMP and MLD snooping, to support routing multicast traffic across bridge domains or VLANs (intersubnet multicast), any multicast sources must be outside of the EVPN-MPLS datacenter in a Layer 3 PIM domain. (An ACX Series router can't act as a multicast first-hop router, so multicast sources can't be connected to ACX routers for intersubnet routing.) Each ACX Series PE device must receive the multicast source traffic through a Layer 3 interface connected to the external PIM domain gateway router. See [Figure 104 on page 1018](#). If your EVPN-MPLS datacenter only needs to forward multicast traffic within bridge domains or VLANs (intrasubnet multicast), you can use a topology similar to [Figure 103 on page 1016](#) with sources inside the datacenter.

In both the intrasubnet and intersubnet use cases, IGMP or MLD state synchronization among multihoming peer PE devices works the same as the EVPN-MPLS multicast with multihoming solutions for other platforms.

Figure 104: Multicast Forwarding for Multihomed Receivers in EVPN-MPLS on ACX Series Routers



The example topology in [Figure 104 on page 1018](#) can support both intrasubnet and intersubnet multicast traffic because it includes a multicast source (Multicast Source Host) in an external PIM domain, and Layer 3 connections from the PE devices to the external PIM gateway router. Multicast receiver Host 1 behind CE2 is multihomed to peer PE devices PE1 and PE2. Single-homed multicast receiver Host 4 connects to PE3 by way of CE3. All of the EVPN PEs serving multicast receivers have IGMP (or MLD) snooping enabled. To optimize multicast forwarding on the access side, you can also enable IGMP or MLD snooping on the CE devices (if supported).

Multihoming peer PE2 receives a membership request from Host 1, and advertises the EVPN Type 7 Join Sync Route to the EVPN core. PE1 imports the Type 7 route to synchronize the IGMP or MLD state among the PE devices to which the receiver is multihomed (PE1 and PE2). PE3 receives a membership request from single-homed Host 4, and with no multihoming peers, doesn't need to synchronize IGMP or MLD state information.

ACX Series routers support IGMP and MLD snooping with EVPN-MPLS multihoming under the following constraints:

- All multihomed peer PE devices must support BGP EVPN Type 7 Join Sync Routes and Type 8 Leave Sync Routes.
- You can configure IGMP snooping or MLD snooping for one or more routing instances of type evpn only.
- You must configure all multicast bridge domains or VLANs on all PE devices in the EVI.
- Routing multicast traffic between bridge domains or VLANs is through IRB interfaces only. You can't have a topology that uses an external multicast router for intersubnet multicast routing instead of IRB interfaces. (Using an external multicast router instead of IRB interfaces is an available solution with EVPN-VXLAN networks on some platforms.)
- As mentioned previously, to support intersubnet multicast traffic routing, multicast sources must reside outside the data center in an external PIM domain. All PEs connect to the PIM gateway and send PIM join messages through the PIM rendezvous point (RP). The PIM gateway forwards traffic for the group to all PE devices.

NOTE: In this environment, multicast sources can be within the EVI only when the multicast traffic for a group is all within the same bridge domain or VLAN (intrasubnet multicast only).

- Selective multicast forwarding (advertising EVPN Type 6 Selective Multicast Ethernet Tag (SMET) routes for forwarding only to interested receivers) in the EVPN core is not supported.
- IGMP and MLD snooping are not supported with point-to-multipoint (P2MP) multipoint LDP/RSVP replication; only ingress replication is supported.

Leave Message Processing with Multihoming Peer PE Devices

Processing leave messages and membership route withdrawals in a multihomed environment is more complicated when the leave message is not received by the same PE device that processed the join message. PE devices use BGP EVPN Type 8 routes to facilitate leave operations as follows:

- A PE device that receives an IGMP or MLD leave message for a group advertises a Type 8 route. Other PE devices import the Type 8 route.
- The PE device that advertised the Type 8 route originates a membership query for any remaining group members, and starts a leave timer. The other PE devices that imported the Type 8 route likewise start a leave timer.

- If no join membership reports are received by the time the timer expires, the PE device that advertised the Type 7 route withdraws the Type 7 route. The PE device that originated the Type 8 route withdraws the Type 8 route.

IGMP or MLD Versions and Supported Group Membership Report Modes

Any-source multicast (ASM) or (*,G) group membership mode includes all sources for a multicast group's traffic. Source-specific multicast (SSM) or (S,G) mode handles multicast group membership based on a specific source and the group address.

By default, PE devices in this EVPN-MPLS environment support ASM mode with IGMPv2, IGMPv3, MLDv1, and MLDv2 under certain conditions. You can alternatively configure PE devices to process only SSM membership reports with IGMPv3 and MLDv2 in environments that support intersubnet traffic. PE devices don't support processing both ASM and SSM membership reports at the same time.

[Table 41 on page 1020](#) summarizes membership report modes supported with each IGMP or MLD version. IGMPv1 is not supported with EVPN-MPLS multicast.

Table 41: Group Membership Report Modes Supported for Each IGMP and MLD Version

IGMP or MLD Version	ASM (*,G) Only	SSM (S,G) Only	ASM (*,G) + SSM (S,G)
IGMPv1	-	-	-
IGMPv2	Yes	No	No
IGMPv3	Yes (Only on ACX Series)	Yes (Only on ACX Series, when explicitly configured)	No
MLDv1	Yes	No	No
MLDv2	Yes (ACX Series: Only with MLDv2 configured on all interfaces that receive multicast traffic)	Yes (Only when explicitly configured)	No

The next sections give more details on platform-specific PE device behavior when processing IGMP or MLD membership reports.

ASM and SSM Mode Behavior on EX9200 Switches, MX Series Routers, and vMX Routers

By default, the EVPN-MPLS network can process ASM (*,G) membership reports with IGMPv2, MLDv1, and MLDv2. If the network has hosts sending both MLDv1 and MLDv2 ASM reports for a given group, PE devices will process MLDv1 and MLDv2 reports for the group as MLDv1 membership reports. They drop and do not process any SSM reports.

You can alternatively enable IGMP snooping or MLD snooping with the `evpn-ssm-reports-only` option to explicitly configure PEs to accept and process SSM (S,G) membership reports with MLDv2. With this option, PEs only accept SSM reports, and drop and do not process any ASM reports.

- You can enable SSM-only processing for one or more bridge domains in the EVI.
- When enabling this option for a virtual switch, this behavior applies to all bridge domains in the virtual switch instance.

ASM and SSM Mode Behavior on ACX Series Routers

With IGMP snooping enabled and multicast traffic flow is *within* bridge domains or VLANs (intrasubnet):

- By default, PE devices process IGMPv2 ASM (*,G) membership reports, but discard any IGMPv3 reports.
- If you configure all interfaces that receive multicast traffic with IGMPv3, PE devices can process IGMPv3 reports in ASM (*,G) mode, but they discard IGMPv3 SSM (S,G) reports.

With MLD snooping enabled and multicast traffic flow is *within* bridge domains or VLANs (intrasubnet):

- By default, PE devices can process MLDv1 ASM (*,G) membership reports but discard any MLDv2 reports.
- If you configure all interfaces that receive multicast traffic with MLDv2, PE devices can process both MLDv1 and MLDv2 reports in ASM (*,G) mode, but they discard MLDv2 SSM (S,G) reports.

With IGMP snooping or MLD snooping enabled and routing multicast traffic *between* bridge domains or VLANs (intersubnet):

- With IGMP snooping, by default, PE devices process IGMPv2 and IGMPv3 membership reports in ASM (*,G) mode. They discard IGMPv3 SSM (S,G) reports.
- With MLD snooping, by default, PE devices process MLDv1 and MLDv2 membership reports in ASM (*,G) mode. They discard MLDv2 SSM (S,G) reports.
- If you enable IGMP snooping or MLD snooping with IGMPv3 or MLDv2 and the SSM-only processing option, `evpn-ssm-reports-only`, then PE devices process IGMPv3 or MLDv2 reports as SSM (S,G) only. They discard any ASM (*,G) reports.

In this case, for best results, we recommend that you configure IGMPv3 or MLDv2 on all IRB interfaces that do intersubnet routing.

How Multicast Traffic Forwarding Works with Single-homed or Multihomed Receivers

In an EVPN-MPLS network where hosts might be multihomed to more than one PE device, when a bridge domain (or VLAN) is configured on a PE device, the PE device signals a BGP EVPN Type 3 (Inclusive Multicast Ethernet Tag [IMET]) route to the other PE devices in the instance to build a core multicast replication tree for each configured bridge domain.

A PE device receiving multicast traffic to be forwarded is referred to as the *ingress PE device*. To ensure multicast traffic reaches all remote PE devices in the EVI, the ingress PE device uses the IMET routing information with ingress replication in the EVPN core, replicating and flooding the packets on the EVPN tunnels to all of the other PE devices (or external edge routers) in the EVI that might need to forward the traffic. IGMP or MLD snooping is not performed in the EVPN core.

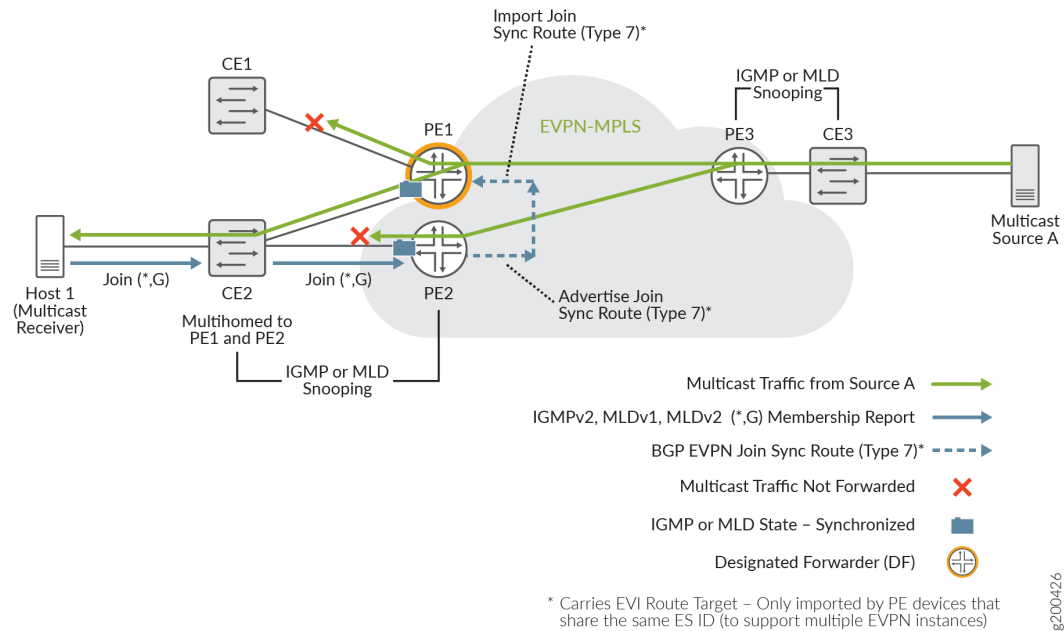
NOTE: When the multicast source is outside the EVPN-MPLS network in an external PIM domain (as it is in the implementation for ACX Series routers), each PE device receives the source traffic directly from the configured Layer 3 interfaces to the PIM gateway router, and not by way of IMET signaling in the EVPN core.

With IGMP or MLD snooping enabled:

- When a multihoming PE device receives multicast traffic and it is the DF for an interested receiver for the multicast group, the PE device forwards the traffic.
- When a multihoming PE device receives multicast traffic and it is not the DF for any interested receivers, the PE device does not forward the traffic.
- On the access side, upon receiving multicast data from the EVPN core, PE devices selectively forward the multicast traffic only to interested receivers. Single-homing PE devices use learned IGMP or MLD snooping information, and multihoming PE devices use both IGMP or MLD snooping information and EVPN Type 7 routes.

For example, let's look again at the use case in [Figure 105 on page 1023](#) with sources and receivers all inside the EVPN-MPLS network.

Figure 105: Multicast Traffic Flow with Sources Inside the EVPN-MPLS Network

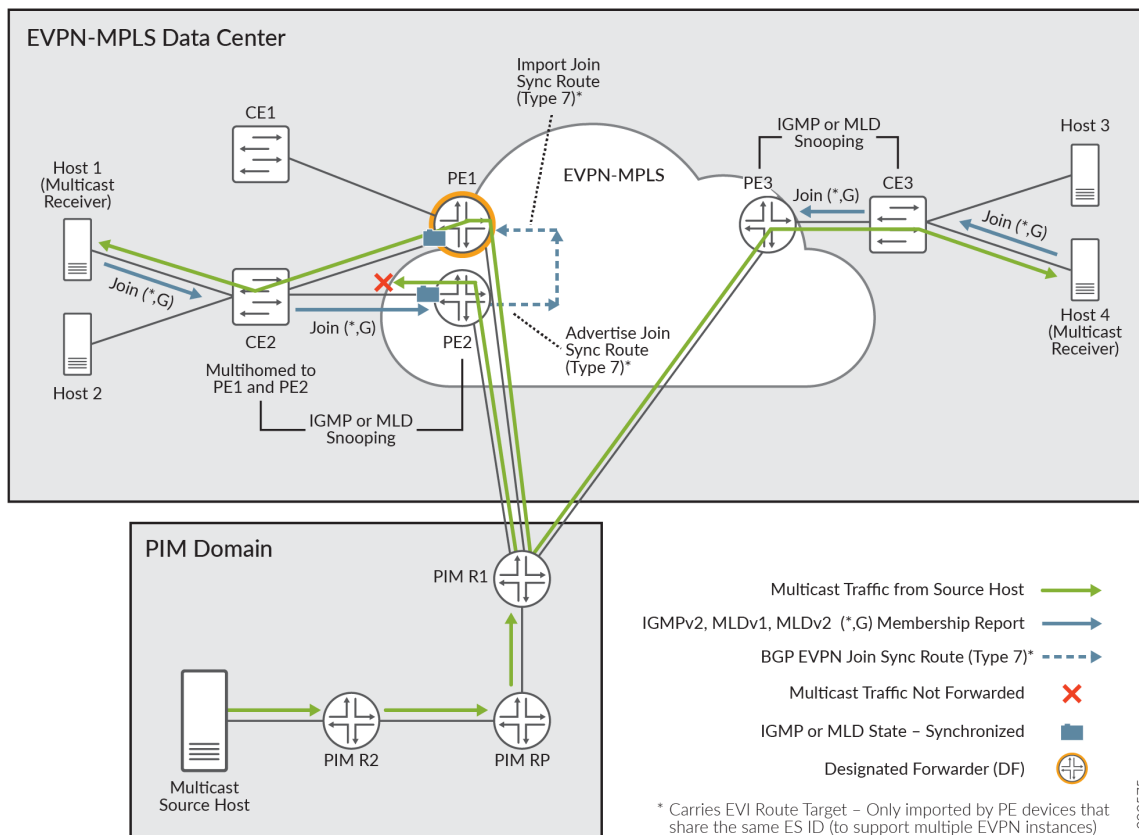


PE3 is the ingress PE device for Multicast Source A and forwards the multicast traffic into the EVPN core. PE1 and PE2 receive the multicast traffic from the EVPN core. Host 1 is a multicast receiver that is multihomed to PE1 and PE2 by way of customer edge device CE2. PE1 and PE2 have synchronized IGMP or MLD state information about receiver Host 1. The PEs forward the traffic as follows:

- Because PE1 is currently the elected DF for Host 1, PE1 forwards the data toward Host 1.
- Based on the IGMP or MLD snooping information and imported EVPN Type 7 routing information, PE1 doesn't forward the data toward CE1 because CE1 doesn't have any interested receivers.
- PE2 is not the DF, so PE2 doesn't forward the traffic toward Host 1, and avoids sending a duplicate stream.

Also consider the use case in [Figure 106 on page 1024](#) for ACX Series routers with EVPN-MPLS multihoming that supports multicast across bridge domains or VLANs, where the sources must be outside the EVPN network in an external PIM domain.

Figure 106: Multicast Traffic Flow With Sources Outside the EVPN-MPLS Network



PE1, PE2, and PE3 all have Layer 3 connections to the PIM gateway router PIM R1, and all PE devices are ingress PEs. In this case, all PEs receive the multicast source traffic from the source through PIM R1. Host 3 and Host 4 are single-homed to PE3 by way of CE3. Host 1 is a multicast receiver that is multihomed to PE1 and PE2 through CE2. The PEs forward the traffic as follows:

- Host 4 is a multicast receiver, so PE3 sends the traffic to CE3 to reach Host 4.
- Host 3 isn't a multicast receiver, so with snooping enabled, CE3 doesn't forward the traffic to Host 3.
- PE1 and PE2 have synchronized IGMP or MLD state information about receiver Host 1, and as peer PE devices, they behave in the same way as the use case with all sources and receivers inside the EVPN-MPLS network:

- As the elected DF for Host 1, PE1 forwards the data to Host 1. PE2 is not the DF and doesn't send a duplicate stream to Host 1.
- Based on the IGMP or MLD snooping information and imported EVPN Type 7 routing information, PE1 doesn't forward the data toward CE1 because CE1 doesn't have any interested receivers.

IGMP or MLD Snooping with Multicast Forwarding Between Bridge Domains or VLANs Using IRB Interfaces

For multicast forwarding between bridge domains or VLANs in this environment, PE devices use Protocol Independent Multicast (PIM) in distributed designated router (DDR) mode on IRB interfaces.

- You must configure IRB interfaces on all PE devices in the EVI that need to route multicast traffic locally between bridge domains or VLANs on the device.

NOTE: If you configured a bridge domain (or VLAN) and an IRB interface on all PE devices, and an IRB interface is down on a PE device, traffic routing between bridge domains (or VLANs) for receivers served by that IRB will be impacted.

- To route Layer 3 multicast traffic between bridge domains or VLANs, you must also configure PIM distributed designated router (DDR) mode on all participating IRB interfaces. (See [distributed-dr](#) for more details.) PIM passive mode is not supported.

The IRB interfaces on the PE devices route multicast traffic between bridge domains or VLANs as follows:

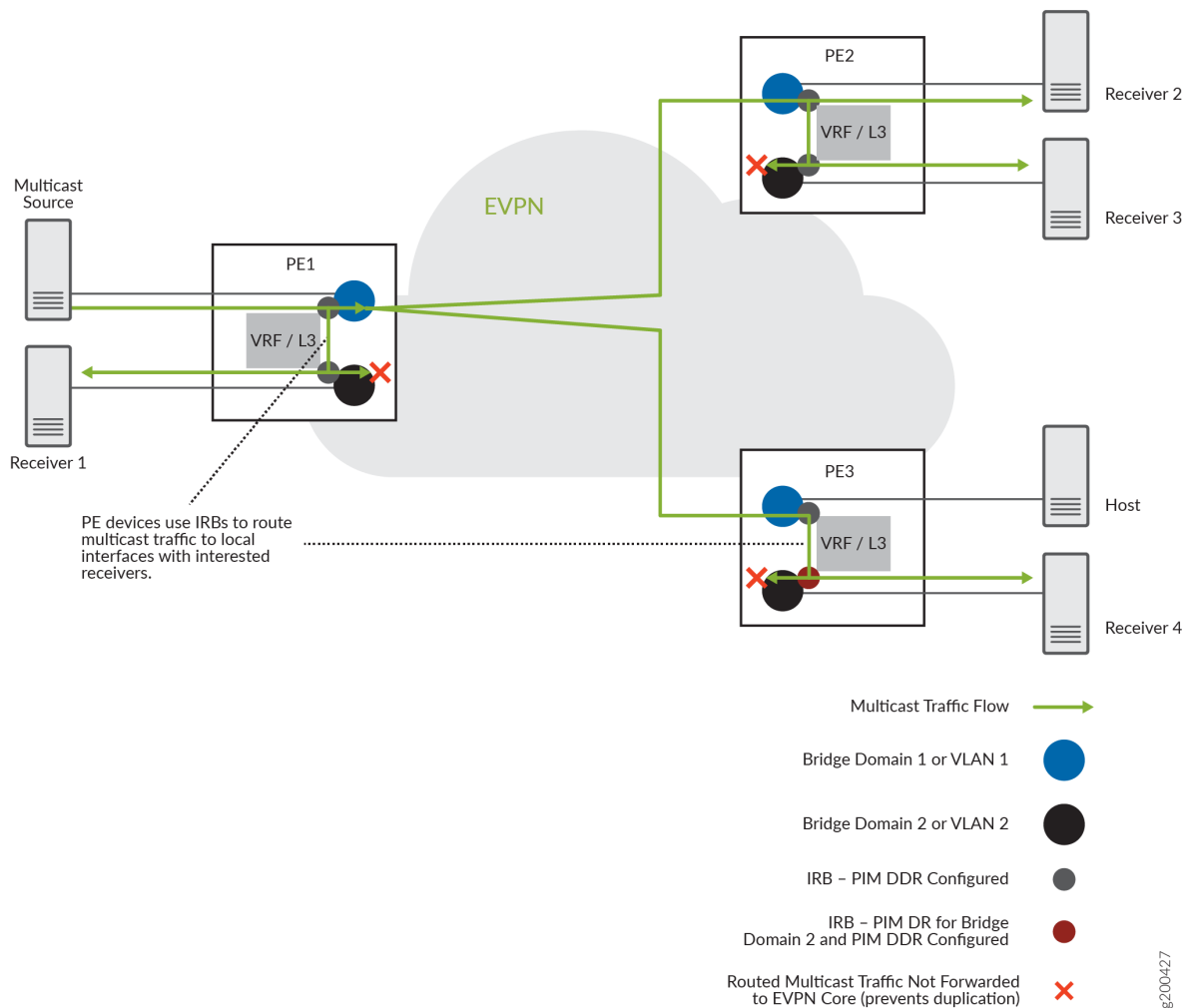
- Upon receiving multicast traffic on an IRB interface from a multicast source, the PE device routes the traffic to any IRBs that have PIM enabled and are configured with bridge domains or VLANs that have interested local receivers for the multicast group.
- In PIM DDR mode, the IRB interfaces route multicast traffic to local receivers whether or not the IRB is the elected PIM designated router (DR).
- To prevent multicast traffic duplication, IRB-routed multicast traffic is not forwarded back out to the EVPN core.

The IRB interfaces route ingress multicast traffic between bridge domains or VLANs toward receivers in the same way whether the multicast source is inside the EVPN-MPLS network or is in an external PIM domain.

For example, [Figure 107 on page 1026](#) shows PE devices with IRB interfaces where the PE devices receive source traffic through the EVPN-MPLS network.

NOTE: ACX Series routers support do not support having multicast sources within the EVPN-MPLS network for routing inter-VLAN multicast traffic. See [Figure 108 on page 1027](#) instead.

Figure 107: Routing Multicast Traffic Between Bridge Domains (or VLANs) with IRBs and PIM DDR in EVPN-MPLS

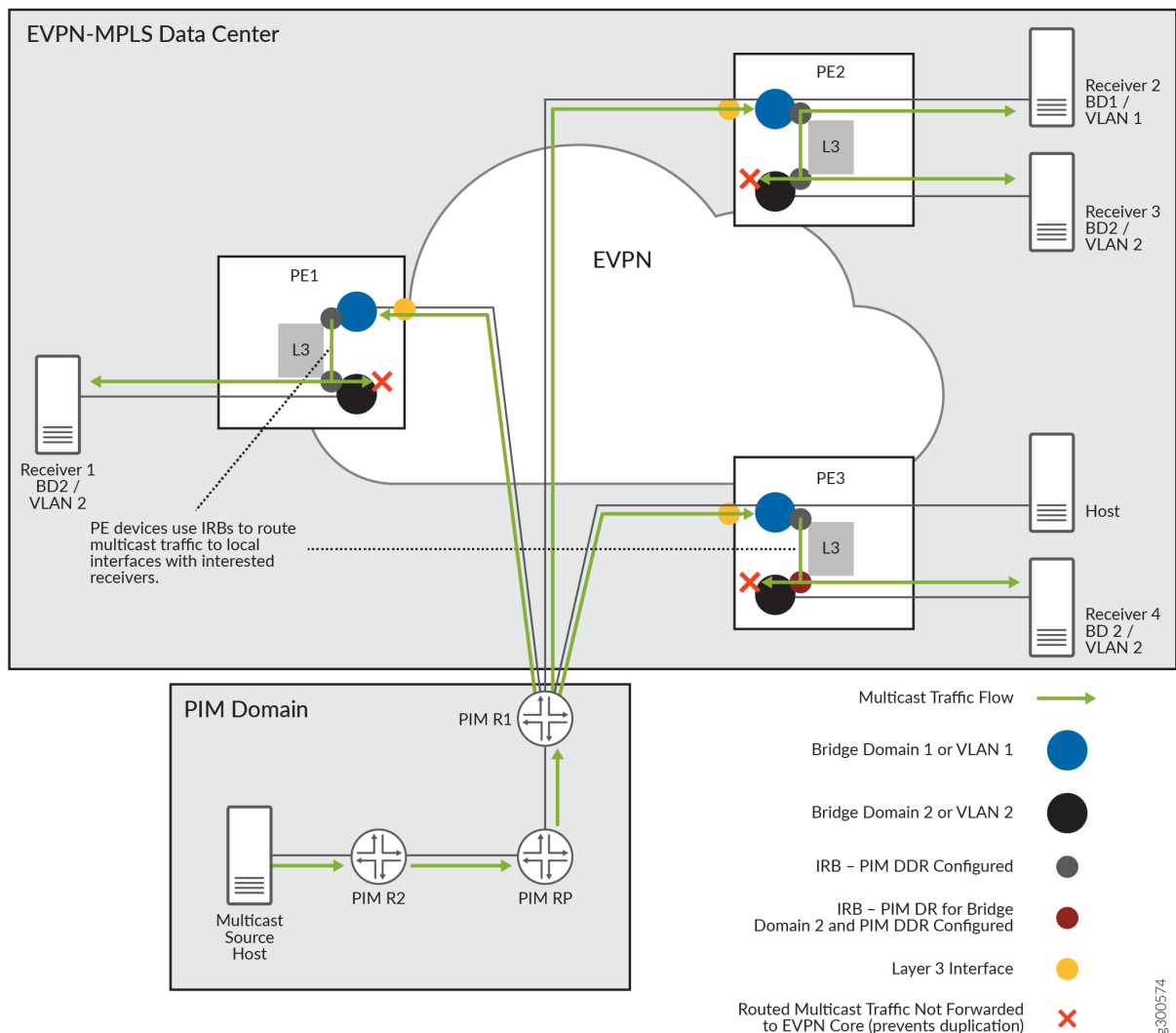


In this use case, the EVPN instance has three PE devices, PE1, PE2, and PE3, with IRB interfaces configured on each PE device for two bridge domains or VLANs. The multicast source on bridge domain 1 (VLAN 1) sends multicast traffic to PE1, and PE1 uses inclusive multicast forwarding with ingress replication to forward the traffic to the EVPN core to reach other PE devices with interested receivers. PE1 routes the traffic locally to interested receivers on bridge domain 2 (VLAN 2) through the IRB interfaces, even though PE3 is the PIM DR for bridge domain 2 (VLAN 2). The IRB interfaces on all PE

devices that route traffic to receivers on bridge domain 2 (VLAN 2) do not forward the routed traffic back out into the EVPN core. PE2 receives the multicast traffic from the EVPN core and forwards or locally routes the traffic on each bridge domain (or VLAN) that has interested receivers. PE3 routes the traffic locally to bridge domain 2 only, because there are no interested receivers on bridge domain 1 (or VLAN 1).

Figure 108 on page 1027 shows the PE devices similarly configured with IRB interfaces in DDR mode, but each of the PE devices receives the multicast traffic from an external source through Layer 3 interfaces connected to the PIM gateway router. This is the use case supported on ACX Series routers for routing inter-VLAN multicast traffic in an EVPN-MPLS network.

Figure 108: Routing External PIM Domain Source Traffic Between Bridge Domains (or VLANs) with IRBs and PIM DDR in EVPN-MPLS



In this use case, PE1, PE2, and PE3 also have IRB interfaces configured on each PE device for two bridge domains or VLANs. Each PE might have interested receivers, so each one has a Layer 3 connection to the source PIM domain and receives the multicast traffic on BD1 (VLAN 1). PE1 and PE3 each route the traffic locally through the IRB interfaces to their receivers (Receiver 1 and Receiver 4) on BD2 (VLAN 2), and PE2 does the same for its Receiver 3 on BD2, even though PE3 is the PIM DR for BD2 (VLAN 2). PE2 also forwards the traffic locally to Receiver 2 on BD1 (VLAN 1). The PE devices don't send source traffic out into the EVPN core at all in this model.

RELATED DOCUMENTATION

[Configure Multicast Forwarding with IGMP Snooping in an EVPN-MPLS Environment | 1028](#)

[Configure Multicast Forwarding with MLD Snooping in an EVPN-MPLS Environment | 1038](#)

[EVPN Multihoming Overview | 96](#)

Configure Multicast Forwarding with IGMP Snooping in an EVPN-MPLS Environment

IN THIS SECTION

- [Configure IGMP Snooping for an EVPN or Virtual Switch Routing Instance | 1029](#)
- [Configure IGMP Snooping with IGMPv3 on ACX Series Routers to Process Source-Specific Multicast Group Membership Reports Only | 1031](#)
- [Configure Multicast Routing Across Bridge Domains or VLANs with PIM in EVPN-MPLS | 1032](#)
- [Viewing IGMP Snooping Multicast Information for EVPN-MPLS in the CLI | 1035](#)

IGMP snooping with multicast forwarding ensures that IPv4 multicast traffic reaches all subscribed receivers within and between bridge domains or VLANs, and preserves bandwidth on the access side by reducing the amount of multicast control and data traffic being forwarded.

You can configure multicast with IGMP snooping on provider edge (PE) devices in an Ethernet VPN (EVPN) over MPLS environment. You must enable IGMP snooping in this environment to support IPv4 multicast traffic for receivers that are multihomed to EVPN PE devices as follows:

- Peer PE devices must be operating in all-active multihoming mode.
- You enable IGMP snooping in proxy mode.

- You configure and commit the same configuration on each multihoming peer PE device.

NOTE: MX Series routers and EX9200 switches in this environment only support IGMPv2 with any-source multicast (ASM) membership reports. ACX Series routers in this environment support processing IGMPv2 and IGMPv3 membership reports in ASM mode and IGMPv3 membership reports in source-specific multicast (SSM) mode with a special configuration option. See ["IGMP or MLD Versions and Supported Group Membership Report Modes" on page 1020](#) for details on how PEs process IGMP reports.

In addition, to route multicast traffic between bridge domains or VLANs, PEs with hosts in multicast groups that span bridge domains or VLANs use PIM distributed designated router (PIM DDR) mode on IRB interfaces. You set up IRB interfaces associated with each bridge domain or VLAN with interested receivers in a multicast group. With PIM DDR configured on the IRB interfaces, the PE devices send the traffic locally through the IRB interfaces to their interested receivers on the corresponding bridge domains or VLANs even if an IRB is not the elected PIM designated router (DR) for that bridge domain or VLAN.

This topic describes configuration tasks to set up IGMP snooping in proxy mode, and configure the IRB interfaces for routing between bridge domains or VLANs on EVPN PE devices. It also summarizes the CLI commands you can use to verify IGMP snooping and multicast operation in this environment.

Configure IGMP Snooping for an EVPN or Virtual Switch Routing Instance

On ACX Series routers, you can configure IGMP snooping on one or more routing instances of type `evpn`. On other types of devices, you can configure IGMP snooping for routing instances of type `evpn` or `virtual-switch`.

- To configure IGMP snooping in proxy mode on a PE device for a routing instance with `instance-type evpn` (for any or all bridge domains or VLANs):

```
user@device# set routing-instances routing-instance-name protocols igmp-snooping proxy
```

For example, the following configuration includes enabling IGMP snooping in proxy mode for EVPN routing instance `evpn-A` configured as `instance-type evpn`:

```
routing-instances {
  evpn-A {
    instance-type evpn;
    vlan-id 600
    interface ge-0/0/2.600;
    route-distinguisher 10.255.255.1:100;
```



```

    vrf-target target:64510:100;
    protocols {
        evpn {
            interface ge-0/0/2.600;
            label-allocation per-instance;
        }
        igmp-snooping proxy;
    }
}

```

- To configure IGMP snooping on a PE device for specific bridge domains or VLANs for an EVPN instance with instance-type virtual-switch:

```

user@device# set routing-instances routing-instance-name bridge-domain bridge-domain-name
protocols igmp-snooping proxy

```

For example, the following configuration includes enabling IGMP snooping in proxy mode for bridge domains V200, V201, and V202 in EVPN routing instance EVPN-2, which is configured as instance-type virtual-switch:

```

routing-instances {
    ...
    EVPN-2 {
        instance-type virtual-switch;
        route-distinguisher 90.90.90.10:2;
        vrf-target target:64510:2;
        protocols {
            evpn {
                encapsulation mpls;
                extended-vlan-list 200-202;
                default-gateway advertise;
            }
        }
        bridge-domains {
            V200 {
                domain-type bridge;
                vlan-id 200;
                interface ae1.200;
                routing-interface irb.200;
                protocols {

```

```

        igmp-snooping proxy;
    }
}
V201 {
    domain-type bridge;
    vlan-id 201;
    interface ae1.201;
    routing-interface irb.201;
    protocols {
        igmp-snooping proxy;
    }
}
V202 {
    domain-type bridge;
    vlan-id 202;
    interface ae1.202;
    routing-interface irb.202;
    protocols {
        igmp-snooping proxy;
    }
}
}
...
}

```

Configure IGMP Snooping with IGMPv3 on ACX Series Routers to Process Source-Specific Multicast Group Membership Reports Only

By default, the EVPN-MPLS network processes only ASM (*,G) membership reports with IGMPv2. ACX Series PE routers also support processing IGMPv3 membership reports in ASM (*,G) mode. You can alternatively configure ACX Series PEs to process only IGMPv3 SSM (S,G) membership reports in an EVPN-MPLS network that supports intersubnet multicast. To do this, use the [evpn-ssm-reports-only](#) option in the [edit protocols [igmp-snooping](#)] configuration statement hierarchy when you configure IGMP snooping.

When you enable this option, the device doesn't process ASM reports and drops those packets. See ["IGMP or MLD Versions and Supported Group Membership Report Modes" on page 1020](#) for details on how PEs process IGMP membership reports in this environment.

You can enable SSM-only processing when you configure IGMP snooping with IGMPv3 for all bridge domains or VLANs in a routing instance of type `evpn`.

For example, to configure IGMP snooping with IGMPv3 to process only SSM membership reports for all bridge domains or VLANs in a routing instance:

```
user@device# set routing-instances routing-instance-name protocols igmp-snooping evpn-ssm-
reports-only
```

In the following sample configuration, IGMP snooping is configured to process only SSM membership reports for a routing instance of type `evpn` named `EVPN-4`:

```
.
.
.
    EVPN-4 {
        instance-type evpn;
        protocols {
            evpn {
                remote-ip-host-routes;
                designated-forwarder-preference-least;
            }
            igmp-snooping {
                evpn-ssm-reports-only;
                proxy;
            }
        }
        vlan-id 400;
        interface ae2.400;
        l3-interface irb.400;
        route-distinguisher 90.90.90.10:2;
        vrf-target target:64510:4;
    }
.
.
.
```

Configure Multicast Routing Across Bridge Domains or VLANs with PIM in EVPN-MPLS

PE devices in EVPN-MPLS environments use IRB interfaces and PIM to route multicast traffic between bridge domains or VLANs (intersubnet multicast).

On ACX Series routers, any multicast sources must be outside of the EVPN-MPLS data center in a Layer 3 PIM domain. Each ACX Series PE device receives the multicast source traffic through a Layer 3 interface connected to an external PIM domain gateway router. The receivers must be within the EVPN-MPLS datacenter.

With other platforms, all multicast sources and receivers must be within the EVPN-MPLS data center, and each PE device either receives the multicast traffic locally or through the EVPN core.

In either case (multicast sources outside or inside the EVPN instance), you enable PIM in distributed DR mode on the IRB interfaces in an EVPN instance, and the IRB interfaces send the multicast group traffic to any receivers on the associated bridge domains or VLANs.

- To configure IRB interfaces to use PIM DDR mode to route multicast traffic between bridge domains or VLANs in an EVPN routing instance:

```
user@device# set routing-instances routing-instance-name protocols pim interface irb-
interface-name distributed-dr
```

For example, the following configuration shows PIM DDR mode enabled on interfaces irb.400, irb.401, and irb.402 in an instance-type evpn routing instance named evpn-400 on ACX Series routers (which require a Layer 3 interface on which to receive source traffic from an external PIM domain):

```
routing-instances {
  evpn-400 {
    instance-type evpn;
    vlan-id 400;
    interface xe-0/0/32.400;
    l3-interface irb.400;
    route-distinguisher 10.255.65.227:400;
    vrf-target target:64510:400;
    protocols {
      evpn {
        interface xe-0/0/32.400;
      }
      igmp-snooping proxy;
    }
  }
  protocols {
    pim {
      rp {
        static {
          address 10.255.67.48;
        }
      }
    }
  }
}
```

```

    }
  }
  interface lo0.0;
  interface all {
    mode {
      sparse;
    }
  }
  ...
  interface irb.400 {
    distributed-dr;
  }
  interface irb.401 {
    distributed-dr;
  }
  interface irb.402 {
    distributed-dr;
  }
}
}
...
}

```

The following configuration example shows PIM DDR mode enabled on interfaces irb.200, irb.201, and irb.202 in the VRF routing instance IPVPN-2:

```

routing-instances {
  ...
  IPVPN-2 {
    instance-type vrf;
    interface irb.200;
    interface irb.201;
    interface irb.202;
    interface lo0.2;
    route-distinguisher 90.90.90.10:222;
    vrf-target target:64510:2;
    vrf-table-label;
    protocols {
      pim {
        rp {
          local {

```

```

        address 10.88.88.10;
    }
}
interface irb.200 {
    distributed-dr;
}
interface irb.201 {
    distributed-dr;
}
interface irb.202 {
    distributed-dr;
}
}
}
}

```

Viewing IGMP Snooping Multicast Information for EVPN-MPLS in the CLI

The following EVPN commands are supported to view IGMP snooping multicast information in an EVPN-MPLS environment. The output of these commands includes information learned from native IGMP snooping on a PE device and learned from EVPN Type 7 Join Sync Route and Type 8 Leave Sync Route messages.

- `show evpn igmp-snooping database`
- `show evpn multicast-snooping next-hops`
- `show igmp snooping evpn database`
- `show igmp snooping evpn membership`

The following CLI commands are supported to view the native IGMP snooping information on a PE. The output of these commands will not include information learned from the exchange of EVPN Type 7 and Type 8 IGMP messages.

- `show igmp snooping interface`
- `show igmp snooping membership`
- `show igmp snooping statistics`

The following commands can be used to view EVPN Type 7 IGMP Join Sync Routes or Type 8 Leave Sync Routes in BGP and EVPN routing tables:

- `show route table bgp.evpn.0 match-prefix 7*`

```
show route table __default_evpn__.evpn.0 match-prefix 7:*

show route table __default_evpn__.evpn.0 match-prefix 7:* protocol evpn

show route table __default_evpn__.evpn.0 match-prefix 7:* protocol bgp

• show route table bgp.evpn.0 match-prefix 8:*

show route table __default_evpn__.evpn.0 match-prefix 8:*

show route table __default_evpn__.evpn.0 match-prefix 8:* protocol evpn

show route table __default_evpn__.evpn.0 match-prefix 8:* protocol bgp
```

For example, to view EVPN Type 7 IGMP routes in the __default_evpn__.evpn.0 routing table, use the following command:

```
user@host> show route table __default_evpn__.evpn.0
protocol evpn match-prefix 7:* extensive
Sep 21 02:25:12
__default_evpn__.evpn.0: 32 destinations, 32 routes (32 active, 0 holddown, 0 hidden)
7:90.90.90.10:1::1111111111111111::100::0.0.0.0::233.252.0.1::90.90.90.10/600 (1 entry, 1
announced)
TSI:
Page 0 idx 0, (group IBGP type Internal) Type 1 val 0xb23c68c (adv_entry)
  Advertised metrics:
    Nexthop: 90.90.90.10
    Localpref: 100
    AS path: [64510] I
    Communities: es-import-target:11-11-11-11-11-11 evi-rt:64510:1 Path
7:90.90.90.10:1::1111111111111111::100::0.0.0.0::233.252.0.1::90.90.90.10 Vector len 4. Val: 0
  *EVPN Preference: 170
    Next hop type: Indirect, Next hop index: 0
    Address: 0xb2e5e10
    Next-hop reference count: 61
    Protocol next hop: 90.90.90.10
    Indirect next hop: 0x0 - INH Session ID: 0x0
    State: <Active Int Ext>
    Age: 2:12:30
    Validation State: unverified
    Task: __default_evpn__-evpn
    Announcement bits (1): 1-BGP_RT_Background
    AS path: I
    Communities: es-import-target:11-11-11-11-11-11 evi-rt:65530:1
    IGMP flags: 0x2
```

```

7:90.90.90.10:3::333333333333333333::100::0.0.0.0::233.252.0.1::90.90.90.10/600 (1 entry, 1
announced)
TSI:
Page 0 idx 0, (group IBGP type Internal) Type 1 val 0xb23c6a8 (adv_entry)
  Advertised metrics:
    Nexthop: 90.90.90.10
    Localpref: 100
    AS path: [64510] I
    Communities: es-import-target:33-33-33-33-33-33 evi-rt:64510:3 Path
7:90.90.90.10:3::333333333333333333::100::0.0.0.0::233.252.0.1::90.90.90.10 Vector len 4. Val: 0
  *EVPN Preference: 170
    Next hop type: Indirect, Next hop index: 0
    Address: 0xb2e5e10
    Next-hop reference count: 61
    Protocol next hop: 90.90.90.10
    Indirect next hop: 0x0 - INH Session ID: 0x0
    State: <Active Int Ext>
    Age: 2:12:30
    Validation State: unverified
    Task: __default_evpn__-evpn
    Announcement bits (1): 1-BGP_RT_Background
    AS path: I
    Communities: es-import-target:33-33-33-33-33-33 evi-rt:65530:3
    IGMP flags: 0x2

```

RELATED DOCUMENTATION

Overview of Multicast Forwarding with IGMP or MLD Snooping in an EVPN-MPLS Environment | 1013

Configure Multicast Forwarding with MLD Snooping in an EVPN-MPLS Environment

IN THIS SECTION

- [Configure MLD Snooping for Default Any-Source Multicast \(ASM\) Group Membership Processing with MLDv1 or MLDv2 | 1039](#)
- [Configure MLD Snooping with MLDv2 to Process Source-Specific Multicast Group Membership Reports Only | 1042](#)
- [Viewing MLD Snooping Multicast Information for EVPN-MPLS in the CLI | 1043](#)

MLD snooping with multicast forwarding ensures that IPv6 multicast traffic reaches all subscribed receivers within and between bridge domains or VLANs, and preserves bandwidth on the access side by reducing the amount of multicast control and data traffic being forwarded.

You can configure multicast with MLD snooping on provider edge (PE) devices in an Ethernet VPN (EVPN) over MPLS environment. You must enable MLD snooping in this environment to support IPv6 multicast traffic for receivers that are multihomed to EVPN PE devices as follows:

- Peer PE devices must be operating in all-active multihoming mode.
- You enable MLD snooping in proxy mode.
- You configure and commit the same configuration on each multihoming peer PE device.

NOTE: PEs in this environment support processing MLDv1 and MLDv2 membership reports in any-source multicast (ASM) mode and MLDv2 membership reports in source-specific multicast (SSM) mode with a special configuration option. See ["IGMP or MLD Versions and Supported Group Membership Report Modes" on page 1020](#) for details on how PEs process MLD reports.

In addition, to route multicast traffic between bridge domains or VLANs, PEs with hosts in multicast groups that span bridge domains or VLANs use PIM distributed designated router (PIM DDR) mode on IRB interfaces. With PIM DDR configured on the IRB interfaces, all the PE devices send the traffic locally through the IRB interfaces to their interested receivers on the corresponding bridge domains or VLANs even if an IRB is not the elected PIM designated router (DR) for that bridge domain or VLAN. See ["Configure Multicast Routing Across Bridge Domains or VLANs with PIM in EVPN-MPLS" on page 1032](#) for details.

This topic describes configuration tasks to set up MLD snooping on PE devices in a multihoming EVPN-MPLS environment with either the default ASM group membership report processing (MLDv1 and MLDv2) or SSM-only group membership report processing (MLDv2). This topic also summarizes the CLI commands you can use to verify MLD snooping and multicast operation in this environment.

Configure MLD Snooping for Default Any-Source Multicast (ASM) Group Membership Processing with MLDv1 or MLDv2

By default, the EVPN-MPLS network processes only ASM (*,G) membership reports with MLDv1 and MLDv2.

On ACX Series routers, you can configure MLD snooping only on one or more routing instances of type `evpn`. On other types of devices, you can configure MLD snooping with ASM for routing instances of type `evpn` or `virtual-switch` and for all bridge domains or VLANs or only for a specific bridge domain or VLAN.

For example:

- To configure MLD snooping in proxy mode on PE devices in an EVPN-MPLS network for the default-switch routing instance:

```
user@device# set protocols mld-snooping proxy
```

- To configure MLD snooping in proxy mode on a PE device for a particular routing instance configured as `instance-type evpn` or `instance-type virtual-switch` for all bridge domains or VLANs in the instance:

```
user@device# set routing-instances routing-instance-name protocols mld-snooping proxy
```

- To configure MLD snooping on a PE device for a specific bridge domain or VLAN for an EVPN instance with `instance-type virtual-switch`:

```
user@device# set routing-instances routing-instance-name bridge-domain bridge-domain-name
protocols mld-snooping proxy
```

PE devices can't process both ASM (*,G) reports and SSM (S,G) reports at the same time, but you can alternatively configure PEs to process only SSM (S,G) membership reports. See ["Configure MLD Snooping with MLDv2 to Process Source-Specific Multicast Group Membership Reports Only"](#) on page 1042.

In the following sample configuration, MLD snooping is enabled in proxy mode (and the same for IGMP snooping) for an instance of type evpn named EVPN-1:

```
.
.
.
    EVPN-1 {
        instance-type evpn;
        protocols {
            evpn {
                remote-ip-host-routes;
                designated-forwarder-preference-least;
            }
            igmp-snooping {
                proxy;
            }
            mld-snooping {
                proxy;
            }
        }
        vlan-id 100;
        interface ae1.100;
        l3-interface irb.100;
        route-distinguisher 90.90.90.10:1;
        vrf-target target:64510:1;
    }
.
.
.
```

In following sample configuration, MLD snooping is enabled for three bridge domains (V100, V200, V300) in a routing instance of type virtual-switch named CUST-2:

```
.
.
.
    CUST-2 {
        instance-type virtual-switch;
```

```

route-distinguisher 10.255.255.1:100;
vrf-target target:64510:100;
protocols {
    evpn {
        extended-vlan-list [ 100 200 300 ];
    }
}
bridge-domains {
    V100 {
        domain-type bridge;
        vlan-id 100;
        interface ae0.100;
        routing-interface irb.100;
        protocols {
            igmp-snooping;
            mld-snooping;
        }
    }
    V200 {
        domain-type bridge;
        vlan-id 200;
        interface ae0.200;
        routing-interface irb.200;
        protocols {
            igmp-snooping;
            mld-snooping;
        }
    }
    V300 {
        domain-type bridge;
        vlan-id 300;
        interface ge-0/0/5.300;
        routing-interface irb.300;
        protocols {
            igmp-snooping;
            mld-snooping;
        }
    }
}
}

```

.

.

Configure MLD Snooping with MLDv2 to Process Source-Specific Multicast Group Membership Reports Only

By default, the EVPN-MPLS network processes only ASM (*,G) membership reports with MLDv1 and MLDv2. You can alternatively configure PEs to process only MLDv2 SSM (S,G) membership reports. To do this, use the [evpn-ssm-reports-only](#) option in the [edit protocols [mld-snooping](#)] configuration statement hierarchy when you configure MLD snooping.

When you enable this option, the device doesn't process ASM reports and drops those packets. See ["IGMP or MLD Versions and Supported Group Membership Report Modes" on page 1020](#) for details on how PEs process MLD reports in this environment.

You can enable SSM-only processing when you configure MLD snooping for:

- All bridge domains or VLANs in a routing instance of type `evpn`.
- All bridge domains or a specific bridge domain or VLAN in a `virtual-switch` instance.

NOTE: On ACX Series routers, you can configure MLD snooping only on one or more routing instances of type `evpn`. On other devices, you can configure MLD snooping for routing instances of type `evpn` or `virtual-switch`.

For example:

- To configure MLD snooping to process only SSM membership reports with MLDv2 for the default-switch EVPN routing instance:

```
user@device# set protocols mld-snooping evpn-ssm-reports-only
```

- To configure MLD snooping with MLDv2 to process only SSM membership reports for all bridge domains or VLANs in a routing instance:

```
user@device# set routing-instances routing-instance-name protocols mld-snooping evpn-ssm-reports-only
```

- To configure MLD snooping in proxy mode to process only SSM membership reports for a specific bridge domain or VLAN for an EVPN instance with instance-type virtual-switch:

```
user@device# set routing-instances routing-instance-name bridge-domain bridge-domain-name
protocols mld-snooping evpn-ssm-reports-only proxy
```

In following sample configuration, MLD snooping is configured to process only SSM membership reports for a routing instance of type evpn named EVPN-2:

```
.
.
.
    EVPN-2 {
        instance-type evpn;
        protocols {
            evpn {
                remote-ip-host-routes;
                designated-forwarder-preference-least;
            }
            mld-snooping {
                evpn-ssm-reports-only;
                proxy;
            }
        }
        vlan-id 200;
        interface ae2.200;
        l3-interface irb.200;
        route-distinguisher 90.90.90.10:2;
        vrf-target target:64510:1;
    }
.
.
.
```

Viewing MLD Snooping Multicast Information for EVPN-MPLS in the CLI

The following EVPN commands are supported to view MLD snooping multicast information in an EVPN-MPLS environment. The output of these commands includes information learned from native MLD snooping on a PE device and learned from EVPN Type 7 Join Sync Route and Type 8 Leave Sync Route messages.

- [show evpn mld-snooping database](#)
- [show evpn multicast-snooping next-hops](#)
- [show mld snooping evpn database](#)
- [show mld snooping evpn membership](#)

RELATED DOCUMENTATION

[Overview of Multicast Forwarding with IGMP or MLD Snooping in an EVPN-MPLS Environment | 1013](#)

4

PART

EVPN-VPWS

Configuring VPWS Service with EVPN Mechanisms | 1046

Configuring VPWS Service with EVPN Mechanisms

IN THIS CHAPTER

- Overview of VPWS with EVPN Signaling Mechanisms | 1046
- Control word for EVPN-VPWS | 1050
- Overview of Flexible Cross-Connect Support on VPWS with EVPN | 1054
- Overview of Headend Termination for EVPN VPWS for Business Services | 1060
- Configuring VPWS with EVPN Signaling Mechanisms | 1068
- Example: Configuring VPWS with EVPN Signaling Mechanisms | 1070
- FAT Flow Labels in EVPN-VPWS Routing Instances | 1096

Overview of VPWS with EVPN Signaling Mechanisms

An Ethernet VPN (EVPN) enables you to connect dispersed customer sites using a Layer 2 virtual bridge. As compared with other types of Layer 2 VPNs, an EVPN consists of customer edge (CE) devices (host, router, or switch) connected to provider edge (PE) routers. The PE routers can include an MPLS edge switch (MES) that acts at the edge of the MPLS infrastructure. Either an MX Series 5G Universal Routing Platform or a standalone switch can be configured to act as an MES. You can deploy multiple EVPNs within a service provider network, each providing network connectivity to a customer while ensuring that the traffic sharing on that network remains private.

Virtual private wire service (VPWS) Layer 2 VPNs employ Layer 2 services over MPLS to build a topology of point-to-point connections that connect end customer sites in a VPN. The service provisioned with these Layer 2 VPNs is known as VPWS. You can configure a VPWS instance on each associated edge device for each VPWS Layer 2 VPN.

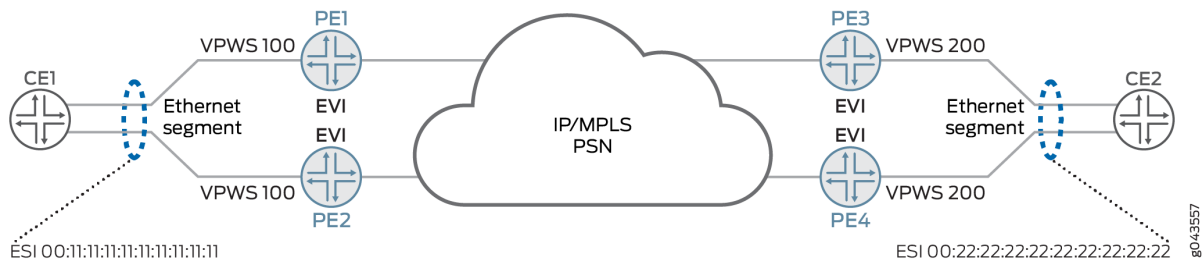
An EVPN-VPWS network provides a framework for delivering VPWS with EVPN signaling mechanisms. The advantages of VPWS with EVPN mechanisms are single-active or all-active multihoming capabilities and support for Inter-autonomous system (AS) options associated with BGP-signaled VPNs. Metro Ethernet Forum (MEF) describes the following two service models for VPWS:

- Ethernet private line (EPL)— EPL provides a point-to-point Ethernet virtual connection (EVC) between a pair of dedicated user-network interfaces (UNIs) that is between a pair of Ethernet segment identifiers (ESIs) with a high degree of transparency.
- Ethernet virtual private line (EVPL)— EVPL provides point-to-point EVC between {ESI, VLAN} pairs. EVPL allows service multiplexing; that is multiple EVCs or Ethernet services per UNI.

An EVPN-VPWS network is supported on an EVPN-MPLS network.

Figure 109 on page 1047 illustrates an EVPN-VPWS network. Device CE1 is multihomed to Routers PE1 and PE2 and Device CE2 is multihomed to Routers PE3 and PE4. Device CE2 has two potential paths to reach CE1, and depending on the multihoming mode of redundancy, only one path or all paths can be active at any one time. When a CE device is multihomed to two or more PE routers, the set of Ethernet links constitutes an Ethernet segment. An Ethernet segment appears as a link aggregation group (LAG) to the CE device. The links from PE1 and PE2 to CE1 and PE3 and PE4 to CE2 form an Ethernet segment.

Figure 109: VPWS in EVPN



An Ethernet segment must have a unique nonzero identifier, called the *Ethernet segment identifier (ESI)*. The ESI is encoded as a 10-octet integer. When manually configuring an ESI value, the most significant octet, known as the *type byte*, must be 00. When a single-homed CE device is attached to an Ethernet segment, the entire ESI value is zero. The Ethernet segment of the multihomed Device CE1 has an ESI value of 00:11:11:11:11:11:11:11:11:11 and the Ethernet segment of the multihomed Device CE2 has an ESI value of 00:22:22:22:22:22:22:22:22:22.

An EVPN instance (EVI) is an EVPN routing and forwarding instance spanning across all the PE routers participating in that VPN. An EVI is configured on the PE routers on a per-customer basis. Each EVI has a unique route distinguisher and one or more route targets. An EVI is configured on PE1, PE2, PE3, and PE4. An Ethernet tag identifies a particular broadcast domain, such as a VLAN. An EVI consists of one or more broadcast domains. Ethernet tags are assigned to the broadcast domains of a given EVI by the provider of that EVPN. Each PE router in that EVI performs a mapping between broadcast domain identifiers understood by each of its attached CE devices and the corresponding Ethernet tag. Depending on the multihoming mode of redundancy, only one path or all paths can be active at any one time.

The multihoming mode of operation along with VPWS service identifiers determine which PE router or routers forward and receive traffic in the EVPN-VPWS network. The VPWS service identifier identifies the endpoints of the EVPN-VPWS network. These endpoints are autodiscovered by BGP and are used to exchange the service labels(learned from the respective PE routers) that are used by autodiscovered routes per EVI route type. The service identifier is of two types:

- Local— Unique local VPWS service identifier. This is a logical identifier mapped to a physical interface of a PE router connected to the customer site that is forwarding the traffic to a remote VPWS service identifier.
- Remote—Unique remote VPWS service identifier. This is a logical identifier mapped to a physical interface of a PE router connected to the customer site that is receiving the traffic forwarded from the local VPWS service identifier.

The PE router forwarding traffic to the CE device uses MPLS LSP to forward traffic. If a failure occurs over this path, a new designated forwarder is elected to forward the traffic to the CE device.

The EVPN-VPWS network uses only an autodiscovered route per ESI and an autodiscovered router per EVI route types. In autodiscovered routes per EVI, the 24-bit values of the Ethernet tag are encoded with the VPWS service identifier. The autodiscovered route per ESI is encoded with the route targets of all the EVPN-VPWS instances connected to the ESI on the advertising PE router . When the PE router loses its connectivity to the ESI, it withdraws the autodiscovered route per ESI, resulting in faster convergence. The receiving PE router updates the forwarding next hop of the VPWS service identifier impacted by the failure. Depending on the mode of operation, these two endpoints of the EVPN-VPWS network can be colocated on the same PE router or on different PE routers. The different modes of operation in an EVPN-VPWS network are as follows:

- Local switching—In this mode, the VPWS endpoints (that is, local and remote service identifiers) are connected through the local interfaces configured on the same PE router. Traffic from one CE router is forwarded to another CE router through the same PE router.
- Single-homing—When a PE router is connected to a single-homed customer site, this mode is in operation.
- Active-standby multihoming—In this mode, only a single PE router among a group of PE routers with the same VPWS service identifier attached to an Ethernet segment is allowed to forward traffic to and from that Ethernet segment. The process of electing one among many PE routers with the same VPWS service identifier is known as the *designated forwarder (DF) election*. Each PE router connected to the Ethernet segment with the same VPWS service identifier participates in the DF election and informs the network of its status. The status can be as follows:
 - Designated forwarder(DF)—This is the designated forwarder for forwarding the current traffic.
 - Backup designated forwarder(BDF)—This becomes the DF in case the current DF encounters a failure.

- Non-designated forwarder(non-DF)—This is neither the DF nor the BDF. When more than two PE routers are part of an ESI redundancy group, then this PE router becomes a non-DF.

To configure the active-standby mode, include the ESI value and the `single-active` statement under the `[edit interfaces]` hierarchy level. Configure the local and the remote VPWS service identifier under the `[edit routing-instances vpws-routing-instance protocols evpn interface interface-name vpws-service-id]` for each PE router connected to the Ethernet segment.

In [Figure 109 on page 1047](#), for the PE routers connected to Device CE1, Router PE1 is assumed to be the DF and PE2 is assumed to be the BDF for VPWS service identifier 100. For the PE routers connected to CE2, PE3 is assumed to be the DF and Router PE4 is assumed to be the BDF for VPWS service identifier 200. PE2 and PE4 indicate their BDF status to CE1 and CE2 by setting their circuit cross-connect (CCC)-Down flag on their respective interfaces.

- Active-active multi-homing—In active-active multi homing, the CE device is connected to the PE routers with the same local VPWS identifier through the LAG interface so that the traffic is load-balanced among the set of multihomed PE routers with the same remote VPWS service identifiers. Here, election of DF is not required, because all the PE routers connected to the LAG interface participate in forwarding the traffic. In [Figure 109 on page 1047](#), Device CE1 forwards traffic to PE1 and PE2 with VPWS service identifier 100 and CE2 forwards traffic to PE3 and PE4 with VPWS service identifier 200. PE1 forwards the traffic to PE3 and PE4 with remote VPWS service identifier 200. PE2 forwards the traffic to PE3 and PE4. Similarly, traffic from CE2 to PE3 and PE4 with VPWS service identifier is load-balanced across PE routers with VPWS service identifier 100 connected to CE1.

Starting in cRPD Release 20.3R1, EVPN VPWS with EVPN Type 5 is supported to provide VPWS with EVPN signaling mechanisms on cRPD. VPWS with EVPN network is supported on single-homing.

NSR and Unified ISSU Support on VPWS with EVPN

Nonstop active routing (NSR) and graceful Routing Engine switchover (GRES) minimize traffic loss when there is a Routing Engine switchover. When a Routing Engine fails, NSR and GRES enable a routing platform with redundant Routing Engines to switch over from a primary Routing Engine to a backup Routing Engine and continue forwarding packets. Unified in-service software upgrade (ISSU) allows you to upgrade your Junos OS software on your MX Series router with no disruption on the control plane and with minimal disruption of traffic. Both GRES and NSR must be enabled to use unified ISSU.

To enable GRES, include the `graceful-switchover` statement at the `[edit chassis redundancy]` hierarchy level.

To enable NSR, include the `nonstop-routing` statement at the `[edit routing-options]` hierarchy level and the `commit synchronize` statement at the `[edit system]` hierarchy level.

Release History Table

Release	Description
20.3R1	Starting in cRPD Release 20.3R1, EVPN VPWS with EVPN Type 5 is supported to provide VPWS with EVPN signaling mechanisms on cRPD. VPWS with EVPN network is supported on single-homing.

RELATED DOCUMENTATION

[EVPN Multihoming Overview | 96](#)

[Configuring VPWS with EVPN Signaling Mechanisms | 1068](#)

[Example: Configuring VPWS with EVPN Signaling Mechanisms | 1070](#)

[vpws-service-id | 1665](#)

[show evpn vpws-instance | 1903](#)

[instance-type | 1589](#)

Control word for EVPN-VPWS

EVPN-VPWS is built on a MPLS network and the transit device's load-balancing hashing algorithm can cause out-of-order delivery of packets. The transit device can incorrectly identify an Ethernet payload as an IPv4 or IPv6 payload if the first nibble of the destination address MAC is 0x4 or 0x6, respectively. By inserting a control word between the label stack and the L2 header of the packet on the MPLS packet switched network, you can ensure that the top nibble is 0, thus preventing the packet from being identified as an IPv4 or IPv6 packet. The PE devices then negotiate support for control word in the EVPN-VPWS service. When you enable control word, the PE devices advertise their support in the auto-discovery route for each EVPN instance (EVI). Before a control word is inserted into the data packet, you must configure all the PE devices in an EVI on the EVPN-VPWS service and all the PE devices in the EVI agree to support control word. If any PE device in an EVI does not support control word, then PE devices will not include the control word in their packet.

NOTE: You do not need to enable control word on the devices in your transit network when the transit devices consist of Juniper Networks EX 9200 switches, MX series routers, or PTX Series routers. These Juniper devices correctly identify the Ethernet payload as an IPv4/IPv6 payloads, even when the Ethernet destination MAC address starts with 0x4 or 0x6 nibble. The Juniper devices perform hashing based on the IP header fields inside the Ethernet frame and will not

send out-of-order packets. In this case, we recommend not using control word as there are no benefits.

Figure 110 on page 1051 and Figure 111 on page 1052 illustrate a network with EVPN-VPWS service terminating in a Layer 3 VPN. In Figure 110 on page 1051, the customer device connects to an access device (A-PE1) which in turn connects to a service-edge device (PE1), that terminates into the layer 3 VPN. You must enable control word on A-PE1 and PE1, so that both devices can advertise their control word support in their route advertisement. Once control word support is established, the PEs will start inserting the control word in their packet.

Figure 110: Single-homed network with EVPN-VPWS service terminating in a Layer 3 VPN

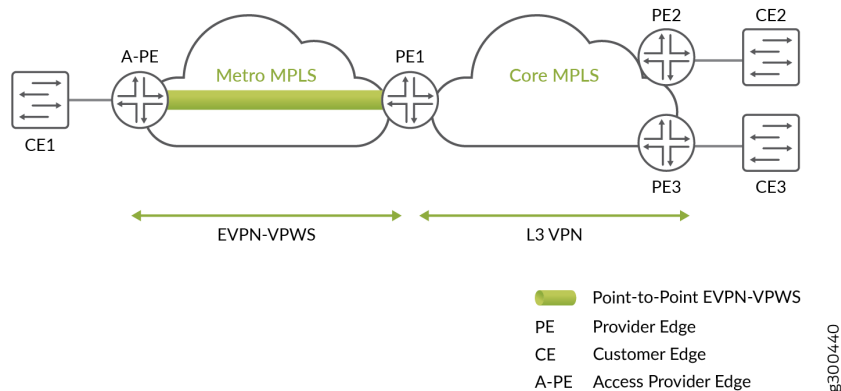
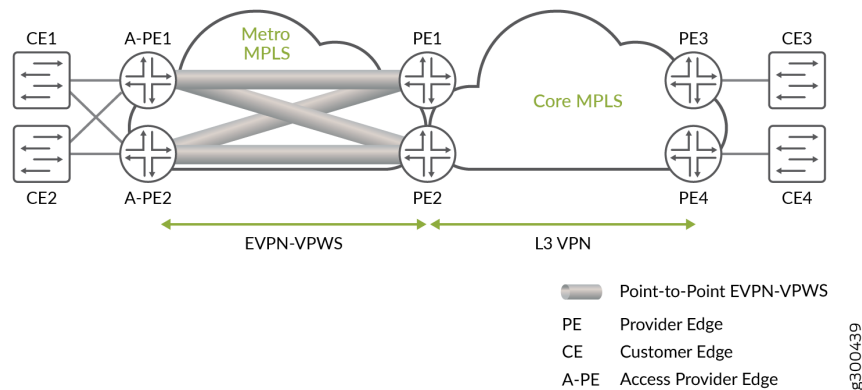


Figure 111 on page 1052 illustrates a topology where the customer device is multihomed to two access devices (A-PE1 and A-PE2), which in turn are multihomed to two service devices (PE1 and PE2). In both single-active and all-active multihoming, you must enable control word on A-PE1, A-PE2, PE1, and PE2

so that the devices can exchange their control word support. When control word support is confirmed for all the PEs in the EVPN-VPWS service, the PEs will start inserting the control word in the packet.

Figure 111: Multihomed network with EVPN-VPWS service terminating in a Layer 3 VPN



To enable control word, set control-word for the evpn protocol for a specified routing instance.

The following output shows a sample multihomed routing instance with control word configured.

```
user@router1# show routing-instances
MHEVPN {
  instance-type evpn-vpws;
  interface ge-0/0/1.0;
  interface ge-0/0/3.100;
  route-distinguisher 10.255.0.1:1;
  vrf-target target:123:123;
  protocols {
    evpn {
      control-word;
      interface ge-0/0/1.0 {
        vpws-service-id {
          local 9999;
          remote 1111;
        }
      }
      interface ge-0/0/3.100 {
        no-control-word;
        vpws-service-id {
          local 500;
          remote 200;
        }
      }
    }
  }
}
```

```
    }
  }
}
}
```

NOTE: The configuration for the interface takes precedence over the configuration for the EVPN protocol.

To view routes where control word is supported, use the `show route table mpls.0 protocol evpn operational` command. Egress routes display an offset of 252. Ingress routes display an offset of 4. When control word is not enable, the offset is not displayed.

```
show route table mpls.0 protocol evpn
300064      *[EVPN/7] 03:23:31, remote-pe 10.255.0.1, routing-instance mhevpn, route-type
Egress, vlan-id 9999
            > to 10.1.1.2 via ge-0/0/4.0, Push 299840, Push 300768(top) Offset: 252
ge-0/0/1.0  *[EVPN/7] 03:23:27, route-type Egress
            > to 10.1.1.2 via ge-0/0/4.0, Push 299840, Push 300768(top) Offset: 252
...
299984      *[EVPN/7] 03:24:48
            > via ge-0/0/1.0, Pop      Offset: 4
...
```

RELATED DOCUMENTATION

<i>Control Word for BGP VPLS Overview</i>
<i>control-word</i>
<i>no-control-word</i>

Overview of Flexible Cross-Connect Support on VPWS with EVPN

IN THIS SECTION

- [Benefits of Pseudowire Headend Termination \(PWHT\) with EVPN-VPWS FXC Pseudowires | 1055](#)
- [FXC on the PE Device \(Service Edge Router\) | 1055](#)
- [VLAN signaled FXC on the PE Device | 1056](#)
- [VLAN Tagging | 1057](#)
- [Signaling the Optional Bits Introduced for EVPN FXC | 1057](#)
- [Access Side Multi-homing | 1057](#)
- [Support EVPN FXC for MX Series as Access Router | 1059](#)

Ethernet VPN virtual private wire service (EVPN-VPWS) delivers the point-to-point services between a pair of Attachment Circuits (ACs). An AC is an attachment circuit or access interface participating in a EVPN E-LINE service. Multi-homing and fast convergence capabilities are also provided by EVPN-VPWS. You can now multiplex a large number of ACs across several physical interfaces onto a single VPWS service tunnel.

The EVPN-VPWS flexible cross-connect (FXC) is introduced to address label resource issue that occurs on some low end access routers (also known as A-leaf) by having a group of ACs under the same EVPN instance (EVI) share the same label. FXC is used to establish the point-to-point Ethernet services.

NOTE: The MPLS label resource issue does not apply to the PE device (also known as service edge router) or vMX (or MX), that uses pseudowire subscriber logical interface.

The control plane signaling is based on the exchange of EVPN per EVI auto-discovery (AD) routes between the pair of PEs, for FXC. It is a mandatory requirement that all the point-to-point Ethernet services under the same EVI is uniquely identified by either a single VLAN tag or double VLAN tags, and to ensure the uniqueness, VLAN normalization must be performed at the ingress. This mandatory requirement is applicable on both PE devices for both VLAN-unaware and VLAN-aware services.

The forwarding plane on the router that has limited label resource (access router), the MPLS label is used to identify the EVI. On both access router and PE device, the VLAN(s) carried in the data packet is used as the de-multiplexer to uniquely identify each local AC for the point-to-point service.

When the access device is a MX router, on the PE device, only single-homing and single-active multi-homing mode is supported. The customer edge (CE) device on the access side is either single-homed to an access router or multi-homed to a set of access routers in a single-active or all-active mode.

Benefits of Pseudowire Headend Termination (PWHT) with EVPN-VPWS FXC Pseudowires

- EVPN-VPWS FXC is an extension to basic EVPN-VPWS PWHT which bundles VLANs from multiple physical ports on access node into *single bundled* EVPN-VPWS pseudowire. The same pseudowire also shares more than one E-LINE service.
- Supports all true VLAN-aware bundle services for the VLAN signaled FXC.
- The FXC gives the benefit by the efficient usage of MPLS label resources. As a result, less labels and pseudowires are required, because of pseudowire bundling. This is useful on low-end access devices to conserve MPLS labels.

FXC on the PE Device (Service Edge Router)

EVPN-VPWS does not signal the VLAN in the control plane. At the PE device, label is allocated per pseudowire subscriber transport logical interface. The PE device interops with the access router that use different label allocation scheme. Encapsulation type ethernet-ccc is used on pseudowire subscriber transport logical interface.

Similar to the VLAN bundle service provided by the pseudowire subscriber logical interface at the PE device, for FXC, a group of ACs, each is represented by its unique normalized VLAN(s), is bundled together.

Single Home Support

All ACs at the access router are connected to the single homed end devices under the same EVI are multiplexed into a single point-to-point service tunnel. This bundle shares the same service instance ID and MPLS service label. Only one EVPN per EVI AD route is advertised for this bundle of ACs.

NOTE: The EVPN per EVI AD route advertised for the bundle of local ACs do not get withdrawal until all the ACs go down or until the EVI itself is deactivated or deleted.

To terminate one particular group of ACs at the access router, a separate pseudowire subscriber logical interface must be configured with the corresponding service instance ID at the PE device. The pseudowire subscriber logical interface works in a VLAN bundle mode.

Multi-home Support

In multi-homing of FXC, a separate per EVI AD route is advertised for each multi-home Ethernet segment. The same Ethernet segment identifier (ESI) ACs are bundled together in the same group. This is because the same PE may play the designated forwarder (DF) and non-designated forwarder (NDF) role independently for different Ethernet segment, but the same PE always acts as the DF or NDF for the ACs on the same Ethernet segment at the same time. As pseudowire subscriber logical interface configured with the corresponding service instance ID is used to terminate point-to-point Ethernet segment for the bundle of ACs, different pseudowire subscriber logical interface has to be used to terminate the different group of multi-homed ACs.

To support interoperability with VLAN-unaware multi-homing FXC at the access router, summary is as follows:

- ACs on the same Ethernet segment are bundled to use the same local service instance ID on the access router.
- Each AC group is assigned a unique local service instance ID, when there are many AC groups on the access router.
- At the PE device, a separate pseudowire subscriber logical interface must be used to terminate the group of ACs on the same ES attached to remote access router.
- There can be many pseudowire subscriber logical interfaces under the same EVI, however, each pseudowire subscriber transport logical interface must be assigned a unique service instance ID.

VLAN signaled FXC on the PE Device

For a group of AC under the same EVI, the VLAN signaled FXC shares the same MPLS label on the access router, which is similar to FXC. Under VLAN signaled FXC mode, each AC is assigned a unique service instance ID and is identified by one unique normalized VLAN identifier (for single tagged frame) or VLAN identifiers (for QinQ) in the forwarding plane. Thus the VLAN signaled FXC requires that each individual point-to-point Ethernet segment to be signaled separately through its own pair of EVPN Ethernet AD per EVI routes in the control plane

Pseudowire subscriber logical interface is used to support EVPN FXC VLAN-aware service. The local service instance ID is a property associated with the pseudowire subscriber service logical interface instead of pseudowire subscriber transport logical interface. For single-active multi-homing support, you can configure each pseudowire subscriber service logical interface with the multi-homing Ethernet segment using ESI per logical interface. There is no ESI configuration required for the pseudowire subscriber transport logical interface. Hence, the traffic is load balanced among redundant set of service edge router based on VLAN.

The following is supported on the PE device, for the interoperability with the access router which uses VLAN signaled FXC:

- There is only one pseudowire subscriber logical interface under each EVI. You must now manually configure a pair of local and remote service instance IDs on each of the pseudowire subscriber logical interface. Otherwise, the local service instance ID is auto-derived from the normalized VLAN ID(s) and the same ID is provided for remote service instance ID as well.
- A non-reserved ESI is configured for the pseudowire subscriber service logical interface, for single-active multi-homing mode.
- Each point-to-point Ethernet segment has a separate EVPN Ethernet AD per EVI route which is advertised with its Ethernet tag identifier set to the local instance ID. The Ethernet tag identifier can either be auto-derived based on the normalized VLAN ID(s) or manually configured.

VLAN Tagging

Pseudowire subscriber logical interface supports untagged, single tagged and double tagged frames. For interoperability with EVPN FXC, support for demux on single VLAN identifier or double VLANs (QinQ) for pseudowire subscriber logical interface is a must.

Signaling the Optional Bits Introduced for EVPN FXC

Both M-bit and V-bit are optional bits in the Layer 2 extended community for EVPN FXC. Currently, only the M-bit is signaled in the EVPN Layer 2 extended community, to indicate VLAN-unaware or VLAN signaled FXC.

Access Side Multi-homing

Access Side Multi-homing with Service Side Single-homing

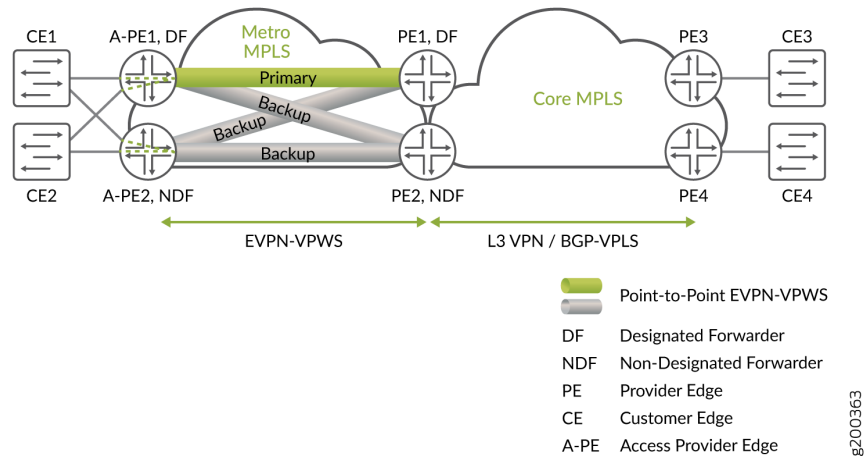
Currently, CE multi-homing to the access routers is either single-active or all-active mode (when the access device is a MX router) while the service edge router works in single-home mode is supported.

Access Side Single-active with Service Side Single-active

As shown in [Figure 112 on page 1058](#) below, this is a typical square topology consisting of two A-PE routers, A-PE1 and A-PE2, and two service edger routers, PE1 and PE2. PE1 and PE2 work in single-active mode. A-PE1 and A-PE2 also work in single-active mode. One of the service edge routers is elected as DF and one of access routers is elected as DF. There is only one active or primary pseudowire between DF access router and DF service edge router. If one of the DFs has an access link down or suffers node failure, the NDF or backup PE becomes the DF. Hence, the existing primary pseudowire

goes down and a new primary pseudowire is established among the DFs. This is done as per pseudowire subscriber service logical interface or per access link of access node. It is not done based on PE.

Figure 112: EVPN-VPWS Pseudowire Subscriber Interface Single Active

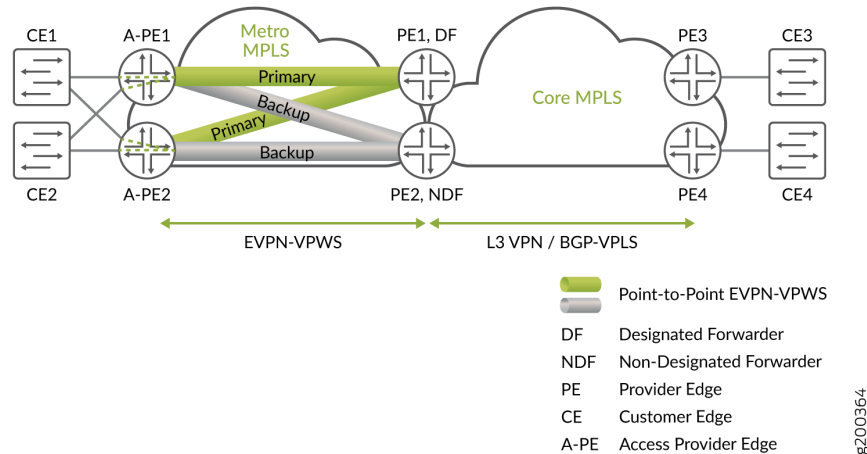


Access Side All-active with Service Side Single-active

As shown in [Figure 113 on page 1059](#) below, it is a square topology consisting of two A-PE routers, A-PE1 and A-PE2, and two service edger routers, PE1 and PE2. PE1 and PE2 work in single-active mode. A-PE1 and A-PE2 also work in all-active mode. There are two primary pseudowires from the primary or DF service router to the A-PEs. Traffic will be load balanced among A-PE1 and A-PE2.

NOTE: Always a MX router is used on the access side, for all-active mode.

Figure 113: EVPN-VPWS Pseudowire Subscriber Interface All Active



Support EVPN FXC for MX Series as Access Router

On the MX Series as access router, label per EVI is used when the service type is either EVPN FXC VLAN-unaware or EVPN FXC VLAN-aware. It is a mandatory that each AC under the same EVI must be uniquely identified through the normalized VLAN(s).

VLAN-Unaware Support on the Access Router

Under this mode of operation, all ACs are grouped separately on the basis of ESI. There is one group for all single-homed ACs and one group for each multi-homed Ethernet segment. ACs belong to the same group share the same local service instance ID. For a given EVI, the service instance ID is always unique. The service instance ID for each group must be configured manually.

VLAN-Aware Support on the Access Router

Under this mode of operation, each AC is assigned to a unique service instance ID. This service instance ID can be either manual configured or auto-derived based on the normalized VLAN identifiers.

Overview of Headend Termination for EVPN VPWS for Business Services

IN THIS SECTION

- [Benefits of Pseudowire Headend Termination \(PWHT\) with EVPN-VPWS Pseudowires: | 1060](#)
- [Pseudowire Subscriber Logical Interface Support for EVPN-VPWS | 1061](#)
- [vMX Scale Out | 1067](#)

Currently, Junos OS only supports pseudowire subscriber logical interface to be used with Layer 2 circuit or Layer 2 VPN for pseudowire headend termination. An Ethernet VPN (EVPN) enables you to connect dispersed customer sites using a Layer 2 virtual bridge. Virtual private wire service (VPWS) Layer 2 VPNs employ Layer 2 services over MPLS to build a topology of point-to-point connections that connect end customer sites in a VPN. EVPN-VPWS as a next generation of pseudowire technology brings the benefit of EVPN to point-to-point service by providing fast convergence upon node failure and link failure through its multi-homing feature. As a result, you can use EVPN-VPWS and pseudowire subscriber interface for headend termination into different services.

The pseudowire headend termination support with pseudowire subscriber interface works with EVPN-VPWS Flexible Cross Connect (FXC) in the vMX (Virtual MX) scale out solution. The vMX scale out support with EVPN-VPWS FXC is terminated into Layer 3 VRF or BGP-VPLS through pseudowire subscriber interface on vMX

NOTE: The multi-home scenario includes support for either two or more than two PEs.

Benefits of Pseudowire Headend Termination (PWHT) with EVPN-VPWS Pseudowires:

- PWHT framework can be used to attach remote CEs (reachable via metro aggregation network) to the services (for example Layer 3 VPN) available on service PE in a similar way to CEs which are directly attached to service PEs.
- Multiple VLANs can be transported inside pseudowire and demultiplexed to different services on service PE using pseudowire subscriber logical interfaces just like VLANs on regular physical interfaces.

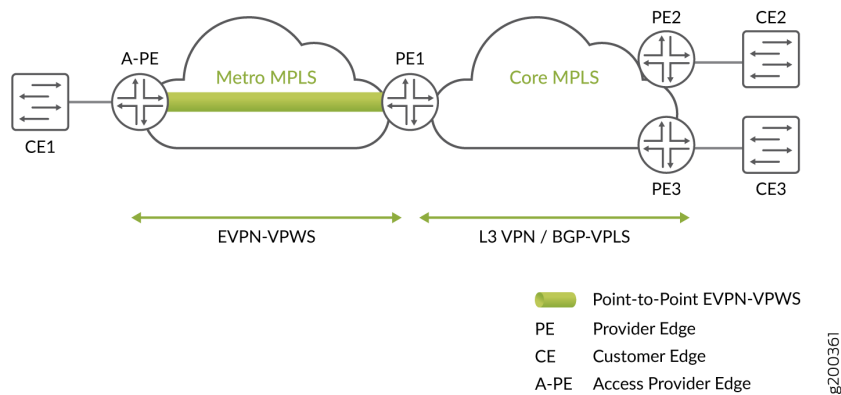
- PWHT with EVPN-VPWS as transport pseudowire brings further numerous benefits; for example unified BGP based control plane for all services in metro aggregation and core network, or redundant active-standby transport connectivity between multiple service PEs and multiple aggregation nodes.

Pseudowire Subscriber Logical Interface Support for EVPN-VPWS

Figure 114 on page 1061 illustrates an EVPN-VPWS network with pseudowire service. A-PE (Access Provider Edge) is the access router and PE1 is the service edge router. A pseudowire subscriber logical interface is used on PE1 to terminate the pseudowire established by EVPN-VPWS into either Layer 3 VPN or Layer 2 BGP-VPLS service. There is only one single service edge router to terminate the pseudowire, in this scenario.

For a given pseudowire subscriber logical interface, for example ps0, only the transport interface of the pseudowire subscriber logical interface, that is ps0.0, is used as an access interface for the EVPN-VPWS on PE1. The service interfaces of ps0, that is ps0.1 to ps0.n, are used at the service side (either under Layer 3 VRF or BGP-VPLS instance).

Figure 114: EVPN-VPWS with Pseudowire Subscriber Interface



NOTE: Prior to Junos OS 18.2 Release, the only encapsulation type that pseudowire subscriber transport logical interface (as an access interface for the pseudowire) supports for Layer 2 circuit and Layer 2 VPN is ethernet-ccc. This encapsulation type will remain the same for EVPN-VPWS with pseudowire subscriber logical interface support as well.

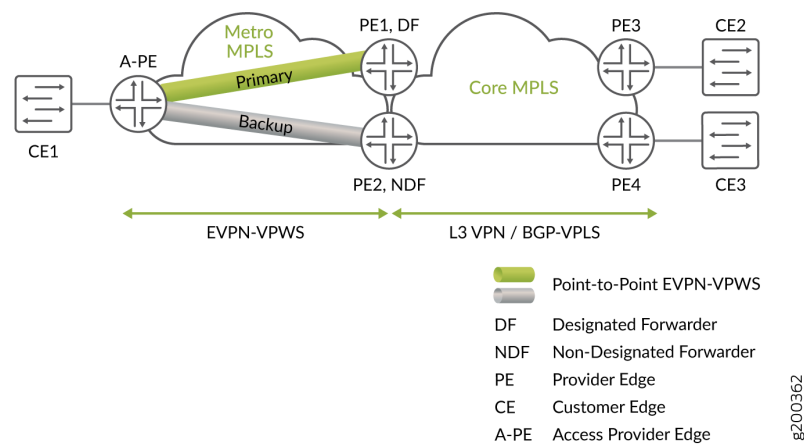
Single Pseudowire Headend Termination

The pseudowire subscriber transport logical interface is used as the access interface for EVPN-VPWS instance. EVPN establishes a point-to-point Ethernet service with pseudowire subscriber transport logical interface as an access interface in the control plane.

Active-Standby Pseudowire Headend Termination

As shown in [Figure 115 on page 1062](#), to achieve resilience for pseudowire headend termination into Layer 3 VPN or BGP-VPLS, you can use a pair of redundant pseudowires on the EVPN-VPWS side. Also, in [Figure 115 on page 1062](#), Layer 3 VPN or BGP-VPLS is treated as if it is multi-homed to the set of redundant service edge routers, PE1 and PE2. PE1 and PE2 work in active-standby mode for EVPN-VPWS. One of them is elected as the primary PE per EVPN normal designated forwarder (DF) election procedure. Only the primary PE is used to forward Layer 2 traffic.

Figure 115: EVPN-VPWS Active/standby with Pseudowire Subscriber Interface



Following are the reasons for the pseudowire failure, when redundancy is provided by the EVPN-VPWS active-standby interface:

- Service edge router node failure.
- Pseudowire subscriber transport logical interface failure on the primary PE.
- Failure on the path towards the primary PE.

NOTE: If any of the above failure cases are detected, the backup PE takes over to become the primary PE. As a result, the customer traffic from the access router, A-PE, is switched to the new primary PE.

The EVPN-VPWS also supports the service side of the network multi-homing to EVPN-VPWS through the pseudowire subscriber logical interface in an active-standby mode.

To achieve active-standby pseudowire headend termination, the following is required:

- *Ethernet segment identifier (ESI)* support on the pseudowire subscriber transport logical interface.
- Synchronizing data path between active pseudowire and Layer 3 VPN.
- MAC flush in the BGP-VPLS when the active-standby pseudowires switch. MAC flush is triggered when the active service edge node suffers a node failure.

ES Support on Pseudowire Subscriber Transport Logical Interface

An Ethernet segment must have a unique nonzero identifier, called the *Ethernet segment identifier (ESI)*. The ESI is encoded as a 10-octet integer. When manually configuring an ESI value, the most significant octet, known as the *type byte*, must be 00. ESI is assigned to pseudowire subscriber transport logical interfaces associated with the PE1 and PE2 the same way as in EVPN multi-homing.

To configure the active-standby mode for the pseudowire subscriber transport logical interface, include the ESI value and the `single-active` statement under the `[edit interfaces]` hierarchy level.

DF Election and Pseudowire Subscriber Transport Logical Interface

Designated forwarder(DF)—This is the designated forwarder for forwarding the current traffic. The DF election procedure ensures each VLAN is associated with single PE acting in DF role.

Synchronizing Data Path between EVPN-VPWS and Layer 3 VPN or BGP-VPLS

There is a need for the data path coordination between the point-to-point pseudowire and the services side of the network (Layer 3 VPN or BGP-VPLS), with active-standby pseudowire headend termination. This is to select the same service edge router for passing the current traffic between the point-to-point pseudowire and Layer 3 VPN or BGP-VPLS domain.

EVPN-VPWS determines the primary path, for both Layer 3 VPN and BGP-VPLS, based on its DF election result. Both the primary and backup PEs for EVPN-VPWS then influence the services side of network to use the primary PE for data traffic.

Layer 3 VPN

When pseudowires are headend terminated into the Layer 3 VPN in an active-standby mode, the primary and the backup PEs use routing-policy and policy-condition to increase or decrease the Local Preference (LP). As a result, the BGP path selection algorithm is carried out on the Layer 3 side and picks up the primary PE.

BGP-VPLS

EVPN-VPWS establishes active-standby pseudowires based on its DF election. The active-standby pseudowires are essentially the two redundant spoke pseudowires attached to BGP-VPLS instance. The BGP-VPLS PEs remain single homed PEs and data plane learning happens through the active pseudowire in the EVPN-VPWS primary path.

NOTE:

- If the pseudowire subscriber service logical interface that belongs to a BGP-VPLS instance is in down state, this does not trigger the active and standby pseudowire switch on the EVPN-VPWS side. This is because the 1 to many relationships between the point-to-point pseudowire and the BGP-VPLS. This may cause traffic loss for traffic coming from the VPLS to the point-to-point pseudowire direction.
- BGP-VPLS does not trigger MAC flush when its access link to the CE device goes up or down. Further, since multi-homed EVPN-VPWS pseudowire headend termination into BGP-VPLS is based on single-homed mode on BGP-VPLS side, MAC flush (achieved via manipulation of F-bit in BGP-VPLS messages) associated with multi-homed BGP-VPLS mode does not apply here, neither. Therefore, when there is DF or NDF state change on EVPN-VPWS side, or if the pseudowire subscriber service logical interface that belongs to a BGP-VPLS instance goes up or down, MAC flush is not triggered. Null-route filtering might occur until expiration of MAC aging timer, unless there is constant traffic from the CE behind the EVPN-VPWS to BGP-VPLS. MAC flush is triggered only during PE node failures.

The following configuration example shows how condition manager is used for Layer 3 VPN routing instance through vrf-export when active-standby EVPN-VPWS terminates into Layer 3 VPN service.

```
[edit]
routing-instances {
  l3vpn_1 {
    instance-type vrf;
    interface ps0.1;
```

```

interface lo0.1;
route-distinguisher 2.2.2.3:6500;
vrf-import l3vpn1_import;
vrf-export l3vpn1_export;
vrf-table-label;
protocols {
    group toPW-CE1 {
        type external;
        export send-direct;
        peer-as 1;
        neighbor 10.1.1.1;
    }
}
}
}
policy-options {
    policy-statement l3vpn_1_export {
        term 1 {
            from condition primary;
            then {
                local-preference add 300;
                community set l3vpn_1;
                accept;
            }
        }
        term 2 {
            from condition standby;
            then {
                community set l3vpn_1;
                accept;
            }
        }
    }
    policy-statement l3vpn1_import {
        term 1 {
            from community l3vpn_1;
            then accept;
        }
        term default {
            then reject;
        }
    }
}
community l3vpn_1 members target:65056:100;

```

```

condition primary {
    if-route-exists {
        address-family {
            ccc {
                ps0.0;
                table mpls.0;
            }
        }
    }
}
condition standby {
    if-route-exists {
        address-family {
            ccc {
                ps0.0;
                table mpls.0;
                standby;
            }
        }
    }
}
}

```

Following is the BGP-VPLS instance configuration when EVPN-VPWS terminates into the BGP-VPLS:

```

[edit]
routing-instances {
    vpls-1 {
        instance-type vpls;
        interface ps0.1;
        route-distinguisher 100:2;
        vrf-target target:100:100;
        protocols {
            vpls {
                site ce3 {
                    site-identifier 3;
                }
            }
        }
    }
}

```

```
}
}
```

Active-Active Pseudowire Headend Termination

Active-active pseudowire headend termination is supported only in Layer 3 VPN and not supported in BGP-VPLS.

vMX Scale Out

The vMX router is a virtual version of the MX Series 5G Universal Routing Platform. Like the MX Series router, the vMX router runs the Junos operating system (Junos OS) and supports Junos OS packet handling and forwarding modeled after the Trio chipset. The vMX instance contains two separate virtual machines (VMs), one for the virtual forwarding plane (VFP) and one for the virtual control plane (VCP). The VFP VM runs the virtual Trio forwarding plane software and the VCP VM runs Junos OS.

The vMX can now be used to scale out the bandwidth, achieve service isolation, and resilience. As the service edge router, vMX supports pseudowire headend termination with active and standby mode.

vMX

A vMX can have active and backup VCP with one or more VFPs. The VCP is the RE and the VFP is the line-card. The communication between VCP and VFP is facilitated through vRouter, if the VCP and VFP reside on the same server. Else, it is through the external network to which the servers are connected. There is no direction communication between VFPs.

Service Isolation

For a given point-to-point pseudowire with only one VFP per vMX, the data traffic is handled by one VFP for both ingress and egress traffic.

Active-Standby Pseudowire Headend Termination into VMX

The vMX acts as a regular MX for the overlay point-to-point Ethernet service. Similar to physical MX router, redundancy is achieved by using multiple vMXs, and only active-standby pseudowire headend termination is supported. The pseudowire is terminated at the loopback IP address that is used as the protocol nexthop address for the pseudowire.

RELATED DOCUMENTATION

[Example: Configuring VPWS with EVPN Signaling Mechanisms](#) | 1070

Configuring VPWS with EVPN Signaling Mechanisms

The virtual private wire service (VPWS) with Ethernet VPN (EVPN) provides single-active or all-active multihoming capabilities along with support for Inter-AS options associated with BGP-signaled VPNs. The VPWS service identifiers identify the endpoints of the EVPN-VPWS network. Each provider edge (PE) router in EVPN-VPWS network is configured with local and remote VPWS service identifiers.

Before you configure VPWS service with EVPN mechanisms, you must do the following:

1. Configure the router interfaces.
2. Configure the router ID and autonomous system number for the device.
3. Configure OSPF.
4. Configure a BGP internal group.
5. Configure ISIS.
6. Include the EVPN signaling network layer reachability information (NLRI) to the internal BGP group.
7. Configure LDP.
8. Configure MPLS.
9. Configure MPLS LSP or GRE tunnels.
10. Configure EVPN all-active multihoming and EVPN single-active multihoming.

To configure a PE device with a VPWS service identifier in an EVPN-VPWS network, do the following:

1. Configure the EVPN-VPWS routing instance.

```
[edit routing-instances]
user@PE# set evpn-vpws-instance instance-type instance-type-value
```

For example, configure routing instance `vpws1004` with instance type `evpn-vpws`.

```
[edit routing-instances]
user@PE# set vpws1004 instance-type evpn-vpws
```

2. Configure the interface names for the EVPN-VPWS routing instance.

```
[edit routing-instances]
user@PE# set evpn-vpws-instance interface interface-name
```

For example, configure interface ge-0/0/1.1004 for the EVPN-VPWS instance vpws1004.

```
[edit routing-instances]
user@PE# set vpws1004 interface ge-0/0/1.1004
```

3. Configure the route distinguisher for the EVPN-VPWS routing instance.

```
[edit routing-instances]
user@PE# set evpn-vpws-instance route-distinguisher route-distinguisher-value
```

For example, configure route distinguisher 10.255.0.1:100 for EVPN-VPWS routing instance vpws1004.

```
[edit routing-instances]
user@PE# set vpws1004 route-distinguisher 10.255.0.1:100
```

4. Configure the VPN routing and forwarding (VRF) target community for the EVPN-VPWS routing instance.

```
[edit routing-instances]
user@PE# set evpn-vpws-instance vrf-target vrf-target
```

For example, configure VRF target target:100:1004 for EVPN-VPWS routing instance vpws1004.

```
[edit routing-instances]
user@PE# set vpws1004 vrf-target target:100:1004
```

5. Configure the interface of a routing instance with local and remote service identifiers. These identifiers identify the PE routers that forward and receive the traffic in the EVPN-VPWS network.

The local service identifier is used to identify the PE router that is forwarding the traffic, and the remote service identifier is used to identify the PE router that is receiving the traffic in the network.

```
[edit routing-instances evpn-vpws-instance protocols evpn interface interface-name]  
user@PE# set vpws-service-id local local-service-id  
user@PE# set vpws-service-id remote remote-service-id
```

For example, configure the interface ge-0/0/1.1004 with the local and remote service identifiers 1004 and 2004 for EVPN-VPWS routing instance vpws1004.

```
[edit routing-instances vpws1004 protocols evpn interface ge-0/0/1.1004]  
user@PE# set vpws-service-id local 1004  
user@PE# set vpws-service-id remote 2004
```

RELATED DOCUMENTATION

[Overview of VPWS with EVPN Signaling Mechanisms](#) | 1046

Example: Configuring VPWS with EVPN Signaling Mechanisms

IN THIS SECTION

- [Requirements](#) | 1070
- [Overview and Topology](#) | 1071
- [Configuration](#) | 1073
- [Verification](#) | 1085

This example shows how to implement Virtual Private Wire Service (VPWS) with Ethernet Virtual Private Network (EVPN) signaling. The use of EVPN signaling provides single-active or all-active multihoming capabilities for BGP-signaled VPNs.

Requirements

This example uses the following hardware and software components:

- Four MX Series routers acting as provider edge (PE) devices, running Junos OS Release 17.1 or later
- Two customer edge (CE) devices (MX Series routers are used in this example)

Overview and Topology

IN THIS SECTION

- [Topology | 1072](#)

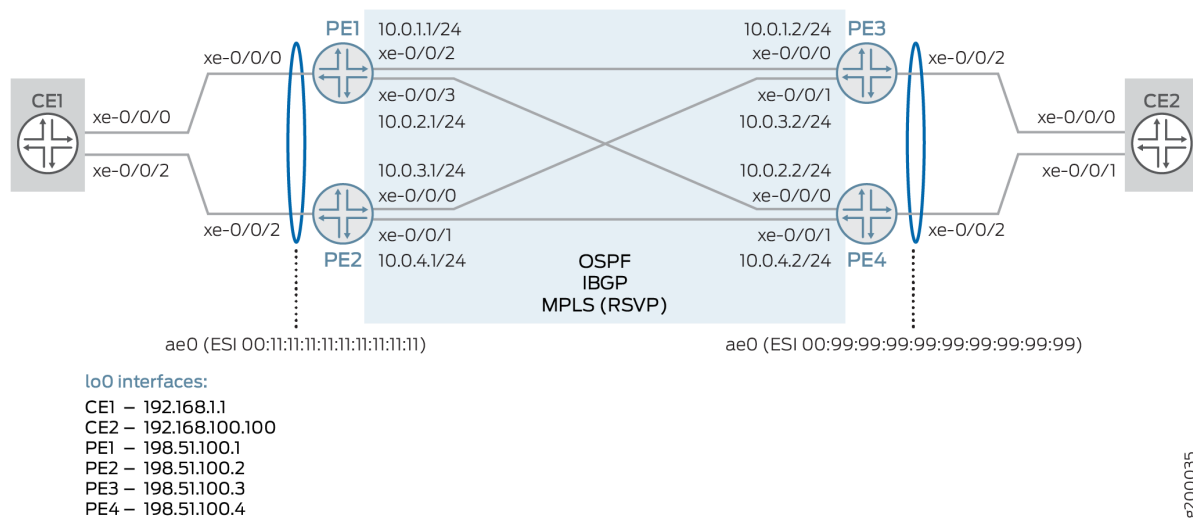
VPWS employs Layer 2 VPN services over MPLS to build a topology of point-to-point connections that connect end customer sites. EVPN enables you to connect dispersed customer sites using a Layer 2 virtual bridge. Starting with Junos OS Release 17.1, these two elements can be combined to provide an EVPN-signaled VPWS.

The `vpws-service-id` statement identifies the endpoints of the EVPN-VPWS based on `local` and `remote` service identifiers configured on the PE routers in the network. These endpoints are autodiscovered using BGP-based EVPN signaling to exchange the service identifier labels.

Topology

This example uses the topology shown in [Figure 116 on page 1072](#), consisting of four PE routers and two CE routers. Router CE1 is multihomed to Routers PE1 and PE2; Router CE2 is multihomed to Routers PE3 and PE4.

Figure 116: VPWS with EVPN Signaling



The following configuration elements are used in this scenario:

- CE devices:
 - Aggregated Ethernet (AE) interface towards related PE devices
- PE devices:
 - AE interface, with EVPN segment identifier (ESI), towards related CE device
 - OSPF and IBGP in the core
 - MPLS LSPs using RSVP in the core
 - Per-packet load balancing
 - Routing instance using instance type `evpn-vpws`, and the `vpws-service-id` statement to define the local and remote endpoints.

Configuration

IN THIS SECTION

- [CLI Quick Configuration | 1073](#)
- [Procedure | 1078](#)
- [Results | 1082](#)

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the [edit] hierarchy level.

CE1

```
set interfaces xe-0/0/0 gigether-options 802.3ad ae0
set interfaces xe-0/0/2 gigether-options 802.3ad ae0
set chassis aggregated-devices ethernet device-count 1
set interfaces ae0 description "to PE1/2"
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 unit 100 encapsulation vlan-bridge
set interfaces ae0 unit 100 vlan-id 1000
set interfaces lo0 unit 0 family inet address 192.168.1.1/32
set policy-options policy-statement LB then load-balance per-packet
set routing-options forwarding-table export LB
set bridge-domains bd100 domain-type bridge
set bridge-domains bd100 vlan-id 1000
set bridge-domains bd100 interface ae0.100
```

CE2

```
set interfaces xe-0/0/0 gigether-options 802.3ad ae0
set interfaces xe-0/0/1 gigether-options 802.3ad ae0
set chassis aggregated-devices ethernet device-count 1
set interfaces ae0 description "to PE3/4"
```

```

set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 unit 100 encapsulation vlan-bridge
set interfaces ae0 unit 100 vlan-id 1000
set interfaces lo0 unit 0 family inet address 192.168.100.100/32
set policy-options policy-statement LB then load-balance per-packet
set routing-options forwarding-table export LB
set bridge-domains bd100 domain-type bridge
set bridge-domains bd100 vlan-id 1000
set bridge-domains bd100 interface ae0.100

```

PE1

```

set interfaces xe-0/0/0 description "to CE1"
set interfaces xe-0/0/0 gigether-options 802.3ad ae0
set interfaces xe-0/0/2 unit 0 description "to PE3"
set interfaces xe-0/0/2 unit 0 family inet address 10.0.1.1/24
set interfaces xe-0/0/2 unit 0 family mpls
set interfaces xe-0/0/3 unit 0 description "to PE4"
set interfaces xe-0/0/3 unit 0 family inet address 10.0.2.1/24
set interfaces xe-0/0/3 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 198.51.100.1/32
set chassis aggregated-devices ethernet device-count 1
set interfaces ae0 description "to CE1"
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 esi 00:11:11:11:11:11:11:11:11
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp system-id 00:00:00:00:00:01
set interfaces ae0 unit 100 encapsulation vlan-ccc
set interfaces ae0 unit 100 vlan-id 1000
set protocols ospf area 0.0.0.0 interface xe-0/0/2.0
set protocols ospf area 0.0.0.0 interface xe-0/0/3.0
set protocols ospf area 0.0.0.0 interface lo0.0
set routing-options autonomous-system 65000
set protocols bgp group IBGP type internal
set protocols bgp group IBGP local-address 198.51.100.1
set protocols bgp group IBGP family evpn signaling
set protocols bgp group IBGP neighbor 198.51.100.2
set protocols bgp group IBGP neighbor 198.51.100.3

```

```

set protocols bgp group IBGP neighbor 198.51.100.4
set protocols rsvp interface xe-0/0/2.0
set protocols rsvp interface xe-0/0/3.0
set protocols mpls interface xe-0/0/2.0
set protocols mpls interface xe-0/0/3.0
set protocols mpls no-cspf
set protocols mpls label-switched-path PE1toPE3 to 198.51.100.3
set protocols mpls label-switched-path PE1toPE4 to 198.51.100.4
set policy-options policy-statement LB then load-balance per-packet
set routing-options forwarding-table export LB
set routing-instances EVPN-VPWS instance-type evpn-vpws
set routing-instances EVPN-VPWS interface ae0.100
set routing-instances EVPN-VPWS route-distinguisher 198.51.100.1:11
set routing-instances EVPN-VPWS vrf-target target:100:11
set routing-instances EVPN-VPWS protocols evpn interface ae0.100 vpws-service-id local 1111
set routing-instances EVPN-VPWS protocols evpn interface ae0.100 vpws-service-id remote 9999

```

PE2

```

set interfaces xe-0/0/0 unit 0 description "to PE3"
set interfaces xe-0/0/0 unit 0 family inet address 10.0.3.1/24
set interfaces xe-0/0/0 unit 0 family mpls
set interfaces xe-0/0/1 unit 0 description "to PE4"
set interfaces xe-0/0/1 unit 0 family inet address 10.0.4.1/24
set interfaces xe-0/0/1 unit 0 family mpls
set interfaces xe-0/0/2 description "to CE1"
set interfaces xe-0/0/2 gigether-options 802.3ad ae0
set interfaces lo0 unit 0 family inet address 198.51.100.2/32
set chassis aggregated-devices ethernet device-count 1
set interfaces ae0 description "to CE1"
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 esi 00:11:11:11:11:11:11:11:11:11
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp system-id 00:00:00:00:00:01
set interfaces ae0 unit 100 encapsulation vlan-ccc
set interfaces ae0 unit 100 vlan-id 1000
set protocols ospf area 0.0.0.0 interface xe-0/0/0.0
set protocols ospf area 0.0.0.0 interface xe-0/0/1.0
set protocols ospf area 0.0.0.0 interface lo0.0
set routing-options autonomous-system 65000

```

```

set protocols bgp group IBGP type internal
set protocols bgp group IBGP local-address 198.51.100.2
set protocols bgp group IBGP family evpn signaling
set protocols bgp group IBGP neighbor 198.51.100.1
set protocols bgp group IBGP neighbor 198.51.100.3
set protocols bgp group IBGP neighbor 198.51.100.4
set protocols rsvp interface xe-0/0/0.0
set protocols rsvp interface xe-0/0/1.0
set protocols mpls interface xe-0/0/0.0
set protocols mpls interface xe-0/0/1.0
set protocols mpls no-cspf
set protocols mpls label-switched-path PE2toPE3 to 198.51.100.3
set protocols mpls label-switched-path PE2toPE4 to 198.51.100.4
set policy-options policy-statement LB then load-balance per-packet
set routing-options forwarding-table export LB
set routing-instances EVPN-VPWS instance-type evpn-vpws
set routing-instances EVPN-VPWS interface ae0.100
set routing-instances EVPN-VPWS route-distinguisher 198.51.100.2:11
set routing-instances EVPN-VPWS vrf-target target:100:11
set routing-instances EVPN-VPWS protocols evpn interface ae0.100 vpws-service-id local 1111
set routing-instances EVPN-VPWS protocols evpn interface ae0.100 vpws-service-id remote 9999

```

PE3

```

set interfaces xe-0/0/0 unit 0 description "to PE1"
set interfaces xe-0/0/0 unit 0 family inet address 10.0.1.2/24
set interfaces xe-0/0/0 unit 0 family mpls
set interfaces xe-0/0/1 unit 0 description "to PE2"
set interfaces xe-0/0/1 unit 0 family inet address 10.0.3.2/24
set interfaces xe-0/0/1 unit 0 family mpls
set interfaces xe-0/0/2 description "to CE1"
set interfaces xe-0/0/2 gigether-options 802.3ad ae0
set interfaces lo0 unit 0 family inet address 198.51.100.3/32
set chassis aggregated-devices ethernet device-count 1
set interfaces ae0 description "to CE2"
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 esi 00:99:99:99:99:99:99:99
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp system-id 00:00:00:00:00:01
set interfaces ae0 unit 100 encapsulation vlan-ccc

```

```

set interfaces ae0 unit 100 vlan-id 1000
set protocols ospf area 0.0.0.0 interface xe-0/0/0.0
set protocols ospf area 0.0.0.0 interface xe-0/0/1.0
set protocols ospf area 0.0.0.0 interface lo0.0
set routing-options autonomous-system 65000
set protocols bgp group IBGP type internal
set protocols bgp group IBGP local-address 198.51.100.3
set protocols bgp group IBGP family evpn signaling
set protocols bgp group IBGP neighbor 198.51.100.1
set protocols bgp group IBGP neighbor 198.51.100.2
set protocols bgp group IBGP neighbor 198.51.100.4
set protocols rsvp interface xe-0/0/0.0
set protocols rsvp interface xe-0/0/1.0
set protocols mpls interface xe-0/0/0.0
set protocols mpls interface xe-0/0/1.0
set protocols mpls no-cspf
set protocols mpls label-switched-path PE3toPE1 to 198.51.100.1
set protocols mpls label-switched-path PE3toPE2 to 198.51.100.2
set policy-options policy-statement LB then load-balance per-packet
set routing-options forwarding-table export LB
set routing-instances EVPN-VPWS instance-type evpn-vpws
set routing-instances EVPN-VPWS interface ae0.100
set routing-instances EVPN-VPWS route-distinguisher 198.51.100.3:11
set routing-instances EVPN-VPWS vrf-target target:100:11
set routing-instances EVPN-VPWS protocols evpn interface ae0.100 vpws-service-id local 9999
set routing-instances EVPN-VPWS protocols evpn interface ae0.100 vpws-service-id remote 1111

```

PE4

```

set interfaces xe-0/0/0 unit 0 description "to PE1"
set interfaces xe-0/0/0 unit 0 family inet address 10.0.2.2/24
set interfaces xe-0/0/0 unit 0 family mpls
set interfaces xe-0/0/1 unit 0 description "to PE2"
set interfaces xe-0/0/1 unit 0 family inet address 10.0.4.2/24
set interfaces xe-0/0/1 unit 0 family mpls
set interfaces xe-0/0/2 description "to CE1"
set interfaces xe-0/0/2 gigether-options 802.3ad ae0
set interfaces lo0 unit 0 family inet address 198.51.100.4/32
set chassis aggregated-devices ethernet device-count 1
set interfaces ae0 description "to CE2"
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services

```



```

set interfaces ae0 esi 00:99:99:99:99:99:99:99
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp system-id 00:00:00:00:00:01
set interfaces ae0 unit 100 encapsulation vlan-ccc
set interfaces ae0 unit 100 vlan-id 1000
set protocols ospf area 0.0.0.0 interface xe-0/0/0.0
set protocols ospf area 0.0.0.0 interface xe-0/0/1.0
set protocols ospf area 0.0.0.0 interface lo0.0
set routing-options autonomous-system 65000
set protocols bgp group IBGP type internal
set protocols bgp group IBGP local-address 198.51.100.4
set protocols bgp group IBGP family evpn signaling
set protocols bgp group IBGP neighbor 198.51.100.1
set protocols bgp group IBGP neighbor 198.51.100.2
set protocols bgp group IBGP neighbor 198.51.100.3
set protocols rsvp interface xe-0/0/0.0
set protocols rsvp interface xe-0/0/1.0
set protocols mpls interface xe-0/0/0.0
set protocols mpls interface xe-0/0/1.0
set protocols mpls no-cspf
set protocols mpls label-switched-path PE4toPE1 to 198.51.100.1
set protocols mpls label-switched-path PE4toPE2 to 198.51.100.2
set policy-options policy-statement LB then load-balance per-packet
set routing-options forwarding-table export LB
set routing-instances EVPN-VPWS instance-type evpn-vpws
set routing-instances EVPN-VPWS interface ae0.100
set routing-instances EVPN-VPWS route-distinguisher 198.51.100.4:11
set routing-instances EVPN-VPWS vrf-target target:100:11
set routing-instances EVPN-VPWS protocols evpn interface ae0.100 vpws-service-id local 9999
set routing-instances EVPN-VPWS protocols evpn interface ae0.100 vpws-service-id remote 1111

```

Procedure

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the [CLI User Guide](#).

NOTE: Only Router PE1 is shown here. Repeat this procedure for all other PE devices, using the appropriate interface names, addresses, and other parameters for each device.

The step-by-step procedure for CE devices is not shown.

To configure Router PE1:

1. Configure the CE-facing interface to be part of the ae0 bundle.

The second interface for the AE bundle will be configured on the other local PE device.

```
[edit interfaces]
user@PE1# set xe-0/0/0 description "to CE1"
user@PE1# set xe-0/0/0 gigether-options 802.3ad ae0
```

2. Configure the core-facing interfaces toward Routers PE3 and PE4.

Be sure to include the MPLS protocol family.

```
[edit interfaces]
user@PE1# set xe-0/0/2 unit 0 description "to PE3"
user@PE1# set xe-0/0/2 unit 0 family inet address 10.0.1.1/24
user@PE1# set xe-0/0/2 unit 0 family mpls
user@PE1# set xe-0/0/3 unit 0 description "to PE4"
user@PE1# set xe-0/0/3 unit 0 family inet address 10.0.2.1/24
user@PE1# set xe-0/0/3 unit 0 family mpls
```

3. Configure the loopback interface.

```
[edit interfaces]
user@PE1# set lo0 unit 0 family inet address 198.51.100.1/32
```

4. Define the number of aggregated Ethernet interfaces to be supported on the device.

```
[edit chassis]
user@PE1# set aggregated-devices ethernet device-count 1
```

5. Configure the ae0 interface.

Alternate VLAN tagging and encapsulation options can be used, depending on your needs.

```
[edit interfaces]
user@PE1# set ae0 description "to CE1"
user@PE1# set ae0 flexible-vlan-tagging
user@PE1# set ae0 encapsulation flexible-ethernet-services
user@PE1# set ae0 unit 100 encapsulation vlan-ccc
user@PE1# set ae0 unit 100 vlan-id 1000
```

6. Assign an Ethernet segment identifier (ESI) value to the ae0 interface and enable EVPN active-active multihoming.

```
[edit interfaces]
user@PE1# set ae0 esi 00:11:11:11:11:11:11:11:11
user@PE1# set ae0 esi all-active
```

7. Configure link aggregation control protocol (LACP) for the ae0 interface.

The system ID used here must be the same on both local PE devices.

```
[edit interfaces]
user@PE1# set ae0 aggregated-ether-options lacp active
user@PE1# set ae0 aggregated-ether-options lacp system-id 00:00:00:00:00:01
```

8. Enable OSPF on the core-facing (and loopback) interfaces.

```
[edit protocols]
user@PE1# set ospf area 0.0.0.0 interface xe-0/0/2.0
user@PE1# set ospf area 0.0.0.0 interface xe-0/0/3.0
user@PE1# set ospf area 0.0.0.0 interface lo0.0
```

9. Configure an IBGP mesh with the other PE devices, using EVPN for signaling.

```
[edit routing-options]
user@PE1# set autonomous-system 65000
[edit protocols]
user@PE1# set bgp group IBGP type internal
user@PE1# set bgp group IBGP local-address 198.51.100.1
```

```

user@PE1# set bgp group IBGP family evpn signaling
user@PE1# set bgp group IBGP neighbor 198.51.100.2
user@PE1# set bgp group IBGP neighbor 198.51.100.3
user@PE1# set bgp group IBGP neighbor 198.51.100.4

```

10. Enable RSVP on the core-facing interfaces.

```

[edit protocols]
user@PE1# set rsvp interface xe-0/0/2.0
user@PE1# set rsvp interface xe-0/0/3.0

```

11. Enable MPLS on the core-facing interfaces, and configure LSPs to the remote PE devices.

For this example, be sure to disable CSPF.

```

[edit protocols]
user@PE1# set mpls interface xe-0/0/2.0
user@PE1# set mpls interface xe-0/0/3.0
user@PE1# set mpls no-cspf
user@PE1# set mpls label-switched-path PE1toPE3 to 198.51.100.3
user@PE1# set mpls label-switched-path PE1toPE4 to 198.51.100.4

```

12. Configure load balancing.

```

[edit policy-options]
user@PE1# set policy-statement LB then load-balance per-packet
[edit routing-options]
user@PE1# set forwarding-table export LB

```

13. Configure a routing instance using the `evpn-vpws` instance type. Add the AE (CE-facing) interface configured earlier, as well as a route distinguisher and VRF target.

In EVPN terms, this is an EVPN instance (EVI).

```

[edit routing-instances]
user@PE1# set EVPN-VPWS instance-type evpn-vpws
user@PE1# set EVPN-VPWS interface ae0.100
user@PE1# set EVPN-VPWS route-distinguisher 198.51.100.1:11
user@PE1# set EVPN-VPWS vrf-target target:100:11

```

14. In the routing instance, enable EVPN and add the AE interface. Then associate local and remote VPWS identifiers to the interface.

```
[edit routing-instances]
user@PE1# set EVPN-VPWS protocols evpn interface ae0.100 vpws-service-id local 1111
user@PE1# set EVPN-VPWS protocols evpn interface ae0.100 vpws-service-id remote 9999
```

Results

From configuration mode, confirm your configuration. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit ]
user@PE1# show chassis
aggregated-devices {
  ethernet {
    device-count 1;
  }
}
```

```
[edit ]
user@PE1# show interfaces
xe-0/0/0 {
  description "to CE1";
  gigether-options {
    802.3ad ae0;
  }
}
xe-0/0/2 {
  unit 0 {
    description "to PE3";
    family inet {
      address 10.0.1.1/24;
    }
    family mpls;
  }
}
xe-0/0/3 {
  unit 0 {
```

```

        description "to PE4";
        family inet {
            address 10.0.2.1/24;
        }
        family mpls;
    }
}
ae0 {
    description "to CE1";
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    esi {
        00:11:11:11:11:11:11:11:11;
        all-active;
    }
    aggregated-ether-options {
        lacp {
            active;
            system-id 00:00:00:00:00:01;
        }
    }
    unit 100 {
        encapsulation vlan-ccc;
        vlan-id 1000;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 198.51.100.1/32;
        }
    }
}
}

```

```

[edit ]
user@PE1# show routing-options
autonomous-system 65000;
forwarding-table {

```

```
export LB;
}
```

```
user@PE1# show protocols
rsvp {
  interface xe-0/0/2.0;
  interface xe-0/0/3.0;
}
mpls {
  no-cspf;
  label-switched-path PE1toPE3 {
    to 198.51.100.3;
  }
  label-switched-path PE1toPE4 {
    to 198.51.100.4;
  }
  interface xe-0/0/2.0;
  interface xe-0/0/3.0;
}
bgp {
  group IBGP {
    type internal;
    local-address 198.51.100.1;
    family evpn {
      signaling;
    }
    neighbor 198.51.100.2;
    neighbor 198.51.100.3;
    neighbor 198.51.100.4;
  }
}
ospf {
  area 0.0.0.0 {
    interface xe-0/0/2.0;
    interface xe-0/0/3.0;
    interface lo0.0;
```

```

    }
}

```

```

[edit ]
user@PE1# show policy-options
policy-statement LB {
    then {
        load-balance per-packet;
    }
}

```

```

[edit ]
user@PE1# show routing-instances
EVPN-VPWS {
    instance-type evpn-vpws;
    interface ae0.100;
    route-distinguisher 198.51.100.1:11;
    vrf-target target:100:11;
    protocols {
        evpn {
            interface ae0.100 {
                vpws-service-id {
                    local 1111;
                    remote 9999;
                }
            }
        }
    }
}

```

If you are done configuring the device, enter `commit` from the configuration mode.

Verification

IN THIS SECTION

- [Verifying Aggregated Ethernet Interfaces and LACP | 1086](#)
- [Verifying OSPF | 1087](#)

- [Verifying BGP | 1088](#)
- [Verifying MPLS | 1089](#)
- [Verifying the VPWS | 1091](#)
- [Verifying Route Exchange and ESI Autodiscovery | 1092](#)
- [Verifying Local EVPN Table Route Information | 1094](#)

Confirm that the configuration is working properly.

Verifying Aggregated Ethernet Interfaces and LACP

Purpose

Verify that the AE interfaces are up and properly.

Action

Verify that the AE interfaces are up, and LACP connections are established between the PE devices and their related CE device..

```
user@CE1> show lacp interfaces extensive
```

```
Aggregated interface: ae0
```

LACP state:	Role	Exp	Def	Dist	Col	Syn	Aggr	Timeout	Activity
xe-0/0/0	Actor	No	No	Yes	Yes	Yes	Yes	Fast	Active
xe-0/0/0	Partner	No	No	Yes	Yes	Yes	Yes	Fast	Active
xe-0/0/2	Actor	No	No	Yes	Yes	Yes	Yes	Fast	Active
xe-0/0/2	Partner	No	No	Yes	Yes	Yes	Yes	Fast	Active

LACP protocol:	Receive State	Transmit State	Mux State
xe-0/0/0	Current	Fast periodic	Collecting distributing
xe-0/0/2	Current	Fast periodic	Collecting distributing

LACP info:	Role	System priority	System identifier	Port priority	Port number	Port key
xe-0/0/0	Actor	127	44:f4:77:99:e3:c0	127	1	1
xe-0/0/0	Partner	127	00:00:00:00:00:01	127	1	1
xe-0/0/2	Actor	127	44:f4:77:99:e3:c0	127	2	1
xe-0/0/2	Partner	127	00:00:00:00:00:01	127	1	1

```
user@PE1> show lacp interfaces extensive
```

```
Aggregated interface: ae0
```

LACP state:	Role	Exp	Def	Dist	Col	Syn	Aggr	Timeout	Activity
xe-0/0/0	Actor	No	No	Yes	Yes	Yes	Yes	Fast	Active
xe-0/0/0	Partner	No	No	Yes	Yes	Yes	Yes	Fast	Active

LACP protocol:	Receive State	Transmit State	Mux State
xe-0/0/0	Current	Fast periodic	Collecting distributing

LACP info:	Role	System priority	System identifier	Port priority	Port number	Port key
xe-0/0/0	Actor	127	00:00:00:00:00:01	127	1	1
xe-0/0/0	Partner	127	44:f4:77:99:e3:c0	127	1	1

```
user@PE2> show lacp interfaces extensive
```

```
Aggregated interface: ae0
```

LACP state:	Role	Exp	Def	Dist	Col	Syn	Aggr	Timeout	Activity
xe-0/0/2	Actor	No	No	Yes	Yes	Yes	Yes	Fast	Active
xe-0/0/2	Partner	No	No	Yes	Yes	Yes	Yes	Fast	Active

LACP protocol:	Receive State	Transmit State	Mux State
xe-0/0/2	Current	Fast periodic	Collecting distributing

LACP info:	Role	System priority	System identifier	Port priority	Port number	Port key
xe-0/0/2	Actor	127	00:00:00:00:00:01	127	1	1
xe-0/0/2	Partner	127	44:f4:77:99:e3:c0	127	2	1

Meaning

The AE interface on each device is up, and there are active LACP connections between the CE device and its local PE devices. Note also that the system ID configured on the PE devices, 00:00:00:00:00:01 (and shown on the PE device outputs as the Actor), matches the Partner system ID value on the CE device.

Verifying OSPF

Purpose

Verify that OSPF is working properly.

Action

Verify that OSPF has adjacencies established with its remote neighbors.

```

user@PE1> show ospf neighbor
Address          Interface          State   ID                Pri  Dead
10.0.1.2 #PE3# xe-0/0/2.0    Full   198.51.100.3     128   37
10.0.2.2 #PE4# xe-0/0/3.0    Full   198.51.100.4     128   33

user@PE3> show ospf neighbor
Address          Interface          State   ID                Pri  Dead
10.0.1.1 #PE1# xe-0/0/0.0    Full   198.51.100.1     128   34
10.0.3.1 #PE2# xe-0/0/1.0    Full   198.51.100.2     128   34

```

Meaning

Adjacencies have been established with remote neighbors.

Verifying BGP

Purpose

Verify that BGP is working properly.

Action

Verify that IBGP has peerings established with its neighbors using EVPN signaling.

```

user@PE1> show bgp summary
Groups: 1 Peers: 3 Down peers: 0
Table          Tot Paths  Act Paths Suppressed  History Damp State   Pending
bgp.evpn.0
              7          4          0          0          0          0
Peer          AS      InPkt    OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
198.51.100.2 #PE2#  65000    12        5        0        0      3:03 Establ
  bgp.evpn.0: 0/3/3/0
  EVPN-VPWS.evpn.0: 0/2/2/0
  __default_evpn__.evpn.0: 0/1/1/0

```

```

198.51.100.3 #PE3# 65000 11 9 0 0 3:03 Establ
  bgp.evpn.0: 2/2/2/0
  EVPN-VPWS.evpn.0: 2/2/2/0
  __default_evpn__.evpn.0: 0/0/0/0
198.51.100.4 #PE4# 65000 9 4 0 0 1:56 Establ
  bgp.evpn.0: 2/2/2/0
  EVPN-VPWS.evpn.0: 2/2/2/0
  __default_evpn__.evpn.0: 0/0/0/0

user@PE3> show bgp summary
Groups: 1 Peers: 3 Down peers: 0
Table Tot Paths Act Paths Suppressed History Damp State Pending
bgp.evpn.0
          7          4          0          0          0          0
Peer          AS    InPkt    OutPkt    OutQ    Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
198.51.100.1 #PE1# 65000 17 11 0 0 5:09 Establ
  bgp.evpn.0: 2/2/2/0
  EVPN-VPWS.evpn.0: 2/2/2/0
  __default_evpn__.evpn.0: 0/0/0/0
198.51.100.2 #PE2# 65000 17 14 0 0 5:04 Establ
  bgp.evpn.0: 2/2/2/0
  EVPN-VPWS.evpn.0: 2/2/2/0
  __default_evpn__.evpn.0: 0/0/0/0
198.51.100.4 #PE34 65000 13 8 0 0 4:02 Establ
  bgp.evpn.0: 0/3/3/0
  EVPN-VPWS.evpn.0: 0/2/2/0
  __default_evpn__.evpn.0: 0/1/1/0

```

Meaning

EVPN-signaled IBGP peerings have been established with all neighbors.

Verifying MPLS

Purpose

Verify that MPLS is working properly.

Action

Verify that MPLS LSPs are established with remote neighbors.

```

user@PE1> show mpls lsp
Ingress LSP: 2 sessions

```

To	From	State	Rt	P	ActivePath	LSPname
198.51.100.3	198.51.100.1	Up	0	*		PE1toPE3
198.51.100.4	198.51.100.1	Up	0	*		PE1toPE4

```

Total 2 displayed, Up 2, Down 0

Egress LSP: 2 sessions

```

To	From	State	Rt	Style	Labelin	Labelout	LSPname
198.51.100.1	198.51.100.4	Up	0	1 FF	3	-	PE4toPE1
198.51.100.1	198.51.100.3	Up	0	1 FF	3	-	PE3toPE1

```

Total 2 displayed, Up 2, Down 0

Transit LSP: 0 sessions
Total 0 displayed, Up 0, Down 0

user@PE3> show mpls lsp
Ingress LSP: 2 sessions

```

To	From	State	Rt	P	ActivePath	LSPname
198.51.100.1	198.51.100.3	Up	0	*		PE3toPE1
198.51.100.2	198.51.100.3	Up	0	*		PE3toPE2

```

Total 2 displayed, Up 2, Down 0

Egress LSP: 2 sessions

```

To	From	State	Rt	Style	Labelin	Labelout	LSPname
198.51.100.3	198.51.100.2	Up	0	1 FF	3	-	PE2toPE3
198.51.100.3	198.51.100.1	Up	0	1 FF	3	-	PE1toPE3

```

Total 2 displayed, Up 2, Down 0

Transit LSP: 0 sessions
Total 0 displayed, Up 0, Down 0

```

Meaning

LSPs have been established with remote neighbors.

Verifying the VPWS

Purpose

Verify that the VPWS is established.

Action

Verify that the PE devices have exchanged and learned service identifiers, and established the VPWS.

```
user@PE1> show evpn vpws-instance
```

```
Instance: EVPN-VPWS
```

```
Route Distinguisher: 198.51.100.1:11
```

```
Number of local interfaces: 1 (1 up)
```

Interface name	ESI	Mode	Role	Status
ae0.100	00:11:11:11:11:11:11:11:11	all-active	Primary	Up
Local SID: 1111 Advertised Label: 300496				
Remote SID: 9999				
PE addr	ESI	Label	Mode	Role
TS	Status			
198.51.100.3	00:99:99:99:99:99:99:99:99	300656	all-active	Primary
2017-05-12 07:30:01.863	Resolved			
198.51.100.4	00:99:99:99:99:99:99:99:99	300704	all-active	Primary
2017-05-12 07:31:23.804	Resolved			

```
Fast Convergence Information
```

```
ESI: 00:99:99:99:99:99:99:99:99 Number of PE nodes: 2
```

```
PE: 198.51.100.3 #PE3#
```

```
Advertised SID: 9999
```

```
PE: 198.51.100.4 #PE4#
```

```
Advertised SID: 9999
```

```
user@PE3> show evpn vpws-instance
```

```
Instance: EVPN-VPWS
```

```
Route Distinguisher: 198.51.100.3:11
```

```
Number of local interfaces: 1 (1 up)
```

Interface name	ESI	Mode	Role	Status
ae0.100	00:99:99:99:99:99:99:99:99	all-active	Primary	Up
Local SID: 9999 Advertised Label: 300656				

```

Remote SID: 1111
      PE addr      ESI      Label  Mode      Role
TS      Status
      198.51.100.1  00:11:11:11:11:11:11:11:11:11:11:11:11:11:11:11 300496 all-active Primary
2017-05-12 07:30:25.702 Resolved
      198.51.100.2  00:11:11:11:11:11:11:11:11:11:11:11:11:11:11:11 300560 all-active Primary
2017-05-12 07:30:25.711 Resolved

Fast Convergence Information
ESI: 00:11:11:11:11:11:11:11:11:11:11:11:11:11:11:11 Number of PE nodes: 2
PE: 198.51.100.1    #PE1#
    Advertised SID: 1111
PE: 198.51.100.2    #PE2#
    Advertised SID: 1111

```

Meaning

The PE devices on each side of the network have advertised their service identifiers, and received the identifiers from their remote neighbors. The VPWS is established.

Verifying Route Exchange and ESI Autodiscovery

Purpose

Verify that EVPN signaling is working properly.

Action

Verify that autodiscovery information is being shared across the VPWS.

```

user@PE1> show route table bgp.evpn.0

bgp.evpn.0: 7 destinations, 7 routes (4 active, 0 holddown, 3 hidden)
+ = Active Route, - = Last Active, * = Both

1:198.51.100.3:0::9999999999999999::FFFF:FFFF/304 AD/ESI
    *[BGP/170] 00:03:17, localpref 100, from 198.51.100.3
    AS path: I, validation-state: unverified
    > to 10.0.1.2 via xe-0/0/2.0, label-switched-path PE1toPE3
1:198.51.100.3:11::9999999999999999::9999/304 AD/EVI
    *[BGP/170] 00:03:18, localpref 100, from 198.51.100.3

```

```

        AS path: I, validation-state: unverified
        > to 10.0.1.2 via xe-0/0/2.0, label-switched-path PE1toPE3
1:198.51.100.4:0::9999999999999999::FFFF:FFFF/304 AD/ESI
        *[BGP/170] 00:01:56, localpref 100, from 198.51.100.4
        AS path: I, validation-state: unverified
        > to 10.0.2.2 via xe-0/0/3.0, label-switched-path PE1toPE4
1:198.51.100.4:11::9999999999999999::9999/304 AD/EVI
        *[BGP/170] 00:01:56, localpref 100, from 198.51.100.4
        AS path: I, validation-state: unverified
        > to 10.0.2.2 via xe-0/0/3.0, label-switched-path PE1toPE4

```

```
user@PE3> show route table bgp.evpn.0
```

```
bgp.evpn.0: 7 destinations, 7 routes (4 active, 0 holddown, 3 hidden)
```

```
+ = Active Route, - = Last Active, * = Both
```

```

1:198.51.100.1:0::1111111111111111::FFFF:FFFF/304 AD/ESI
        *[BGP/170] 00:04:53, localpref 100, from 198.51.100.1
        AS path: I, validation-state: unverified
        > to 10.0.1.1 via xe-0/0/0.0, label-switched-path PE3toPE1
1:198.51.100.1:11::1111111111111111::1111/304 AD/EVI
        *[BGP/170] 00:04:53, localpref 100, from 198.51.100.1
        AS path: I, validation-state: unverified
        > to 10.0.1.1 via xe-0/0/0.0, label-switched-path PE3toPE1
1:198.51.100.2:0::1111111111111111::FFFF:FFFF/304 AD/ESI
        *[BGP/170] 00:04:53, localpref 100, from 198.51.100.2
        AS path: I, validation-state: unverified
        > to 10.0.3.1 via xe-0/0/1.0, label-switched-path PE3toPE2
1:198.51.100.2:11::1111111111111111::1111/304 AD/EVI
        *[BGP/170] 00:04:53, localpref 100, from 198.51.100.2
        AS path: I, validation-state: unverified
        > to 10.0.3.1 via xe-0/0/1.0, label-switched-path PE3toPE2

```

Meaning

The outputs show the ESI routes being shared across the VPWS to the remote PE devices.

The routes beginning with 1:198.51.100.x:0:: are the per-Ethernet-segment autodiscovery Type 1 EVPN routes originating from the remote PE devices. The route distinguishers (RDs) are derived at the global level of the devices.

The routes beginning with 1:198.51.100.x:11:: are the per-EVI autodiscovery Type 1 EVPN routes from the remote PE devices. The RDs are taken from the remote PE devices' routing instances.

Verifying Local EVPN Table Route Information

Purpose

Verify that the local EVPN routing tables are being populated.

Action

Verify that both local and remote reachability information is being added into the EVPN table.

```

user@PE1> show route table EVPN-VPWS.evpn.0

EVPN-VPWS.evpn.0: 7 destinations, 7 routes (5 active, 0 holddown, 2 hidden)
+ = Active Route, - = Last Active, * = Both

1:198.51.100.1:11::1111111111111111::1111/304 AD/EVI
      *[EVPN/170] 00:06:55
      Indirect
1:198.51.100.3:0::9999999999999999::FFFF:FFFF/304 AD/ESI
      *[BGP/170] 00:03:24, localpref 100, from 198.51.100.3
      AS path: I, validation-state: unverified
      > to 10.0.1.2 via xe-0/0/2.0, label-switched-path PE1toPE3
1:198.51.100.3:11::9999999999999999::9999/304 AD/EVI
      *[BGP/170] 00:03:25, localpref 100, from 198.51.100.3
      AS path: I, validation-state: unverified
      > to 10.0.1.2 via xe-0/0/2.0, label-switched-path PE1toPE3
1:198.51.100.4:0::9999999999999999::FFFF:FFFF/304 AD/ESI
      *[BGP/170] 00:02:03, localpref 100, from 198.51.100.4
      AS path: I, validation-state: unverified
      > to 10.0.2.2 via xe-0/0/3.0, label-switched-path PE1toPE4
1:198.51.100.4:11::9999999999999999::9999/304 AD/EVI
      *[BGP/170] 00:02:03, localpref 100, from 198.51.100.4
      AS path: I, validation-state: unverified
      > to 10.0.2.2 via xe-0/0/3.0, label-switched-path PE1toPE4

user@PE3> show route table EVPN-VPWS.evpn.0

EVPN-VPWS.evpn.0: 7 destinations, 7 routes (5 active, 0 holddown, 2 hidden)

```

+ = Active Route, - = Last Active, * = Both

```

1:198.51.100.1:0::1111111111111111::FFFF:FFFF/304 AD/ESI
    *[BGP/170] 00:05:01, localpref 100, from 198.51.100.1
    AS path: I, validation-state: unverified
    > to 10.0.1.1 via xe-0/0/0.0, label-switched-path PE3toPE1
1:198.51.100.1:11::1111111111111111::1111/304 AD/EVI
    *[BGP/170] 00:05:01, localpref 100, from 198.51.100.1
    AS path: I, validation-state: unverified
    > to 10.0.1.1 via xe-0/0/0.0, label-switched-path PE3toPE1
1:198.51.100.2:0::1111111111111111::FFFF:FFFF/304 AD/ESI
    *[BGP/170] 00:05:01, localpref 100, from 198.51.100.2
    AS path: I, validation-state: unverified
    > to 10.0.3.1 via xe-0/0/1.0, label-switched-path PE3toPE2
1:198.51.100.2:11::1111111111111111::1111/304 AD/EVI
    *[BGP/170] 00:05:01, localpref 100, from 198.51.100.2
    AS path: I, validation-state: unverified
    > to 10.0.3.1 via xe-0/0/1.0, label-switched-path PE3toPE2
1:198.51.100.3:11::9999999999999999::9999/304 AD/EVI
    *[EVPN/170] 00:05:25
    Indirect

```

Meaning

In addition to the remote ESI routes being shared across the VPWS, as explained in the previous section, the EVPN table on each PE device also includes a local ESI route. This Type 1 route represents the locally configured Ethernet segment, and is derived from the locally configured RD and ESI values.

RELATED DOCUMENTATION

[EVPN Multihoming Overview | 96](#)

[Overview of VPWS with EVPN Signaling Mechanisms | 1046](#)

[Configuring VPWS with EVPN Signaling Mechanisms | 1068](#)

[vpws-service-id | 1665](#)

[show evpn vpws-instance | 1903](#)

FAT Flow Labels in EVPN-VPWS Routing Instances

IN THIS SECTION

- [How to Enable FAT Pseudowire Flow Labels in an EVPN-VPWS Instance | 1096](#)
- [Verify FAT Flow Labels Are Enabled | 1099](#)

FAT Flow Labels Overview introduces how LDP-signaled pseudowires in an MPLS network can use flow-aware transport (FAT) flow labels (defined in RFC 6391, *Flow-Aware Transport of Pseudowires over an MPLS Packet Switched Network*) to load-balance traffic in virtual private LAN service (VPLS) and virtual private wire service (VPWS) networks.

Starting in Junos OS Release 21.1R1, you can enable provider edge devices in an EVPN-MPLS network to use flow-aware transport (FAT) flow labels to load-balance traffic across pseudowires in an EVPN-VPWS routing instance. In this environment, the local and remote devices establish an EVPN connection between the local and remote provider edge (PE) devices using BGP signaling, and can use LDP or RSVP tunnels to create the pseudowire.

How to Enable FAT Pseudowire Flow Labels in an EVPN-VPWS Instance

You can enable FAT flow labels in a routing instance of type `evpn-vpws` for pseudowires associated with the routing instance.

This environment supports enabling FAT flow label push and pop operations on pseudowire traffic only with a static configuration. The devices don't actively use the signaling mechanism described in RFC 6391 to ensure both ends communicate that they can handle flow labels. As a result, you must use ["flow-label-transmit-static" on page 1579](#) and ["flow-label-receive-static" on page 1577](#) (instead of `flow-label-transmit` and `flow-label-receive`) on all of the PE routers that will transmit, receive, and load-balance traffic with flow labels.

NOTE: We advise that you plan to configure these options during a maintenance window on those devices.

You can configure FAT flow label push and pop operations on the device at either the global routing instance level or at the individual interface level, as follows:

1. Configure the EVPN routing instance of type evpn-vpws. For example:

```
set routing-instances VPWS-SH instance-type evpn-vpws
set routing-instances VPWS-SH protocols evpn interface ge-0/0/1.100 vpws-service-id local 100
set routing-instances VPWS-SH protocols evpn interface ge-0/0/1.100 vpws-service-id remote 200
set routing-instances VPWS-SH interface ge-0/0/1.100
set routing-instances VPWS-SH route-distinguisher 10.255.0.1:100
set routing-instances VPWS-SH vrf-target target:100:100
```

Verify the configuration:

```
user@device# show routing-instances

VPWS-SH {
  instance-type evpn-vpws;
  interface ge-0/0/1.100;
  route-distinguisher 10.255.0.1:100;
  vrf-target target:100:100;
  protocols {
    evpn {
      interface ge-0/0/1.100 {
        vpws-service-id {
          local 100;
          remote 200;
        }
      }
    }
  }
}
```

2. To configure FAT flow label push and pop operations at the evpn-vpws routing instance level:

```
set routing-instances <evpn-vpws-routing-instance-name> protocols evpn flow-label-transmit-
static
set routing-instances <evpn-vpws-routing-instance-name> protocols evpn flow-label-receive-
static
```

For example:

```
set routing-instances VPWS-SH protocols evpn flow-label-transmit-static
set routing-instances VPWS-SH protocols evpn flow-label-receive-static
```

Verify the configuration:

```
user@device# show routing-instances

VPWS-SH {
  instance-type evpn-vpws;
  interface ge-0/0/1.100;
  route-distinguisher 10.255.0.1:100;
  vrf-target target:100:100;
  protocols {
    evpn {
      interface ge-0/0/1.100 {
        vpws-service-id {
          local 100;
          remote 200;
        }
      }
      flow-label-transmit-static;
      flow-label-receive-static;
    }
  }
}
```

3. Alternatively, to enable a specific interface in the `evpn-vpws` routing instance to push and pop FAT flow labels:

```
set routing-instances <evpn-vpws-routing-instance-name> protocols evpn interface <interface-
name> flow-label-transmit-static
set routing-instances <evpn-vpws-routing-instance-name> protocols evpn interface <interface-
name> flow-label-receive-static
```

For example:

```
set routing-instances VPWS-SH protocols evpn interface ge-0/0/1.100 flow-label-transmit-static
set routing-instances VPWS-SH protocols evpn interface ge-0/0/1.100 flow-label-receive-static
```

Verify the configuration:

```
user@device# show routing-instances

VPWS-SH {
  instance-type evpn-vpws;
  interface ge-0/0/1.100;
  route-distinguisher 10.255.0.1:100;
  vrf-target target:100:100;
  protocols {
    evpn {
      interface ge-0/0/1.100 {
        vpws-service-id {
          local 100;
          remote 200;
        }
        flow-label-transmit-static;
        flow-label-receive-static;
      }
    }
  }
}
```

Verify FAT Flow Labels Are Enabled

You can enter the ["show evpn vpws-instance" on page 1903](#) CLI command to see if an evpn-vpws routing instance is configured to handle FAT flow labels. The output from this command displays **Yes** in the **Flow-Label-Tx** or **Flow-Label-Rx** output fields if you configured the device to insert or remove FAT flow labels on pseudowire traffic in the routing instance. These fields display **No** if FAT flow label operations are not enabled.

RELATED DOCUMENTATION

FAT Flow Labels Overview

[flow-label-receive-static](#) | 1577

[flow-label-transmit-static](#) | 1579

5

PART

EVPN-ETREE

[Overview](#) | [1102](#)

[Configuring EVPN-ETREE](#) | [1106](#)

Overview

IN THIS CHAPTER

- [EVPN-ETREE Overview | 1102](#)

EVPN-ETREE Overview

IN THIS SECTION

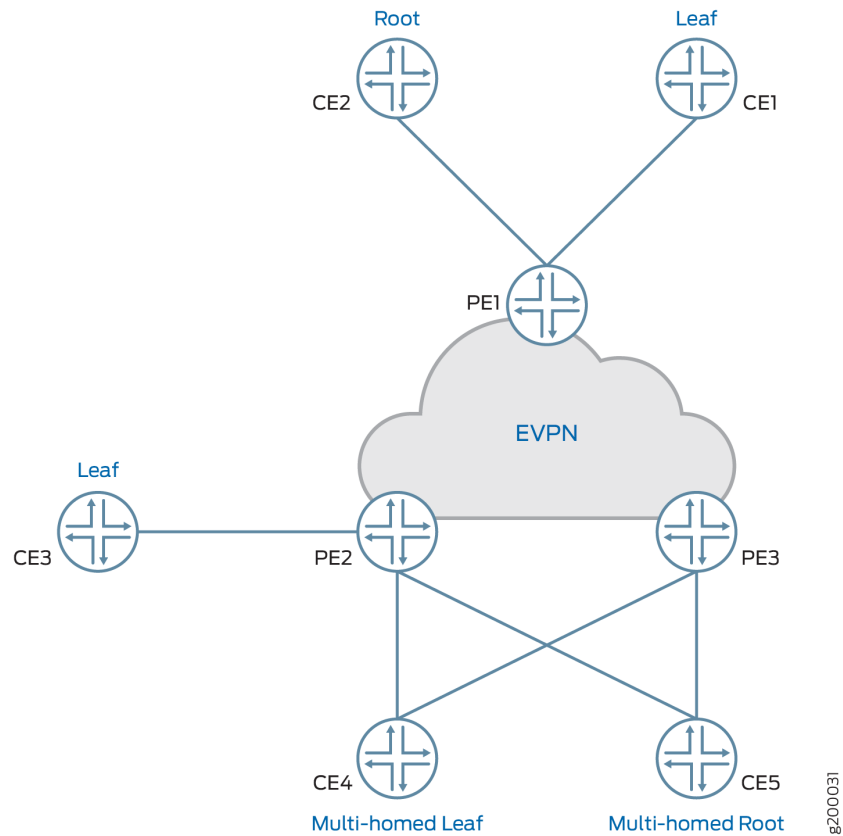
- [NSR and Unified ISSU Support for EVPN-ETREE | 1104](#)

The EVPN-ETREE service is a VPN service where each attachment circuit is designated as either root or leaf. The EVPN E-Tree feature implements E-Tree service as defined by the Metro Ethernet Forum (MEF) in draft-sajassi-l2vpn-evpn-etree-03. The E-Tree service is a rooted-multipoint service that is supported only with EVPN over MPLS in the core. The EVPN E-Tree feature provides a way to categorize the interfaces as either “root” or “leaf” in a routing instance. In an EVPN E-Tree service, each Customer Edge devices attached the service is either a root or a leaf. The EVPN E-Tree service adheres to the following forwarding rules:

- A leaf can send or receive traffic only from a root.
- A root can send traffic to another root or any of the leaves.

- A leaf or root can be connected to provider edge (PE) devices in singlehoming mode or multihoming mode.

Figure 117: EVPN E-TREE Service



The EVPN ETREE service has all the benefits of EVPN such as active-active multihoming, load balancing loop detection for E-Tree

In an EVPN ETREE service, the forwarding rule depends on the traffic source and destination. Table 1 shows the forwarding rules within the ETREE service.

Type of Traffic	Allowed/Not-Allowed	Filtering Location
Known Unicast Traffic from Root to Root	Allowed	
Known Unicast Traffic from Root to Leaf	Allowed	

BUM Traffic from Root to Root	Allowed	
BUM Traffic from Root to Leaf	Allowed	
Known Unicast Traffic from Leaf to Leaf	Not Allowed	At the ingress Packet Forwarding Engine
Known Unicast Traffic from Leaf to Root	Allowed	
BUM Traffic from Leaf to Leaf	Not Allowed	At the Egress Packet Forwarding Engine
BUM Traffic from Leaf to Root	Allowed	

If you do not configure a role for an interface, it will be assigned the role of “root” by default. All leaf interfaces are assigned a new mesh group with no local switching set to TRUE. This enables the ingress filtering for unicast traffic and all the leaf-to-leaf traffic will get dropped at ingress leaf interface. For BUM traffic, the filtering will happen at the egress Provider Edge based on the root/leaf label being carried in the packet.

NSR and Unified ISSU Support for EVPN-ETREE

Nonstop active routing (NSR) and graceful Routing Engine switchover (GRES) minimize traffic loss when there is a Routing Engine switchover. When a Routing Engine fails, NSR and GRES enable a routing platform with redundant Routing Engines to switch over from a primary Routing Engine to a backup Routing Engine and continue forwarding packets. Unified in-service software upgrade (ISSU) allows you to upgrade your Junos OS software on your MX Series router with no disruption on the control plane and with minimal disruption of traffic. Both GRES and NSR must be enabled to use unified ISSU.

To enable GRES, include the `graceful-switchover` statement at the `[edit chassis redundancy]` hierarchy level.

Junos OS mirrors essential data when NSR is enabled. For EVPN ETREE, the local EVPN ETREE leaf label that is advertised to other PE as part of the ETREE extended community will be mirrored on the standby Routing Engine. For information on other mirrored data and NSR data flow, see ["NSR and Unified ISSU Support for EVPN " on page 383](#).

To enable NSR, include the `nonstop-routing` statement at the `[edit routing-options]` hierarchy level and the `commit synchronize` statement at the `[edit system]` hierarchy level.

RELATED DOCUMENTATION

| [Example: Configuring EVPN ETree SERVICE](#)

Configuring EVPN-ETREE

IN THIS CHAPTER

- [Example: Configuring EVPN-ETREE SERVICE | 1106](#)

Example: Configuring EVPN-ETREE SERVICE

IN THIS SECTION

- [Requirements | 1106](#)
- [Overview | 1107](#)
- [Configuration | 1107](#)
- [Verification | 1116](#)

This example shows how to configure EVPN-ETREE service.

Requirements

This example uses the following hardware and software components:

- Three MX Series 5G Universal Routing Platforms configured as provider edge (PE) routers.
- Three customer edge (CE) routers, each connected to the PE routers.
- Junos OS Release 17.2 or later running on all the PE routers.

Before you begin:

- Configure the device interfaces.
- Configure an IGP, such as OSPF, on all the devices.

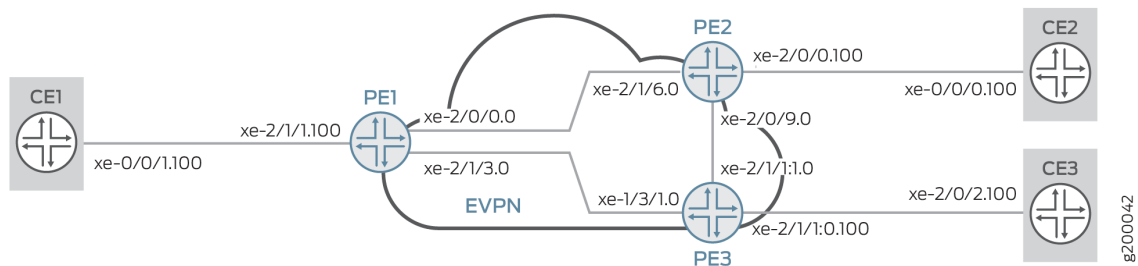
- Establish a BGP session between the PE devices.
- Configure MPLS and LDP on the PE devices.

Overview

The EVPN-ETree service is a VPN service where each attachment circuit is designated as either root or leaf. The E-Tree service is a rooted-multipoint service that is supported only with EVPN over MPLS in the core. In an EVPN E-Tree service, each Customer Edge devices attached the service is either a root or a leaf. The EVPN E-Tree service adheres to the following forwarding rules:

- A leaf can send or receive traffic only from a root.
- A root can send traffic to another root or any of the leaves.
- A leaf or root can be connected to provider edge (PE) devices in singlehoming mode or multihoming mode.

Figure 118: EVPN E-TREE Service



Configuration

IN THIS SECTION

- [CLI Quick Configuration | 1108](#)
- [Procedure | 1111](#)
- [Results | 1114](#)

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the [edit] hierarchy level, and then enter commit from configuration mode.

CE1

```
set interfaces xe-0/0/1 vlan-tagging
set interfaces xe-0/0/1 unit 100 vlan-id 100
set interfaces xe-0/0/1 unit 100 family inet address 10.100.0.1/24
```

PE1

```
set interfaces xe-2/0/0 unit 0 family inet address 10.0.0.1/30
set interfaces xe-2/0/0 unit 0 family mpls
set interfaces xe-2/1/3 unit 0 family inet address 10.0.0.5/30
set interfaces xe-2/1/3 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.0.1/32 primary
set interfaces lo0 unit 0 family inet address 10.255.0.1/32 preferred
set interfaces xe-2/1/1 flexible-vlan-tagging
set interfaces xe-2/1/1 encapsulation flexible-ethernet-services
set interfaces xe-2/1/1 unit 100 encapsulation vlan-bridge
set interfaces xe-2/1/1 unit 100 vlan-id 100
set interfaces xe-2/1/1 unit 100 etree-ac-role root
set routing-options router-id 10.255.0.1
set routing-options autonomous-system 65000
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group evpn local-address 10.255.0.1
set protocols bgp group evpn family evpn signaling
set protocols bgp group evpn peer-as 65000
set protocols bgp group evpn local-as 65000
set protocols bgp group evpn neighbor 10.255.0.2
set protocols bgp group evpn neighbor 10.255.0.3
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set routing-instances evpna instance-type evpn
set routing-instances evpna vlan-id 100
set routing-instances evpna interface xe-2/1/1.100
```

```

set routing-instances evpna route-distinguisher 10.255.0.1:100
set routing-instances evpna vrf-target target:65000:100
set routing-instances evpna protocols evpn interface xe-2/1/1.100
set routing-instances evpna protocols evpn evpn-etree

```

PE2

```

set interfaces xe-2/1/6 unit 0 family inet address 10.0.0.2/30
set interfaces xe-2/1/6 unit 0 family mpls
set interfaces xe-2/0/9 unit 0 family inet address 10.0.0.9/30
set interfaces xe-2/0/9 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.0.2/32 primary
set interfaces lo0 unit 0 family inet address 10.255.0.2/32 preferred
set interfaces xe-2/0/0 flexible-vlan-tagging
set interfaces xe-2/0/0 encapsulation flexible-ethernet-services
set interfaces xe-2/0/0 unit 100 encapsulation vlan-bridge
set interfaces xe-2/0/0 unit 100 vlan-id 100
set interfaces xe-2/0/0 unit 100 etree-ac-role leaf
set routing-options router-id 10.255.0.2
set routing-options autonomous-system 65000
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group evpn local-address 10.255.0.2
set protocols bgp group evpn family evpn signaling
set protocols bgp group evpn peer-as 65000
set protocols bgp group evpn local-as 65000
set protocols bgp group evpn neighbor 10.255.0.1
set protocols bgp group evpn neighbor 10.255.0.3
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set routing-instances evpna instance-type evpn
set routing-instances evpna vlan-id 100
set routing-instances evpna interface xe-2/0/0.100
set routing-instances evpna route-distinguisher 10.255.0.2:100
set routing-instances evpna vrf-target target:65000:100
set routing-instances evpna protocols evpn interface xe-2/0/0.100
set routing-instances evpna protocols evpn evpn-etree

```


PE3

```

set interfaces xe-1/3/1 unit 0 family inet address 10.0.0.6/30
set interfaces xe-1/3/1 unit 0 family mpls
set interfaces xe-2/1/1:1 unit 0 family inet address 10.0.0.10/30
set interfaces xe-2/1/1:1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.0.3/32 primary
set interfaces lo0 unit 0 family inet address 10.255.0.3/32 preferred
set interfaces xe-2/1/1:0 flexible-vlan-tagging
set interfaces xe-2/1/1:0 encapsulation flexible-ethernet-services
set interfaces xe-2/1/1:0 unit 100 encapsulation vlan-bridge
set interfaces xe-2/1/1:0 unit 100 vlan-id 100
set interfaces xe-2/1/1:0 unit 100 etree-ac-role leaf
set routing-options router-id 10.255.0.3
set routing-options autonomous-system 65000
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group evpn local-address 10.255.0.3
set protocols bgp group evpn family evpn signaling
set protocols bgp group evpn peer-as 65000
set protocols bgp group evpn local-as 65000
set protocols bgp group evpn neighbor 10.255.0.1
set protocols bgp group evpn neighbor 10.255.0.2
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set routing-instances evpna instance-type evpn
set routing-instances evpna vlan-id 100
set routing-instances evpna interface xe-2/1/1:0.100
set routing-instances evpna route-distinguisher 10.255.0.3:100
set routing-instances evpna vrf-target target:65000:100
set routing-instances evpna protocols evpn interface xe-2/1/1:0.100
set routing-instances evpna protocols evpn evpn-etree

```

CE2

```

set interfaces xe-0/0/0 vlan-tagging
set interfaces xe-0/0/0 unit 100 vlan-id 100
set interfaces xe-0/0/0 unit 100 family inet address 10.100.0.2/24

```

CE3

```
set interfaces xe-2/0/2 vlan-tagging
set interfaces xe-2/0/2 unit 100 vlan-id 100
set interfaces xe-2/0/2 unit 100 family inet address 10.100.0.3/24
```

Procedure

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure Router PE1:

NOTE: Repeat this procedure for Routers PE2 and PE3, after modifying the appropriate interface names, addresses, and other parameters.

1. Configure Router PE1 interfaces.

```
[edit interfaces]
user@PE1#set xe-2/0/0 unit 0 family inet address 10.0.0.1/30
user@PE1#set xe-2/0/0 unit 0 family mpls
user@PE1#set xe-2/1/3 unit 0 family inet address 10.0.0.5/30
user@PE1#set xe-2/1/3 unit 0 family mpls
user@PE1#set lo0 unit 0 family inet address 10.255.0.1/32 primary
user@PE1#set lo0 unit 0 family inet address 10.255.0.1/32 preferred
user@PE1#set xe-2/1/1 flexible-vlan-tagging
user@PE1#set xe-2/1/1 encapsulation flexible-ethernet-services
user@PE1#set xe-2/1/1 unit 100 encapsulation vlan-bridge
user@PE1#set xe-2/1/1 unit 100 vlan-id 100
```

2. Assign the interface as leaf or root.

```
user@PE1#[edit interfaces]
set xe-2/1/1 unit 100 etree-ac-role root
```

3. Set the router ID and autonomous system number for Router PE1.

```
[edit routing-options]
user@PE1#set routing-options router-id 10.255.0.1
user@PE1#set routing-options autonomous-system 65000
```

4. Enable LDP on all the interfaces of Router PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set ldp interface all
user@PE1# set ldp interface fxp0.0 disable
```

5. Assign local and neighbor addresses to the BGP group for Router PE1 to peer with Routers PE2 and PE3.

```
[edit protocols]
user@PE1#set bgp group evpn local-address 10.255.0.1
user@PE1#set bgp group evpn neighbor 10.255.0.2
user@PE1#set bgp group evpn neighbor 10.255.0.3
```

6. Set up the local and peer autonomous systems.

```
user@PE1#set protocols bgp group evpn peer-as 65000
user@PE1#set protocols bgp group evpn local-as 65000
```

7. Include the EVPN signaling Network Layer Reachability Information (NLRI) to the bgp BGP group.

```
[edit protocols]
user@PE1#set bgp group evpn family evpn signaling
```

8. Configure OSPF on all the interfaces of Router PE1, excluding the management interface.

```
[edit protocols]
user@PE1#set ospf area 0.0.0.0 interface all
user@PE1#set ospf area 0.0.0.0 interface fxp0.0 disable
```

9. Configure MPLS on all the interfaces of Router PE1, excluding the management interface.

```
[edit protocols]
user@PE1#set mpls interface all
user@PE1#set mpls interface fxp0.0 disable
```

10. Configure the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpna instance-type evpn
```

11. Set the VLAN identifier for the bridging domain in the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna vlan-id 100
```

12. Configure the interface name for the evpna routing instance.

```
[edit routing-instances]
user@PE1#set evpna interface xe-2/1/1.100
```

13. Configure the route distinguisher for the evpna routing instance.

```
[edit routing-instances]
user@PE1#set evpna route-distinguisher 10.255.0.1:100
```

14. Assign the interface name that connects the PE1 site to the VPN.

```
[edit routing-instances]
user@PE1#set evpna protocols evpn interface xe-2/1/1.100
```

15. configure Ethernet VPN E-TREE service on PE1.

```
[edit routing-instances]
user@PE1#set evpna protocols evpn evpn-etree
```

16. Configure the VPN routing and forwarding (VRF) target community for the evpna routing instance.

```
[edit routing-instances]
user@PE1#set evpna vrf-target target:65000:100
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show routing-options`, and `show routing-instances` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1 show interfaces
xe-2/0/0 {
  unit 0 {
    family inet {
      address 10.0.0.1/30;
    }
    family mpls;
  }
}
xe-2/1/3 {
  unit 0 {
    family inet {
      address 10.0.0.5/30;
    }
    family mpls;
  }
}
lo0 {
  unit 0 {
    family inet {
      address 10.255.0.1/32 {
        primary;
        preferred;
      }
    }
  }
}
xe-2/1/1 {
  flexible-vlan-tagging;
  encapsulation flexible-ethernet-services;
```

```

    unit 100 {
        encapsulation vlan-bridge;
        vlan-id 100;
        etree-ac-role root;
    }
}

```

```

user@PE1 show routing-options
router-id 10.255.0.1;
autonomous-system 65000;

```

```

user@PE1 show protocols
mpls {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
bgp {
    group evpn {
        local-address 10.255.0.1;
        family evpn {
            signaling;
        }
        peer-as 65000;
        local-as 65000;
        neighbor 10.255.0.2;
        neighbor 10.255.0.3;
    }
}
ospf {
    area 0.0.0.0 {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
}
ldp {
    interface all;
}

```

```

interface fxp0.0 {
    disable;
}
}

```

```

user@PE1 #show routing-instances
evpna {
    instance-type evpn;
    vlan-id 100;
    interface xe-2/1/1.100;
    route-distinguisher 10.255.0.1:100;
    vrf-target target:65000:100;
    protocols {
        evpn {
            interface xe-2/1/1.100;
            evpn-etree;
        }
    }
}

```

Verification

IN THIS SECTION

- [Verifying the EVPN Instance Status | 1117](#)
- [Verifying local and remote MAC property | 1118](#)
- [Verifying EVPN ETREE Instances property | 1119](#)
- [Verifying traffic between leaf and root | 1120](#)
- [Verifying traffic flow between leaf and leaf is not allowed | 1120](#)

Confirm that the configuration is working properly.

Verifying the EVPN Instance Status

Purpose

Verify the EVPN routing instances and their status.

Action

From operational mode, run the `show evpn instance extensive` command.

```

user@PE1>show evpn instance extensive
Instance: __default_evpn__
  Route Distinguisher: 10.255.0.1:0
  Number of bridge domains: 0
  Number of neighbors: 0

Instance: evpna
  Route Distinguisher: 10.255.0.1:100
  VLAN ID: 100
  Per-instance MAC route label: 16
  Etree Leaf label: 20
  MAC database status
    Local Remote
  MAC advertisements:          1      1
  MAC+IP advertisements:      0      0
  Default gateway MAC advertisements: 0      0
  Number of local interfaces: 1 (1 up)
    Interface name  ESI                               Mode          Status    AC-Role
  xe-2/1/1.100    00:00:00:00:00:00:00:00:00 single-homed   Up        Root
  Number of IRB interfaces: 0 (0 up)
  Number of bridge domains: 1
    VLAN  Domain ID  Intfs / up  IRB intf  Mode          MAC sync  IM route label  SG
  100      100      1    1      Extended    Enabled   30
  Disabled
  Number of neighbors: 2
    Address      MAC    MAC+IP    AD    IM    ES Leaf-label
  10.255.0.2    0      0      1    1    0      20
  10.255.0.3    1      0      1    1    0      20
  Number of ethernet segments: 0

```


Meaning

The output provides the following information:

- List of EVPN and virtual switch routing instances
- Mode of operation of each interface
- Neighbors of each routing instance
- Number of different routes received from each neighbor
- Number of Ethernet segments on each routing instance
- VLAN ID and MAC labels for each routing instance

Verifying local and remote MAC property

Purpose

Verify EVPN MAC table information.

Action

From operational mode, run the `show evpn mac-table` command.

```
user@PE1>show evpn mac-table
MAC flags      (S -static MAC, D -dynamic MAC, L -locally learned, C -Control MAC
  O -OVSDB MAC, SE -Statistics enabled, NM -Non configured MAC, R -Remote PE MAC, P -Pinned
  MAC)
```

```
Routing instance : evpn_100
```

```
Bridging domain : __evpn_100__, VLAN : 100
```

MAC address	MAC flags	Logical interface	NH Index	MAC property
00:1d:b5:a2:15:2c	DC		1048579	Leaf
64:87:88:5f:05:c0	DC		1048578	Leaf
a8:d0:e5:54:38:21	D	xe-2/1/1.100		Root

Meaning

The output provides the following information:

- List of MAC addresses learned locally and via control-plane.
- Property of MAC whether it is learned on a leaf or root interface.

Verifying EVPN ETREE Instances property

Purpose

Verify EVPN ETREE Instances property.

Action

From operational mode, run the `show evpn instance evpna extensive` command.

```
user@PE1>show evpn instance evpna extensive
Instance: evpna
Route Distinguisher: 10.255.0.1:100
VLAN ID: 100
Per-instance MAC route label: 16
Etree Leaf label: 20
MAC database status
MAC advertisements:
MAC+IP advertisements:
Default gateway MAC advertisements:
Local Remote
0 0
0 0
0 0
Number of local interfaces: 1 (1 up)
Interface name ESI Mode Status AC-Role
xe-2/1/1.100 00:00:00:00:00:00:00:00:00 single-homed Up Root
Number of IRB interfaces: 0 (0 up)
Number of bridge domains: 1
VLAN Domain ID Intfs / up IRB intf Mode MAC sync IM route label SG
sync IM core nexthop
100 1 1 Extended Enabled 30
Disabled
Number of neighbors: 2
Address MAC MAC+IP AD IM ES Leaf-label
10.255.0.2 0 0 1 1 0 20
10.255.0.3 0 0 1 1 0 20
Number of ethernet segments: 0
```

Meaning

The output provides the following information:

- List the details of specific instance “evpna”.
- Lists the interfaces associated to this routing instance and its property (leaf or root).
- Lists the bridge-domains associated to this routing instance.
- Lists the neighbors and routes received.

Verifying traffic between leaf and root

Purpose

Verifying traffic flow between leaf and root

Action

From operational mode of CE2 (leaf), ping CE1 (root) to check traffic flow.

```
user@CE2> ping 10.100.0.1

PING 10.100.0.1 (10.100.0.1): 56 data bytes
64 bytes from 10.100.0.1: icmp_seq=0 ttl=64 time=1.063 ms
64 bytes from 10.100.0.1: icmp_seq=1 ttl=64 time=1.057 ms
64 bytes from 10.100.0.1: icmp_seq=2 ttl=64 time=1.038 ms
^C
--- 10.100.0.1 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 1.038/1.053/1.063/0.011 ms
```

Meaning

The output shows Ping is successful between CE2 (leaf) and CE1 (root).

Verifying traffic flow between leaf and leaf is not allowed

Purpose

Verifying traffic flow between leaf and leaf is not allowed.

Action

From operational mode of CE2 (leaf), ping CE3 (leaf) to check traffic flow.

```
user@CE2> ping 10.100.0.1

PING 10.100.0.3 (10.100.0.3): 56 data bytes
^C
--- 10.100.0.3 ping statistics ---
3 packets transmitted, 0 packets received, 100% packet loss
```

Meaning

The output shows Ping failed between CE2 and CE3 because traffic is not allowed between leaf and leaf interfaces.

RELATED DOCUMENTATION

| [EVPN-ETREE Overview](#)



Using EVPN for Interconnection

Interconnecting VXLAN Data Centers With EVPN | 1123

Interconnecting EVPN-VXLAN Data Centers Through an EVPN-MPLS WAN | 1163

Extending a Junos Fusion Enterprise Using EVPN-MPLS | 1359

Interconnecting VXLAN Data Centers With EVPN

IN THIS CHAPTER

- [VXLAN Data Center Interconnect Using EVPN Overview | 1123](#)
- [Example: Configuring VXLAN Data Center Interconnect Using EVPN | 1142](#)

VXLAN Data Center Interconnect Using EVPN Overview

IN THIS SECTION

- [Technology Overview of VXLAN-EVPN Integration for DCI | 1123](#)
- [Implementation Overview of VXLAN-EVPN Integration for DCI | 1136](#)
- [Supported and Unsupported Features for VXLAN DCI Using EVPN | 1141](#)

Starting in Junos OS Release 16.1, Ethernet VPN (EVPN) technology can be used to interconnect Virtual Extensible Local Area Network (VXLAN) networks over an MPLS/IP network to provide data center connectivity. This is done through Layer 2 intra-subnet connectivity and control-plane separation among the interconnected VXLAN networks.

The following sections describe the technology and implementation overview of integrating EVPN with VXLAN to be used as a data center interconnect (DCI) solution.

Technology Overview of VXLAN-EVPN Integration for DCI

The following sections provide a conceptual overview of VXLAN, EVPN, the need for their integration for DCI and the resulting benefits.

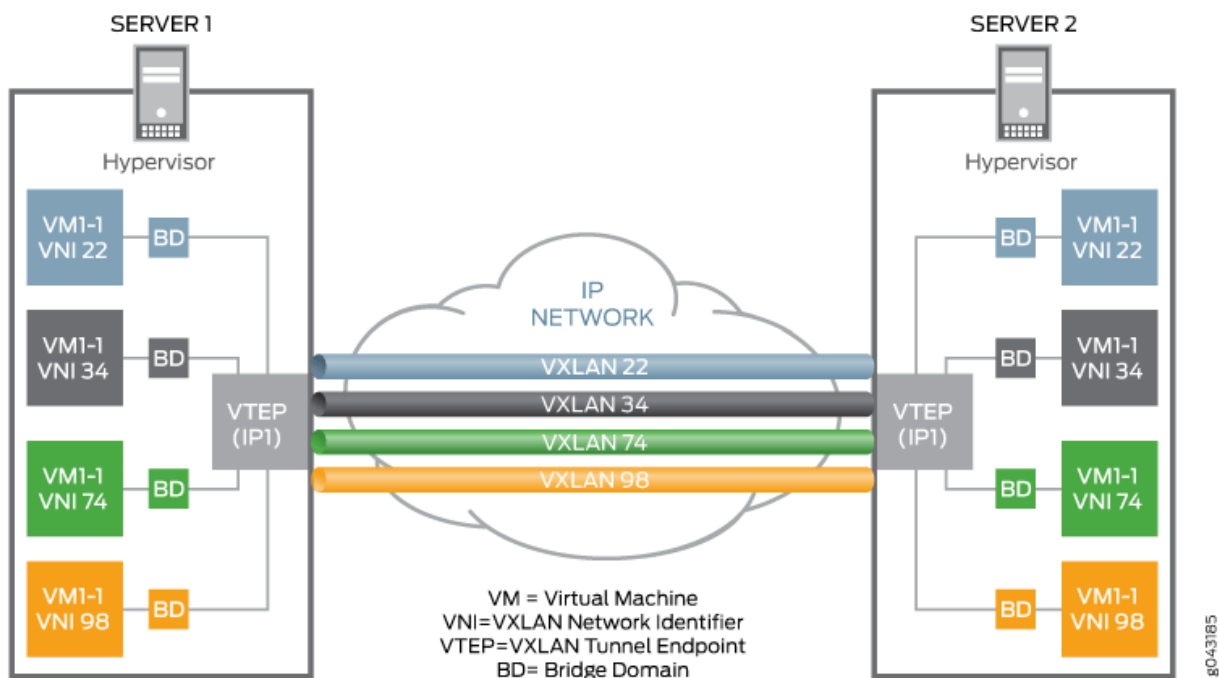
Understanding VXLAN

Virtual Extensible Local Area Network (VXLAN) is a Layer 3 encapsulation protocol that enables MX Series routers to push Layer 2 or Layer 3 packets through a VXLAN tunnel to a virtualized data center or the Internet. Communication is established between two virtual tunnel endpoints (VTEPs), which can be end hosts or network switches or routers, that encapsulate and de-encapsulate the virtual machine (VM) traffic into a VXLAN header.

VXLAN is often described as an overlay technology because it allows you to stretch Layer 2 connections over an intervening Layer 3 network by encapsulating (tunneling) Ethernet frames in a VXLAN packet that includes IP addresses. This feature of VXLAN addresses the requirements of a multi-tenant datacenter, where each tenant's VM might be sharing the physical server with other tenants that are distributed across physical servers within or across different data centers, by meeting the growing need to provide seamless Layer 2 connectivity between all the VMs owned by a tenant, in addition to isolating each tenant's traffic for security and potential MAC address overlaps.

VXLAN tunnels are created between the physical servers by the hypervisors. Since a physical server can host multiple tenants, each hypervisor creates multiple VXLAN tunnels.

Figure 119: VXLAN Overview



VXLAN is a technology that allows you to segment your networks (as VLANs do) but that also solves the scaling limitation of VLANs and provides benefits that VLANs cannot. Some of the important benefits of using VXLANs include:

- You can theoretically create as many as 16 million VXLANs in an administrative domain (as opposed to 4094 VLANs on a Juniper Networks device).

MX Series routers support as many as 32K VXLANs. This means that VXLANs provide network segmentation at the scale required by cloud builders to support very large numbers of tenants.

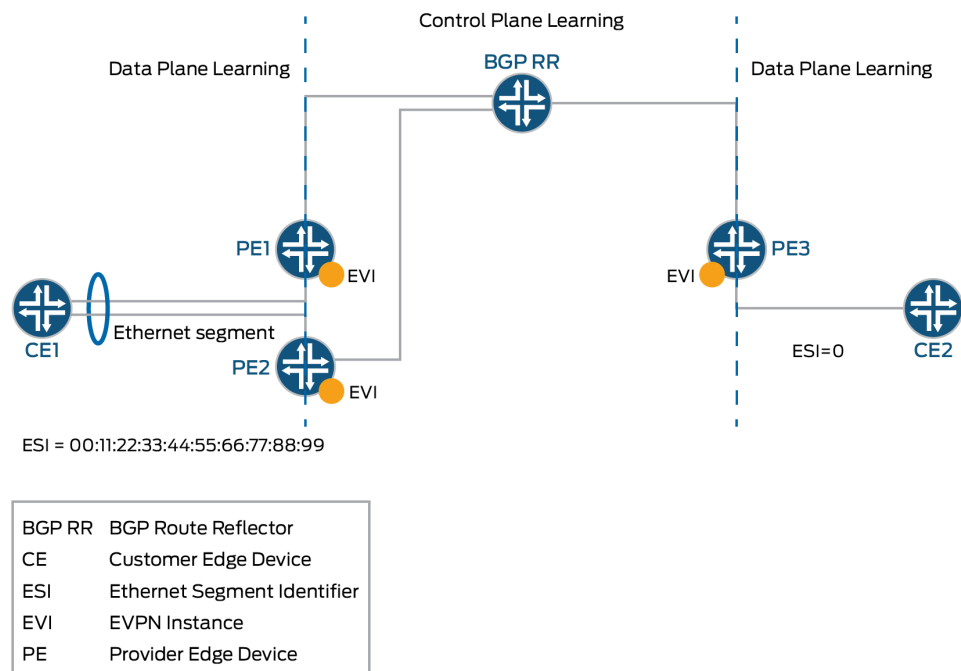
- You can enable migration of virtual machines between servers that exist in separate Layer 2 domains by tunneling the traffic over Layer 3 networks. This functionality allows you to dynamically allocate resources within or between data centers without being constrained by Layer 2 boundaries or being forced to create large or geographically stretched Layer 2 domains.

Understanding EVPN

EVPN is a new standards-based technology that provides virtual multi-point bridged connectivity between different Layer 2 domains over an IP or IP/MPLS backbone network. Similar to other VPN technologies, such as IPVPN and VPLS, EVPN instances (EVIs) are configured on PE routers to maintain logical service separation between customers. The PEs connect to CE devices which can be a router, switch, or host. The PE routers then exchange reachability information using Multi-Protocol BGP (MP-BGP) and encapsulated traffic is forwarded between PEs. Because elements of the architecture are

common with other VPN technologies, EVPN can be seamlessly introduced and integrated into existing service environments.

Figure 120: EVPN Overview



The EVPN technology provides mechanisms for next generation Data Center Interconnect (DCI) by adding extended control plane procedures to exchange the Layer 2 (MAC address) and Layer 3 (IP address) information among the participating Data Center Border Routers (DCBRs). These features help to address some of the DCI challenges, such as seamless VM mobility and optimal IP routing. Seamless VM mobility refers to the challenge of Layer 2 extension and maintaining connectivity in the face of VM mobility, and optimal IP routing refers to the challenge of supporting default gateway behavior for a VM's outbound traffic and triangular routing avoidance of a VM's inbound traffic.

The EVPN technology is used by the data center operator to offer multi-tenancy, flexible and resilient services that can be extended on demand. This flexibility and resiliency can require using compute resources among different physical data centers for a single service (Layer 2 extension), and VM motion.

EVPN supports all-active multihoming which allows a CE device to connect to two or more PE routers such that traffic is forwarded using all of the links between the devices. This enables the CE to load balance traffic to the multiple PE routers. More importantly, it allows a remote PE to load balance traffic to the multihomed PEs across the core network. This load balancing of traffic flows between data centers is known as aliasing. EVPN also has mechanisms that prevent the looping of broadcast, unknown unicast, and multicast (BUM) traffic in an all-active multi-homed topology.

Multihoming provides redundancy in the event that an access link or one of the PE routers fails. In either case, traffic flows from the CE towards the PE use the remaining active links. For traffic in the other direction, the remote PE updates its forwarding table to send traffic to the remaining active PEs connected to the multihomed Ethernet segment. EVPN provides a fast convergence mechanism so that the time it takes to make this adjustment is independent of the number of MAC addresses learned by the PE.

EVPN's MP-BGP control plane allows live virtual machines to be dynamically moved from one data center to another, also known as VM motion. After a VM is moved to a destination server/hypervisor it transmits a Gratuitous ARP that updates the Layer 2 forwarding table of the PE at the destination data center. The PE then transmits a MAC route update to all remote PEs, which in turn update their forwarding tables. In this manner, an EVPN tracks the movement of the VM, also known as MAC Mobility. EVPN also has mechanisms to detect and stop MAC flapping.

The EVPN technology, similar to Layer 3 MPLS VPN, is a technology that introduces the concept of routing MAC addresses using MP-BGP over MPLS core. Some of the important benefits of using EVPNs include:

- Ability to have a dual active multihomed edge device.
- Provides load balancing across dual-active links.
- Provides MAC address mobility.
- Provides multi-tenancy.
- Provides aliasing.
- Enables fast convergence.

VXLAN-EVPN Integration Overview

VXLAN defines a tunneling scheme to overlay Layer 2 networks on top of Layer 3 networks. It allows for optimal forwarding of Ethernet frames with support for multipathing of unicast and multicast traffic with the use of UDP/IP encapsulation for tunneling, and is mainly used for intra-datacenter site connectivity.

On the other hand, a unique characteristic of EVPN is that MAC address learning between PE devices occurs in the control plane. A new MAC address detected from a CE device is advertised by the local PE, using MP-BGP, to all the remote PE devices. This method differs from existing Layer 2 VPN solutions such as VPLS, which learn by flooding unknown unicast in the data plane. This control plane-based MAC learning method is the key enabler of the many useful features provided by EVPN.

Because MAC learning is handled in the control plane, this leaves EVPN with the flexibility to support different data plane encapsulation technologies between PEs. This is important because not every backbone network may be running MPLS, especially in Enterprise networks.

There is a lot of interest in EVPN today because it addresses many of the challenges faced by network operators that are building data centers to offer cloud and virtualization services. The main application of EVPN is Data Center Interconnect (DCI), the ability to extend Layer 2 connectivity between different data centers that are deployed to improve the performance of delivering application traffic to end users and for disaster recovery.

Although there are various DCI technologies available, EVPN has an added advantage over the other MPLS technologies because of its unique features, such as active-active redundancy, aliasing, and mass MAC withdrawal. As a result, to provide a solution for DCI, VXLAN is integrated with EVPN.

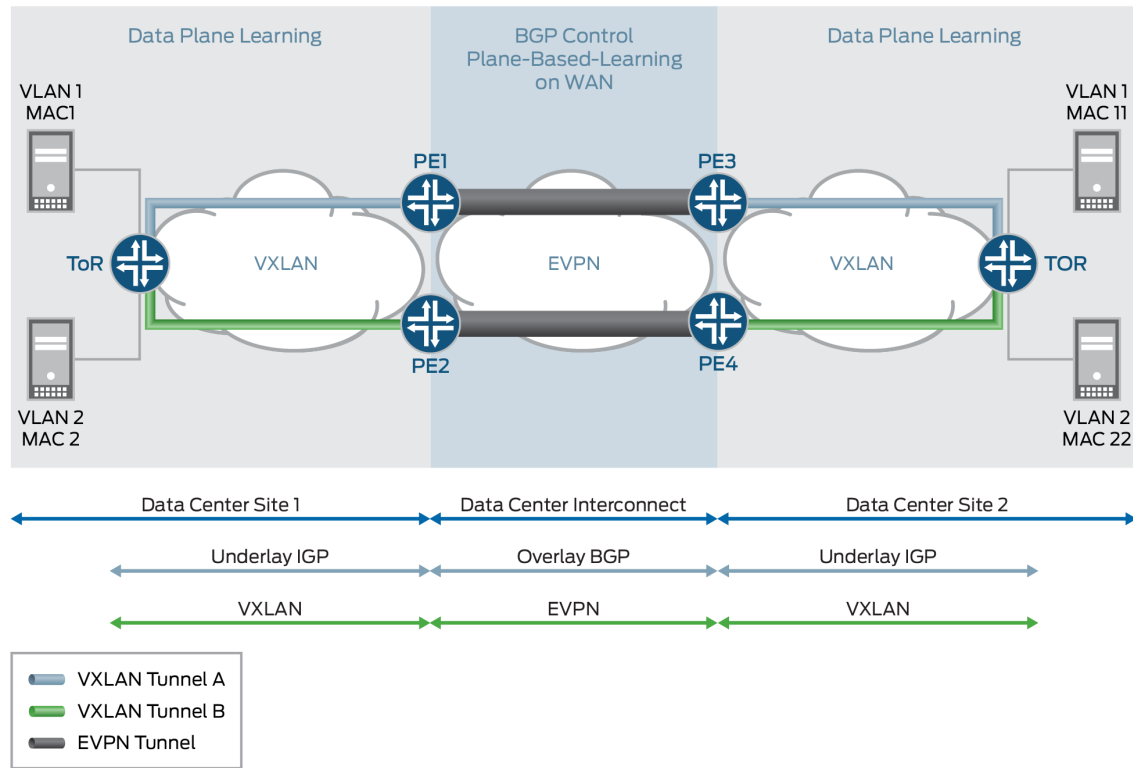
Every VXLAN network, which is connected to the MPLS or IP core, runs an independent instance of the IGP control plane. Each PE device participates in the IGP control plane instance of its VXLAN network. Here, each customer is a datacenter so it has its own virtual router for VXLAN underlay.

Each PE node may terminate the VXLAN data plane encapsulation where each VNI or VSID is mapped to a bridge domain. The PE router performs data plane learning on the traffic received from the VXLAN network.

Each PE node implements EVPN to distribute the client MAC addresses learnt over the VXLAN tunnel into BGP. Each PE node encapsulates the VXLAN or Ethernet frames with MPLS when sending the

packets over the MPLS core and with the VXLAN tunnel header when sending the packets over the VXLAN network

Figure 121: VXLAN-EVPN Integration Overview

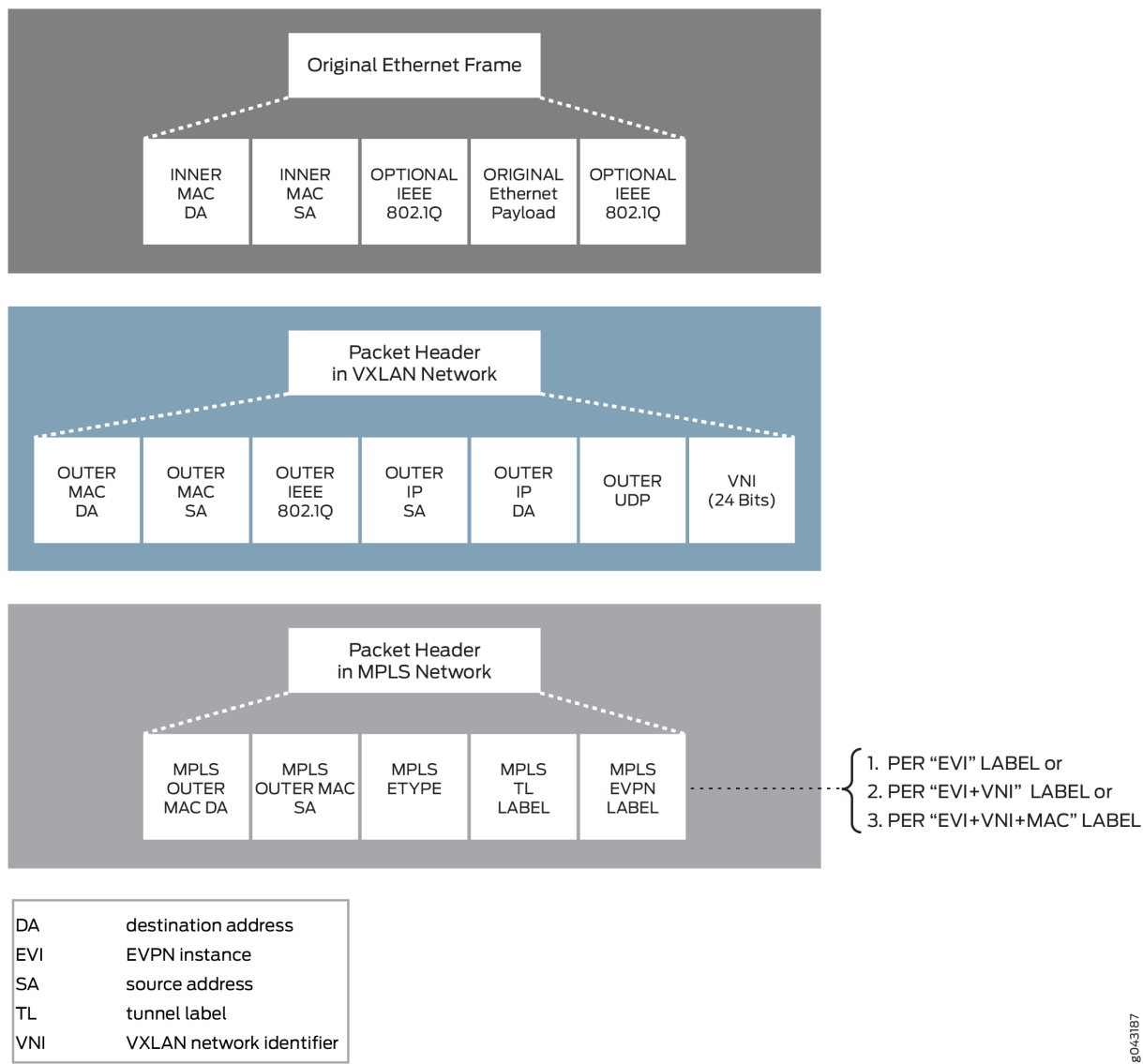


g043186

VXLAN-EVPN Packet Format

The VXLAN and EVPN packet format is as follows:

Figure 122: VXLAN-EVPN Packet Format

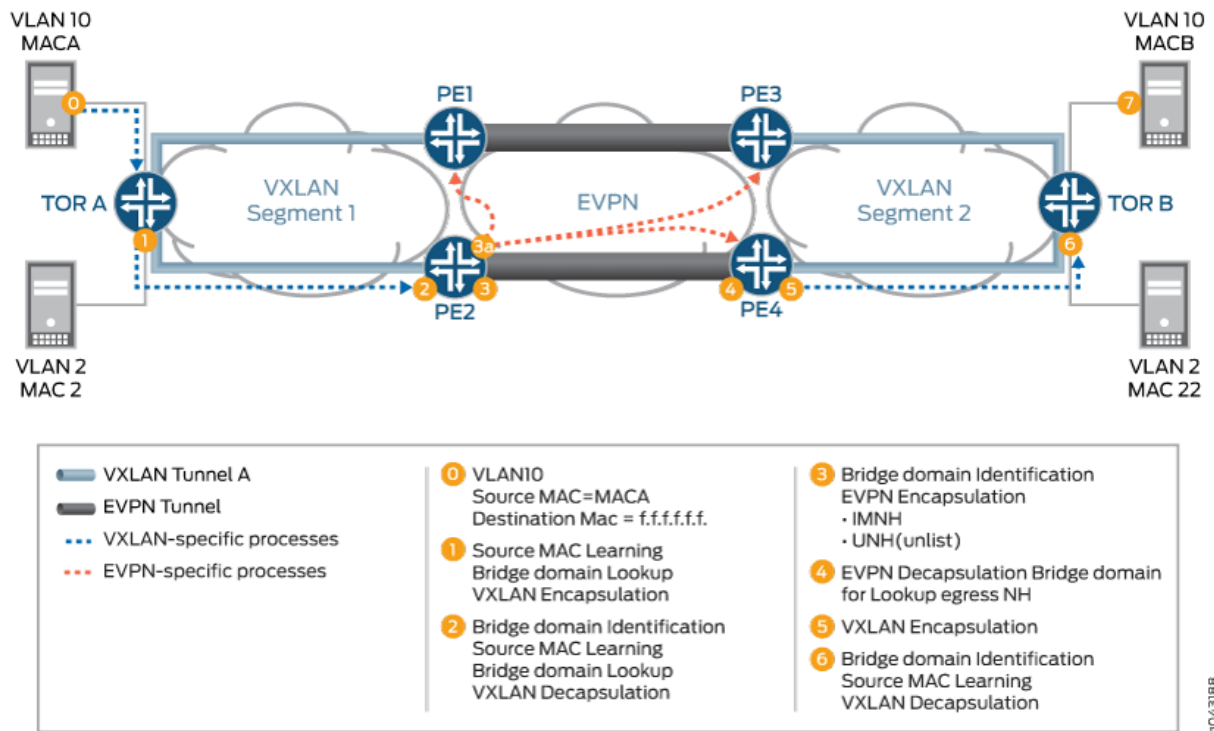


VXLAN-EVPN Packet Walkthrough

The following sections describe the packet walkthrough for two types of traffic between the VXLAN and EVPN networks:

BUM Traffic Handling

Figure 123: VXLAN-EVPN BUM Traffic Handling



The VXLAN to EVPN BUM traffic from VXLAN segment1 to VXLAN segment2 over the EVPN cloud is handled as follows:

1. 0—On boot up, Server A wants to send traffic to Server B. Because Server A does not have an ARP binding for Server B in its ARP table, Server A builds an ARP broadcast request and sends it.

The contents of the ARP packets are as follows:

- VLAN ID = VLAN 10
- Source MAC= MAC1
- Destination MAC = ff.ff.ff.ff.ff
- Source IP address = IP address of Server A or VM IP address
- Destination IP address = IP address of Server B
- Ether type of packet = 0x0806

A Layer 2 frame is sent to top-of-rack (TOR) switch TOR A, which is VXLAN enabled.

2. **1**—The ARP request (broadcast) frame is received by switch TOR A. TOR A is the originator and terminator of the VXLAN VTEP for VNI 1000. The VTEP for VXLAN 1000 is part of the broadcast domain for Server A VLAN 10.

After receiving the frame, TOR A performs ingress processing, including ingress packet classification. Based on the incoming VLAN in the packet, TOR A classifies the packet into one of the IFL under a given port. The family of this IFL is a bridge family. Based on the IFL bridge family, the bridge domain ID is identified.

After the bridge domain is identified, TOR A learns the incoming frame source MAC so that MAC A becomes reachable through this IFL. Because the frame is a broadcast frame, TOR A needs to send the frame to all the members of the broadcast domain (other than the member on which the frame was received). One of the members of the broadcast domain is the VTEP for VNI 1000. To send the frame on the VXLAN segment, TOR A completes VXLAN BUM next hop processing on the frame. The next hop pushes the VXLAN header.

The contents of the VXLAN header is as follows:

- Source MAC Address = MAC Address or source IP address interface
- Destination MAC address = Multicast MAC address
- Source IP address = 10.10.10.1
- Destination IP Address = Multicast group address (226.0.39.16)
- Source UDP port = Calculated based on the hash on the incoming frame header
- Destination UDP port = 4789 (well known port for VXLAN tunnel)

After building the VXLAN encapsulated frame, TOR A sends the frame to Router PE2.

3. **2**—Router PE2 receives the VXLAN frame and identifies the frame as a VXLAN frame by looking at the well-known destination UDP port. This VXLAN frame's VNI ID is used for bridge domain identification. After router PE2 identifies the bridge domain, PE2 completes MAC learning for the inner source MAC to the outer source IP address (MACA to 10.10.10.1 mapping). After the mapping is done, the VXLAN header is removed by VDNH.
4. **3A**—After MAC learning is done, the learnt source MAC (MAC1 to outer source IP) is sent to the L2ALD. This MAC route is sent by L2ALD to RPD for control plane learning of this MAC through BGP MAC route advertisement to BGP peers. After the BGP peer routers receive the MAC route advertisement, the routers install this MAC reachability (MACA, MPLS LABEL L1) in the bridge-domain table.
5. **3**—The given bridge domain points to the multicast next hop route for forwarding the packet over the EVPN cloud. This next hop pushes the service label (multicast MPLS label associated with VNI per

peer ID, bridge domain, label is the per peer ID and VNI ID). The MPLS packet is formed and sent over the MPLS cloud.

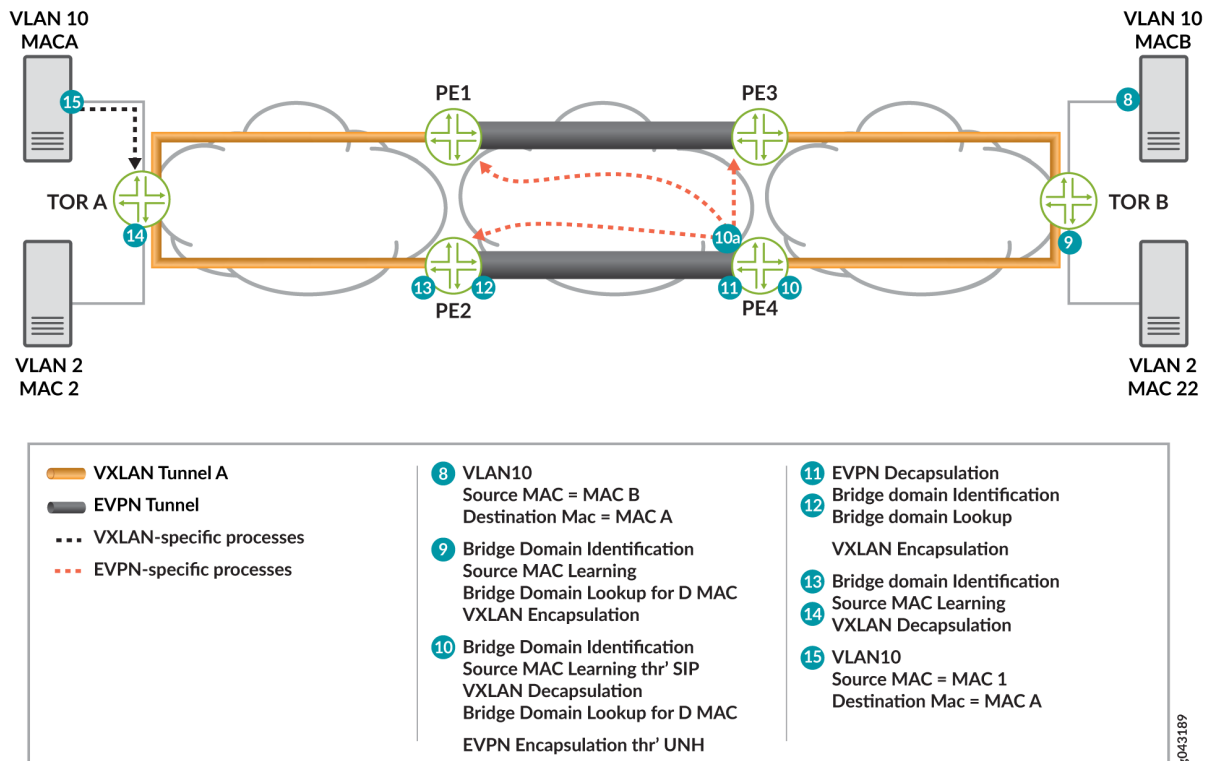
6. 4—Router PE4 receives the frame as an MPLS packet. Here, PE4 identifies the bridge domain by looking up the MPLS label L1 in the mpls.0 table. MPLS lookup points to the table next hop for the bridge domain next hop. After the bridge domain is identified and the packet is identified as a broadcast packet, the BUM composite flood next hop is executed. The BUM composite next hop also points to the VXLAN next hop (which is used for building the VXLAN multicast packet).
7. 5—The VXLAN next hop contains information for building the VXLAN header.

The VXLAN header information is as follows:

- Source MAC Address = MAC Address or the source IP address interface
 - Destination MAC address = Multicast MAC Address
 - Source IP address = 11.10.10.1
 - Destination IP Address = Multicast group address (226.0.39.16)
 - Source UDP port = Calculated based on the hash on the incoming frame header
 - Destination UDP port = 4789 (well known port for the VXLAN tunnel)
8. 6—Frame handling for this step is same as Step 1. After the VXLAN header is removed, the frame is forwarded to the CE flood route associated with the broadcast domain, and the packet is forwarded as a Layer 2 frame.
 9. 7—Server B receives an ARP request packet and sends an ARP reply to Server A.

Unicast Traffic Handling

Figure 124: VXLAN-EVPN Unicast Traffic Handling



Assuming that both data and control plane MAC learning has already happened, the VXLAN to EVPN unicast traffic (ARP reply) from Server B is handled as follows:

1. 8—Server B generates an ARP reply.

The contents of the ARP packets are as follows:

- VLAN ID = VLAN 10
- Source MAC = MACB (Server B interface MAC)
- Destination MAC = MACA
- Source IP address = IP address of Server B or VM IP address
- Destination IP address = IP address of Server A

The ARP packet is forwarded to switch TOR B.

2. **9**—After receiving the frame, switch TOR B classifies the incoming frame. The frame is classified in an IFL on the received interface. Based on the IFL family, the bridge domain associated with IFL is identified. On the given bridge domain, TOR B learns the source MAC address. Once TOR B completes the bridge domain destination MAC (MACA) look up, this look up provides the VXLAN unicast next hop. This next hop contains all the information needed to form the VXLAN header.

The content of the next hop that is required to form the packet is as follows:

- Source MAC Address = MAC Address of source IP address interface
- Destination MAC address = MAC address of the next hop
- Source IP address = 11.10.10.2
- Destination IP Address = 11.10.10.1 (as a result of the MAC learning process)
- Source UDP port = Calculated based on the hash on the incoming frame header
- Destination UDP port = 4789 (well known port for the VXLAN tunnel)

NOTE: An earlier version of the VXLAN draft used 8472 as the UDP port.

3. **10**— Router PE receives the VXLAN encapsulated frame. PE4 identifies the frame by completing the lookup using the destination IP address and the destination UDP port. This lookup results in the VXLAN decapsulation. The decapsulation next hop also stores the outer source IP address.

The next lookup is done based on the VNI ID 1000. This lookup results into the bridge domain table.

4. **10A**—Router PE completes the source MAC to source IP address learning and L2ALD receives the MAC learning notification. This MAC is sent to RPD for distribution to other PE routers through BGP-EVPN MAC advertisement route. The BGP control plane distributes this MAC reachability information to all other PE routers.

The destination MAC (MAC1) lookup is done in the bridge domain MAC address table. This lookup results into a unicast next hop (EVPN NH).

5. **11**—The EVPN unicast next hop is executed. This next hop contains an unicast MPLS service label. This label is distributed through the MP-BGP control plane. The downstream peer allocates this MPLS service label. Allocation of this label can be per PE (PE, VLAN) or per MAC address basis. Based on the information in the next hop, the MPLS packet is formed and forwarded on the MPLS network.
6. **12**—Router PE2 receives the frame. The frame is identified as an MPLS packet. An MPLS label lookup is done in the MPLS.0 table. This lookup results in the table next hop and in the bridge domain table.

The destination MAC (MAC1) lookup is done in the bridge domain MAC table. This lookup results in a VXLAN unicast next hop.

7. 13—The VXLAN unicast next hop contains all the information for building the VXLAN encapsulated header. The VXLAN header is imposed on the packet.

The contents of the VXLAN encapsulation next hop header are as follows:

- Source MAC Address = MAC Address of source IP address interface
- Destination MAC address = MAC address of the next hop
- Source IP address = 10.10.10.2
- Destination IP Address = 10.10.10.1 (as a result of the MAC learning process)
- Source UDP port = Calculated based on the hash on the incoming frame header
- Destination UDP port = 4789 (well known port for the VXLAN tunnel)

8. 14—The VXLAN encapsulated frame is received by switch TOR A. TOR A identifies the frame by doing the lookup using the destination IP address and the destination UDP port. This lookup results in the VXLAN decapsulation. The decapsulated next hop also stores the outer source IP address.

The next lookup is done based on the VNI ID 1000. This lookup results into the bridge domain table. TOR A completes the source MAC (MAC2) to source IP address (10.10.10.2) learning. TOR A looks up the destination MAC (MAC1) in the bridge domain MAC address table. This lookup results into a unicast next hop that has the information about the egress interface.

9. 15—Server A receives the ARP reply, and Server A and Server B are ready to communicate.

Implementation Overview of VXLAN-EVPN Integration for DCI

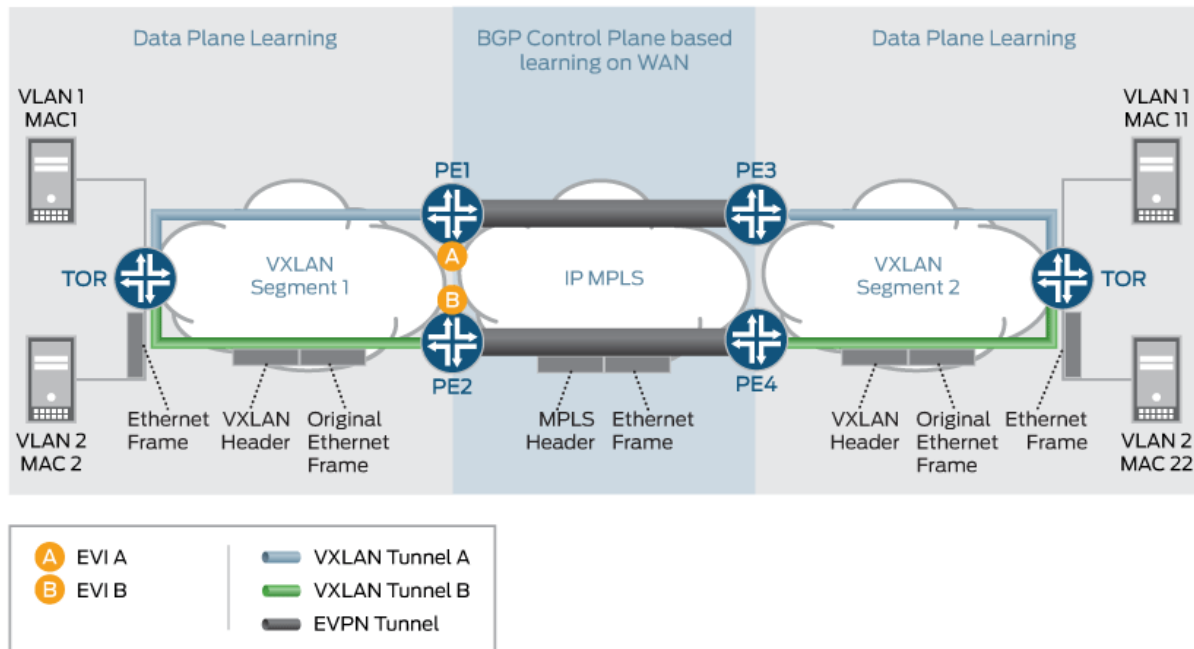
The following sections provide use case scenarios for VXLAN-EVPN integration for DCI.

VNI Base Service Use Case

In the case of the VNI base service, there is one-to-one mapping between a VNI and an EVI. In this case, there is no need to carry the VNI in the MAC advertisement route because the bridge domain ID can be derived from the route-target (RT) associated with this route. The MPLS label allocation is done per EVI basis.

Figure 125 on page 1137 provides an overview for VNI base use case scenarios. The VNI base service is used most commonly for achieving the VNI translation and VNI to VLAN interworking.

Figure 125: VNI Base Service

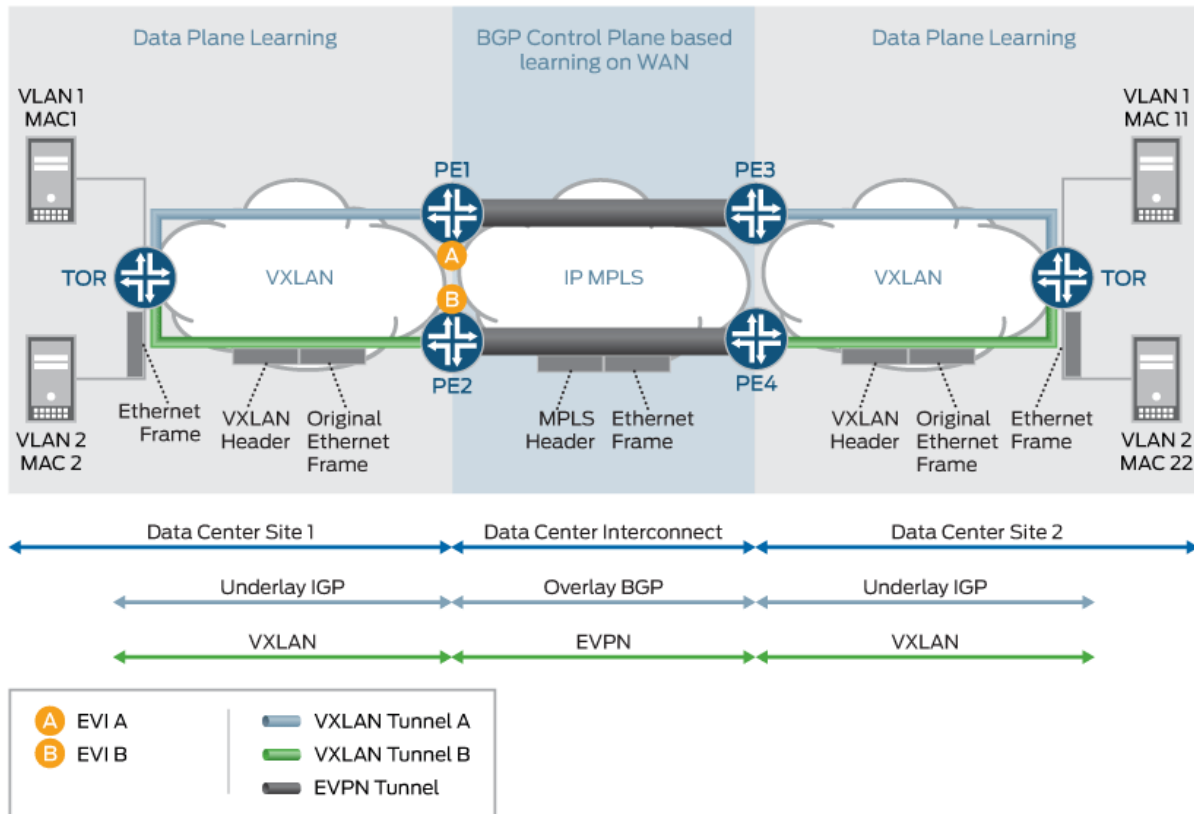


VNI Aware Service Use Case

In the case of VNI aware bundle mode, there are multiple VNIs that can be mapped to the same EVI. The Ethernet tag ID must be set to the VNI ID in the BGP routes advertisements. The MPLS label allocation in this use case should be done per EVI, VNI basis so that the VXLAN can be terminated at the ingress PE router and recreated at the egress PE router.

Figure 126 on page 1138 provides details about the VNI aware service used case.

Figure 126: VNI Aware Service

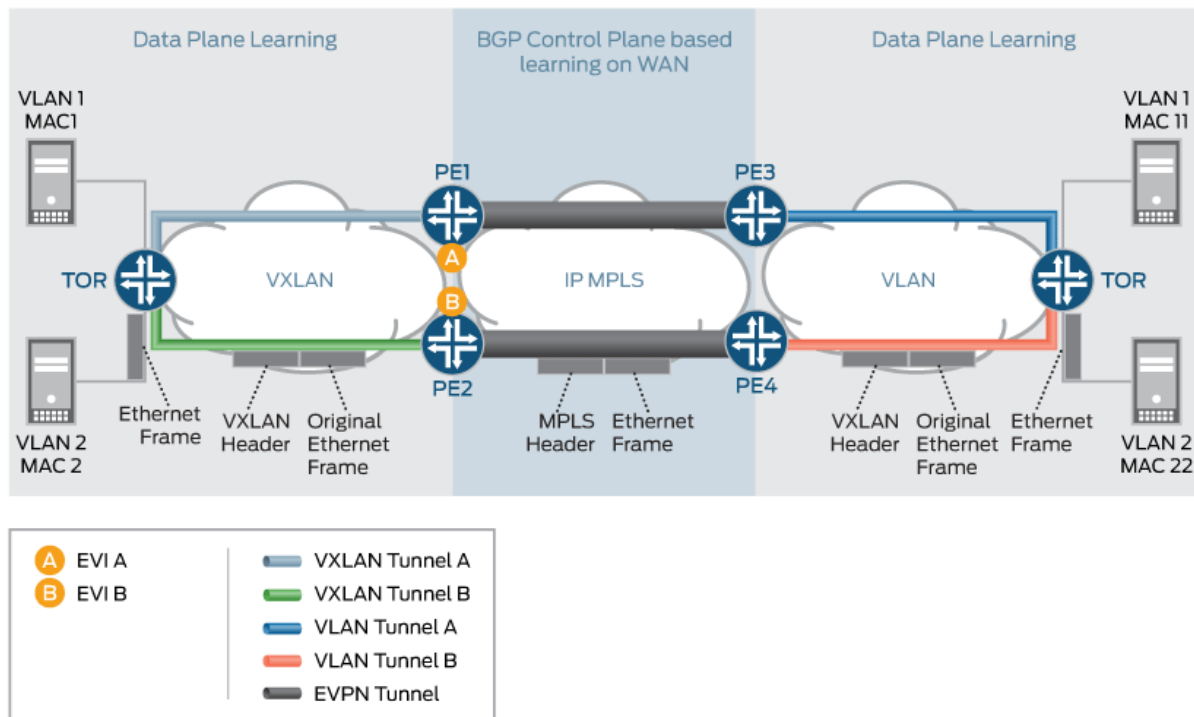


VXLAN-VLAN Interworking Use Case

This use case scenario is required for heterogeneous datacenter sites. In this scenario, the new data center site is a VXLAN-based datacenter site, and the old datacenter sites are based on VLAN. In this scenario, there is a need to have VXLAN interworking with VLAN over EVPN.

Figure 127 on page 1139 provides the detail packet walkthrough for the VXLAN-VLAN interworking use case scenario. There is a need to do the VLAN to VXLAN interworking and vice-versa from the control plane BGP route updates perspective. The label allocation needs to be done on a per EVI-basis.

Figure 127: VXLAN-VLAN Interworking



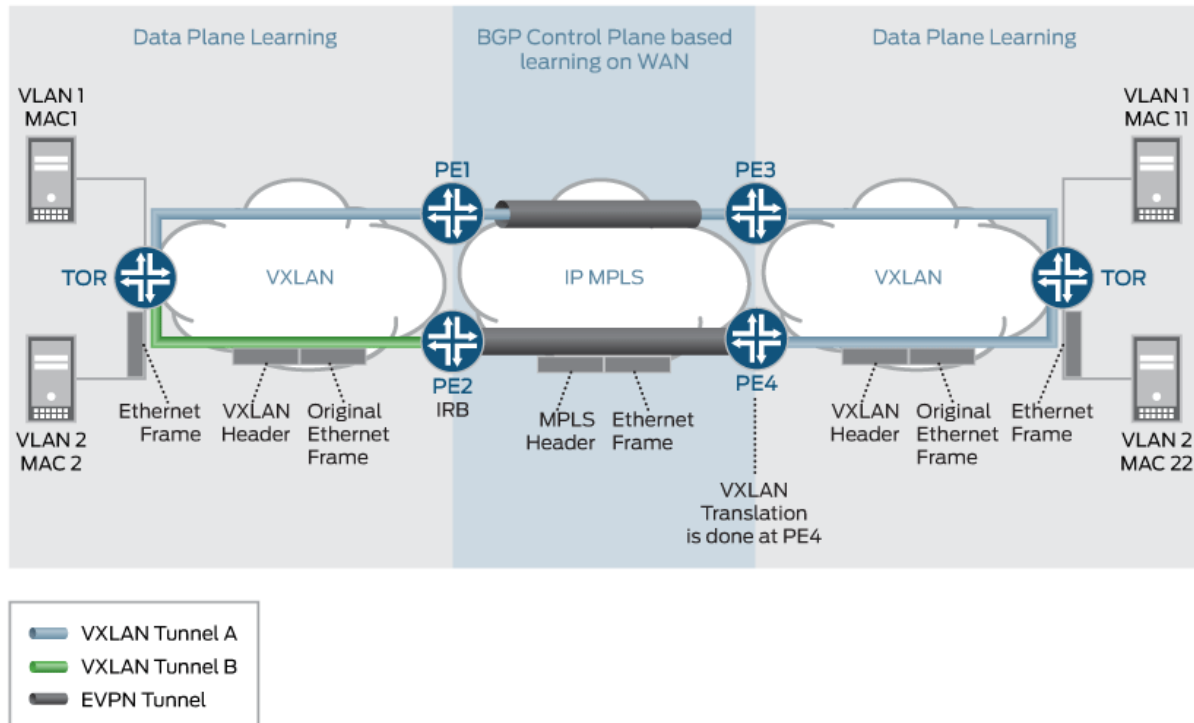
8043194

Inter VXLAN Routing Use Case

In this use case, a VM or host in one subnet (VNI-A) wants to send traffic to a VM or host in a different subnet (VNI-B). In order to provide this communication, the inter VXLAN routing should be supported.

Figure 128 on page 1140 provides the use case scenarios for the inter VXLAN routing use case.

Figure 128: Inter-VXLAN Routing



Redundancy Use Case

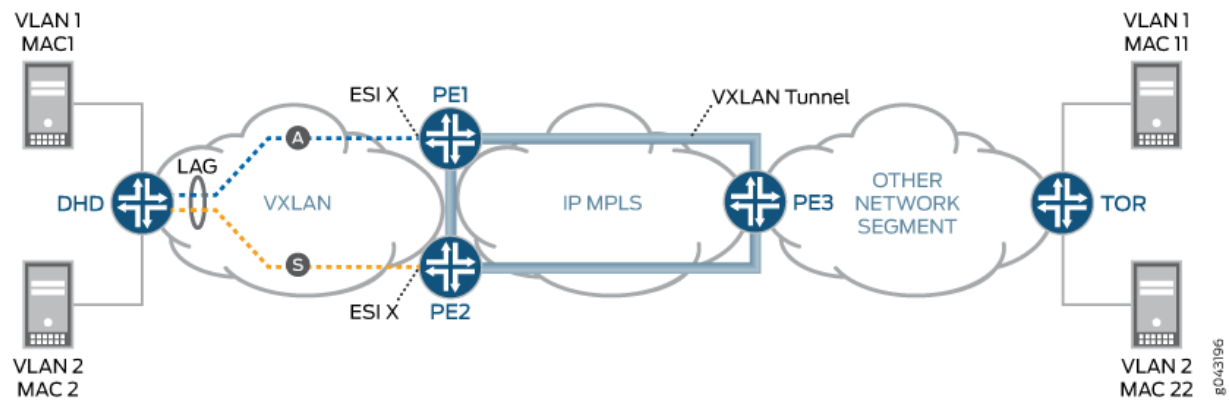
The two types of redundancy use case scenarios include:

Active-Standby Redundancy Use Case

In this use case scenario, the TOR switch (VXLAN originating GW), or the VXLAN network originating the VXLAN tunnel, is dual homed to two PE devices for active-standby redundancy. If the active link or node fails, a backup path takes over.

Figure 129 on page 1141 provides details of the active-standby redundancy use case scenario.

Figure 129: Active-Standby Redundancy



Supported and Unsupported Features for VXLAN DCI Using EVPN

Junos OS supports the following features for VXLAN DCI using EVPN:

- One-to-one mapping of VXLAN tunnel and an EVPN instance. In other words, one-to-one mapping between a VNI and an EVI.
- Many-to-one mapping of VXLAN tunnels over one EVPN instance, where multiple VNIs can be mapped to the same EVI.
- VNI Translation.

NOTE: VNI translation is supported by normalizing a VXLAN tag into a VLAN.

- VXLAN-to-VLAN interworking.
- Inter-VXLAN routing.
- Single active redundancy.
- Active-active redundancy in the PIM BIDIR mode.
- VXLAN tunnel traffic protection using IPSec.
- Graceful Routing Engine switchover.

- ISSU.

Junos OS does not support the following functionality for VXLAN DCI using EVPN:

- VXLAN uses IANA assigned UDP port 4789. Packets destined to the UDP port 4789 are processed only when the VXLAN configuration is enabled. The VXLAN packets are decapsulated by the forwarding plane and an inner Layer 2 packet is processed. MAC learnt packets are generated for the control plane processing for newly learnt MAC entries. These entries are throttled using existing infrastructure for MAC learning. VXLAN generates addition learn messages for the remote endpoints. These messages are also throttled using the existing infrastructure for denial of service detection.
- Packets received on the VXLAN tunnel are processed only if the VXLAN identifier in the packet is a known entity to the device. Unknown entities are discarded by the forwarding plane.
- Using configurable firewall filters can be discarded before it reaches the VXLAN processing module in the forwarding plane of the MX series routers.
- Logical systems.

Release History Table

Release	Description
16.1	Starting in Junos OS Release 16.1, Ethernet VPN (EVPN) technology can be used to interconnect Virtual Extensible Local Area Network (VXLAN) networks over an MPLS/IP network to provide data center connectivity.

Example: Configuring VXLAN Data Center Interconnect Using EVPN

IN THIS SECTION

- [Requirements | 1143](#)
- [Overview | 1143](#)
- [Configuration | 1144](#)
- [Verificatiton | 1156](#)

This example shows how to configure Virtual Extensible Local Area Network (VXLAN) data center connectivity using Ethernet VPN (EVPN) to leverage the benefits of EVPN as a data center interconnect (DCI) solution.

Requirements

This example uses the following hardware and software components:

- Two provider edge (PE) devices in different data centers (DCs) acting as VXLAN tunnel endpoints (VTEPs).
- Two customer edge (CE) devices.
- Four host devices connected to each PE and CE device.

Before you begin:

- Configure the device interfaces.
- Configure an IGP, such as OSPF, on all the devices.
- Establish a BGP session between the PE devices.
- Configure MPLS and RSVP on the PE devices.
- Configure PIM on the CE devices and in the routing instance of the PE devices.

Overview

IN THIS SECTION

- [Topology](#) | [1144](#)

VXLAN is a technology that provides intra data center connectivity using a tunneling scheme to stretch Layer 2 connections over an intervening Layer 3 network.

The Ethernet VPN (EVPN) technology, on the other hand, provides a solution for multipoint Layer 2 VPN services with advanced multihoming capabilities, using BGP control plane over MPLS/IP network.

Although several solutions are available for data center connectivity, the integration of EVPN with VXLAN in Junos OS Release 16.1 and later releases, provides an added advantage over the existing MPLS data center interconnect (DCI) technologies.

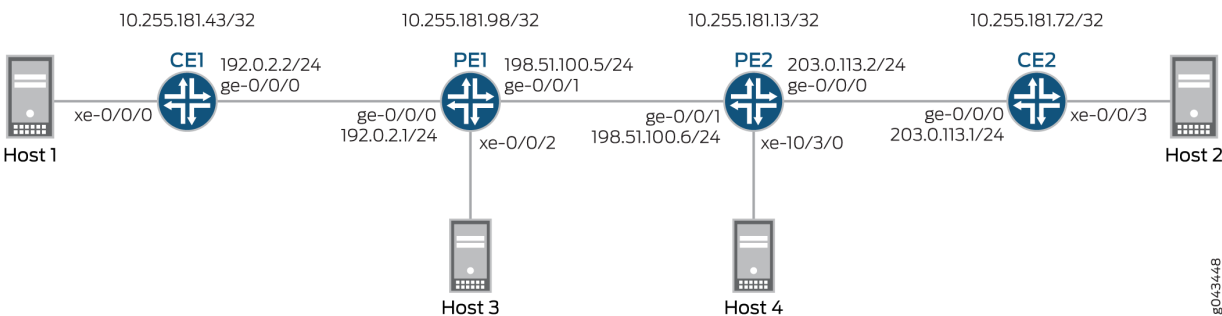
EVPN provides mechanisms for next generation DCI by adding extended control plane procedures to exchange Layer 2 MAC address and Layer 3 IP address information among the participating Data Center

Border Routers (DCBRs). EVPN with its advanced features like active-active redundancy, aliasing, and mass MAC withdrawal helps in addressing the DCI challenges, such as seamless VM mobility and optimal IP routing, thus making it essential to provide VXLAN solutions over EVPN.

Figure 130 on page 1144 illustrates VXLAN data center interconnect using EVPN between devices PE1 and PE2 that are located in different data centers (DC1 and DC2, respectively). Each PE device is connected to one CE device and one host. All the PE and CE devices are configured under VLAN 10, and with the same VXLAN Network Identifier (VNI) of 10. Devices CE1 and PE1 belong to the multicast group of 192.168.1.10, and devices CE2 and PE2 belong to the multicast group of 172.16.1.10.

Topology

Figure 130: VXLAN Data Center Interconnect Using EVPN



Configuration

IN THIS SECTION

- CLI Quick Configuration | 1144
- Procedure | 1148
- Results | 1152

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the [edit] hierarchy level, and then enter commit from configuration mode.

CE1

```

set interfaces xe-0/0/0 vlan-tagging
set interfaces xe-0/0/0 encapsulation flexible-ethernet-services
set interfaces xe-0/0/0 unit 10 encapsulation vlan-bridge
set interfaces xe-0/0/0 unit 10 vlan-id 10
set interfaces ge-0/0/0 unit 0 family inet address 192.0.2.2/24
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.181.43/32
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols pim rp local address 10.255.181.43
set protocols pim interface all
set bridge-domains evpn10 vlan-id 10
set bridge-domains evpn10 interface xe-0/0/0.10
set bridge-domains vxlan vni 10
set bridge-domains vxlan multicast-group 172.16.1.10
set bridge-domains vxlan encapsulate-inner-vlan
set bridge-domains vxlan decapsulate-accept-inner-vlan

```

CE2

```

set interfaces xe-0/0/3 vlan-tagging
set interfaces xe-0/0/3 encapsulation flexible-ethernet-services
set interfaces xe-0/0/3 unit 10 encapsulation vlan-bridge
set interfaces xe-0/0/3 unit 10 vlan-id 10
set interfaces lo0 unit 0 family inet address 10.255.181.72/32
set protocols ospf area 0 interface ge-0/0/0.0
set protocols ospf area 0 interface lo0.0 passive
set protocols ospf area 0 interface fxp0.0 disable
set protocols pim rp local address 10.255.181.72
set protocols pim interface all
set bridge-domains evpn10 vlan-id 10
set bridge-domains evpn10 interface xe-0/0/3.10
set bridge-domains vxlan vni 10
set bridge-domains vxlan multicast-group 192.168.1.10
set bridge-domains vxlan encapsulate-inner-vlan
set bridge-domains vxlan decapsulate-accept-inner-vlan

```

PE1

```

set interfaces xe-0/0/2 vlan-tagging
set interfaces xe-0/0/2 encapsulation flexible-ethernet-services
set interfaces xe-0/0/2 unit 10 encapsulation vlan-bridge
set interfaces xe-0/0/2 unit 10 vlan-id 10
set interfaces ge-0/0/0 unit 0 family inet address 192.0.2.1/24
set interfaces ge-0/0/1 mtu 1600
set interfaces ge-0/0/1 unit 0 family inet address 198.51.100.5/24
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 1 family inet address 10.255.181.98/32
set protocols rsvp interface all
set protocols mpls no-cspf
set protocols mpls label-switched-path to-PE2 to 10.255.181.13
set protocols mpls interface all
set protocols bgp family evpn signaling
set protocols bgp group ibgp type internal
set protocols bgp group ibgp neighbor 10.255.181.13 local-address 10.255.181.98
set protocols ospf area 0 interface ge-0/0/1.0
set protocols ospf area 0 interface fxp0.0 disable
set protocols ospf area 0 interface lo0.0 passive
set routing-instances evpn10 vtep-source-interface lo0.0
set routing-instances evpn10 instance-type evpn
set routing-instances evpn10 vlan-id 10
set routing-instances evpn10 interface xe-0/0/2.10
set routing-instances evpn10 vxlan vni 10
set routing-instances evpn10 vxlan multicast-group 172.16.1.10
set routing-instances evpn10 vxlan encapsulate-inner-vlan
set routing-instances evpn10 vxlan decapsulate-accept-inner-vlan
set routing-instances evpn10 route-distinguisher 10.255.181.98:10
set routing-instances evpn10 vrf-target target:10:10
set routing-instances evpn10 protocols evpn
set routing-instances evpna instance-type vrf
set routing-instances evpna route-distinguisher 10.255.181.98:11
set routing-instances evpna vrf-target target:65000:11
set routing-instances evpna vrf-table-label
set routing-instances vrf instance-type vrf
set routing-instances vrf interface ge-0/0/0.0
set routing-instances vrf interface lo0.0
set routing-instances vrf route-distinguisher 10.255.181.98:100
set routing-instances vrf vrf-target target:100:100
set routing-instances vrf protocols ospf area 0 interface lo0.0 passive

```

```

set routing-instances vrf protocols ospf area 0 interface ge-0/0/0.0
set routing-instances vrf protocols pim rp static address 10.255.181.43
set routing-instances vrf protocols pim interface all

```

PE2

```

set interfaces ge-0/0/1 mtu 1600
set interfaces ge-0/0/1 unit 0 family inet address 198.51.100.6/24
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces xe-10/3/0 vlan-tagging
set interfaces xe-10/3/0 encapsulation flexible-ethernet-services
set interfaces xe-10/3/0 unit 10 encapsulation vlan-bridge
set interfaces xe-10/3/0 unit 10 vlan-id 10
set interfaces lo0 unit 1 family inet address 10.255.181.13/32
set protocols rsvp interface all
set protocols mpls no-cspf
set protocols mpls label-switched-path to-PE1 to 10.255.181.98
set protocols mpls interface all
set protocols bgp family evpn signaling
set protocols bgp group ibgp type internal
set protocols bgp group ibgp neighbor 10.255.181.98 local-address 10.255.181.13
set protocols ospf area 0 interface ge-0/0/1.0
set protocols ospf area 0 interface fxp0.0 disable
set protocols ospf area 0 interface lo0.0 passive
set routing-instances evpn10 vtep-source-interface lo0.0
set routing-instances evpn10 instance-type evpn
set routing-instances evpn10 vlan-id 10
set routing-instances evpn10 interface xe-10/3/0.10
set routing-instances evpn10 vxlan vni 10
set routing-instances evpn10 vxlan multicast-group 192.168.1.10
set routing-instances evpn10 vxlan encapsulate-inner-vlan
set routing-instances evpn10 vxlan decapsulate-accept-inner-vlan
set routing-instances evpn10 route-distinguisher 10.255.181.13:10
set routing-instances evpn10 vrf-target target:10:10
set routing-instances evpn10 protocols evpn
set routing-instances evpna instance-type vrf
set routing-instances evpna route-distinguisher 10.255.181.13:11
set routing-instances evpna vrf-target target:65000:11
set routing-instances evpna vrf-table-label
set routing-instances vrf instance-type vrf
set routing-instances vrf interface xe-10/3/0.0
set routing-instances vrf interface lo0.0

```

```

set routing-instances vrf route-distinguisher 10.255.181.13:100
set routing-instances vrf vrf-target target:100:100
set routing-instances vrf protocols ospf area 0 interface lo0.0 passive
set routing-instances vrf protocols ospf area 0 interface xe-10/3/0.0
set routing-instances vrf protocols pim rp static address 10.255.181.72
set routing-instances vrf protocols pim interface all

```

Procedure

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure Device CE1:

NOTE: Repeat this procedure for Device CE2 after modifying the appropriate interface names, addresses, and other parameters.

1. Configure Device CE1 interfaces.

```

[edit interfaces]
user@CE1# set xe-0/0/0 vlan-tagging
user@CE1# set xe-0/0/0 encapsulation flexible-ethernet-services
user@CE1# set xe-0/0/0 unit 10 encapsulation vlan-bridge
user@CE1# set xe-0/0/0 unit 10 vlan-id 10
user@CE1# set ge-0/0/0 unit 0 family inet address 192.0.2.2/24
user@CE1# set ge-0/0/0 unit 0 family mpls
user@CE1# set lo0 unit 0 family inet address 10.255.181.43/32

```

2. Enable OSPF on Device CE1 interface, excluding the management interface.

```

[edit protocols]
user@CE1# set ospf area 0.0.0.0 interface ge-0/0/0.0
user@CE1# set ospf area 0.0.0.0 interface lo0.0 passive
user@CE1# set ospf area 0.0.0.0 interface fxp0.0 disable

```

3. Enable PIM on all the interfaces of Device CE1.

```
[edit protocols]
user@CE1# set pim rp local address 10.255.181.43
user@CE1# set pim interface all
```

4. Configure an EVPN bridge domain, and assign VLAN ID and interface.

```
[edit bridge-domains]
user@CE1# set evpn10 vlan-id 10
user@CE1# set evpn10 interface xe-0/0/0.10
```

5. Configure a VXLAN bridge domain, assign VXLAN ID, a multicast group address, and encapsulation and decapsulation parameters.

```
[edit bridge-domains]
user@CE1# set vxlan vni 10
user@CE1# set vxlan multicast-group 172.16.1.10
user@CE1# set vxlan encapsulate-inner-vlan
user@CE1# set vxlan decapsulate-accept-inner-vlan
```

Step-by-Step Procedure

To configure Device PE1:

NOTE: Repeat this procedure for Device PE2 after modifying the appropriate interface names, addresses, and other parameters.

1. Configure Device PE1 interfaces.

```
[edit interfaces]
user@PE1# set xe-0/0/2 vlan-tagging
user@PE1# set xe-0/0/2 encapsulation flexible-ethernet-services
user@PE1# set xe-0/0/2 unit 10 encapsulation vlan-bridge
user@PE1# set xe-0/0/2 unit 10 vlan-id 10
user@PE1# set ge-0/0/0 unit 0 family inet address 192.0.2.1/24
user@PE1# set ge-0/0/1 mtu 1600
```



```

user@PE1# set ge-0/0/1 unit 0 family inet address 198.51.100.5/24
user@PE1# set ge-0/0/1 unit 0 family mpls
user@PE1# set lo0 unit 1 family inet address 10.255.181.98/32

```

2. Enable MPLS and RSVP on all the interfaces of Device PE1.

```

[edit protocols]
user@PE1# set rsvp interface all
user@PE1# set mpls no-cspf
user@PE1# set mpls interface all

```

3. Configure a label-switched-path from Device PE1 to Device PE2.

```

[edit protocols]
user@PE1# set mpls label-switched-path to-PE2 to 10.255.181.13

```

4. Configure internal BGP peering between Devices PE1 and PE2, and enable EVPN signaling for the BGP session.

```

[edit protocols]
user@PE1# set bgp family evpn signaling
user@PE1# set bgp group ibgp type internal
user@PE1# set bgp group ibgp neighbor 10.255.181.13 local-address 10.255.181.98

```

5. Configure OSPF on Device PE1 interface, excluding the management interface.

```

[edit protocols]
user@PE1# set ospf area 0 interface ge-0/0/1.0
user@PE1# set ospf area 0 interface fxp0.0 disable
user@PE1# set ospf area 0 interface lo0.0 passive

```

6. Configure an EVPN routing instance, assign the VXLAN tunnel endpoint source interface, VLAN ID, assign route distinguisher and VRF target values, and assign Device PE1 interface to the routing instance.

```

[edit routing-instances]
user@PE1# set evpn10 vtep-source-interface lo0.0
user@PE1# set evpn10 instance-type evpn

```

```

user@PE1# set evpn10 vlan-id 10
user@PE1# set evpn10 interface xe-0/0/2.10
user@PE1# set evpn10 route-distinguisher 10.255.181.13:10
user@PE1# set evpn10 vrf-target target:10:10
user@PE1# set evpn10 protocols evpn

```

7. Assign the VXLAN ID, multicast group address, and encapsulation and decapsulation parameters for the EVPN routing instance.

```

[edit routing-instances]
user@PE1# set evpn10 vxlan vni 10
user@PE1# set evpn10 vxlan multicast-group 172.16.1.10
user@PE1# set evpn10 vxlan encapsulate-inner-vlan
user@PE1# set evpn10 vxlan decapsulate-accept-inner-vlan

```

8. Configure the first VPN routing and forwarding (VRF) routing instance, and assign route distinguisher and vrf-target values.

```

[edit routing-instances]
user@PE1# set evpna instance-type vrf
user@PE1# set evpna route-distinguisher 10.255.181.13:11
user@PE1# set evpna vrf-target target:65000:11
user@PE1# set evpna vrf-table-label

```

9. Configure the second VRF routing instance, and assign Device PE1 interfaces, route distinguisher and vrf-target values.

```

[edit routing-instances]
user@PE1# set vrf instance-type vrf
user@PE1# set vrf interface ge-0/0/0.0
user@PE1# set vrf interface lo0.0
user@PE1# set vrf route-distinguisher 10.255.181.13:100
user@PE1# set vrf vrf-target target:100:100

```

10. Configure OSPF and PIM protocols for the second VRF routing instance.

```

[edit routing-instances]
user@PE1# set vrf protocols ospf area 0 interface lo0.0 passive
user@PE1# set vrf protocols ospf area 0 interface ge-0/0/0.0

```

```

user@PE1# set vrf protocols pim rp static address 10.255.181.43
user@PE1# set vrf protocols pim interface all

```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, and `show routing-instances` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

CE1

```

user@CE1# show interfaces
xe-0/0/0 {
    vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 10 {
        encapsulation vlan-bridge;
        vlan-id 10;
    }
}
ge-0/0/0 {
    unit 0 {
        family inet {
            address 192.0.2.2/24;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.255.181.43/32;
        }
    }
}
user@CE1# show protocols
ospf {
    area 0.0.0.0 {
        interface ge-0/0/0.0;
        interface lo0.0 {
            passive;
        }
    }
}

```

```

        interface fxp0.0 {
            disable;
        }
    }
}
pim {
    rp {
        local {
            address 10.255.181.43;
        }
    }
    interface all;
}
user@CE1# show bridge-domains
evpn10 {
    vlan-id 10;
    interface xe-0/0/0.10;
    vxlan {
        vni 10;
        multicast-group 172.16.1.10;
        encapsulate-inner-vlan;
        decapsulate-accept-inner-vlan;
    }
}

```

PE1

```

user@PE1# show interfaces
xe-0/0/2 {
    vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 10 {
        encapsulation vlan-bridge;
        vlan-id 10;
    }
}
ge-0/0/0 {
    unit 0 {
        family inet {
            address 192.0.2.1/24;
        }
    }
}

```

```

}
ge-0/0/1 {
    mtu 1600;
    unit 0 {
        family inet {
            address 198.51.100.5/24;
        }
        family mpls;
    }
}
lo0 {
    unit 1 {
        family inet {
            address 10.255.181.98/32;
        }
    }
}
user@PE1# show protocols
rsvp {
    interface all;
}
mpls {
    no-cspf;
    label-switched-path to-PE2 {
        to 10.255.181.13;
    }
    interface all;
}
bgp {
    family evpn {
        signaling;
    }
    group ibgp {
        type internal;
        neighbor 10.255.181.13 {
            local-address 10.255.181.98;
        }
    }
}
ospf {
    area 0.0.0.0 {
        interface ge-0/0/1.0;
        interface fxp0.0 {

```

```

        disable;
    }
    interface lo0.0 {
        passive;
    }
}
}
user@PE1# show routing-instances
evpn10 {
    vtep-source-interface lo0.0;
    instance-type evpn;
    vlan-id 10;
    interface xe-0/0/2.10;
    vxlan {
        vni 10;
        multicast-group 172.16.1.10;
        encapsulate-inner-vlan;
        decapsulate-accept-inner-vlan;
    }
    route-distinguisher 10.255.181.13:10;
    vrf-target target:10:10;
    protocols {
        evpn;
    }
}
evpna {
    instance-type vrf;
    route-distinguisher 10.255.181.98:11;
    vrf-target target:65000:11;
    vrf-table-label;
}
vrf {
    instance-type vrf;
    interface ge-0/0/0.0;
    interface lo0.0;
    route-distinguisher 10.255.181.98:100;
    vrf-target target:100:100;
    protocols {
        ospf {
            area 0.0.0.0 {
                interface lo0.0 {
                    passive;
                }
            }
        }
    }
}

```

```
        interface ge-0/0/0.0;
    }
}
pim {
    rp {
        static {
            address 10.255.181.43;
        }
    }
    interface all;
}
}
```

Verificatiton

IN THIS SECTION

- [Verifying MAC Learning | 1156](#)
- [Verifying PIM Reachability | 1157](#)
- [Verifying VXLAN Reachability | 1160](#)

Confirm that the configuration is working properly.

Verifying MAC Learning

Purpose

Verify the bridging and EVPN MAC table entries on CE and PE devices.

Action

On Device CE1, determine the bridging MAC table entries.

From operational mode, run the **show bridge mac-table** command.

```
user@CE1> show bridge mac-table
```

```
MAC flags      (S -static MAC, D -dynamic MAC, L -locally learned, C -Control MAC
                O -OVSDDB MAC, SE -Statistics enabled, NM -Non configured MAC, R -Remote PE MAC)
```

```
Routing instance : default-switch
```

```
Bridging domain : evpn10, VLAN : 10
```

MAC address	MAC flags	Logical interface	NH Index	RTR ID
00:00:00:00:00:11	D	xe-0/0/0.10		
00:00:00:00:00:22	D	vtep.32769		

On Device PE1, determine the EVPN MAC table entries.

From operational mode, run the **show evpn mac-table** command.

```
user@PE1> show evpn mac-table
```

```
MAC flags      (S -static MAC, D -dynamic MAC, L -locally learned, C -Control MAC
                O -OVSDDB MAC, SE -Statistics enabled, NM -Non configured MAC, R -Remote PE MAC)
```

```
Routing instance : evpn10
```

```
Bridging domain : __evpn10__, VLAN : 10
```

MAC address	MAC flags	Logical interface	NH Index	RTR ID
00:00:00:00:00:11	D	vtep.32769		
00:00:00:00:00:22	DC		1048576	1048576

Meaning

The bridging and EVPN MAC tables have learned the VLAN configurations.

Verifying PIM Reachability

Purpose

Verify that the PIM configuration is working properly on the CE and PE devices.

Action

On Device CE1, verify PIM configuration.

From operational mode, run the **show pim rps extensive** command.

```

user@CE1> show pim rps extensive
Instance: PIM.master

address-family INET

RP: 10.255.181.43
Learned via: static configuration
Mode: Sparse
Time Active: 00:06:08
Holdtime: 150
Device Index: 161
Subunit: 32769
Interface: pd-0/2/0.32769
Static RP Override: Off
Group Ranges:
    224.0.0.0/4
Register State for RP:

```

Group	Source	FirstHop	RP Address	State	Timeout
172.16.1.10	203.1.113.11	203.1.113.11	10.255.181.43	Receive	171

```

address-family INET6

```

From operational mode, run the **show pim join extensive** command.

```

user@CE1> show pim join extensive
Instance: PIM.master Family: INET
R = Rendezvous Point Tree, S = Sparse, W = Wildcard

Group: 172.16.1.10
Source: *
RP: 10.255.181.43
Flags: sparse,rptree,wildcard
Upstream interface: Local
Upstream neighbor: Local
Upstream state: Local RP
Uptime: 00:06:08
Downstream neighbors:
    Interface: ge-0/0/0.0 (assert winner)
    192.0.2.1 State: Join Flags: SRW Timeout: 201

```

```

        Uptime: 00:05:08 Time since last Join: 00:00:08
        Assert Winner: 192.0.2.2 Metric: 0 Pref: 2147483648 Timeout: 82
        Interface: Pseudo-VXLAN
        Number of downstream interfaces: 2

Group: 172.16.1.10
  Source: 10.255.181.43
  Flags: sparse,spt
  Upstream interface: Local
  Upstream neighbor: Local
  Upstream state: Local Source, Local RP, No Prune to RP
  Keepalive timeout: 338
  Uptime: 00:04:15
  Downstream neighbors:
    Interface: ge-0/0/0.0
      192.0.2.1 State: Join Flags: S Timeout: 201
      Uptime: 00:04:15 Time since last Join: 00:00:08
    Interface: Pseudo-VXLAN
    Number of downstream interfaces: 2

Group: 172.16.1.10
  Source: 203.1.113.11
  Flags: sparse,spt
  Upstream interface: ge-0/0/0.0
  Upstream neighbor: 192.0.2.1 (assert winner)
  Upstream state: Local RP, Join to Source, No Prune to RP
  Keepalive timeout: 338
  Uptime: 00:04:15
  Downstream neighbors:
    Interface: ge-0/0/0.0 (pruned)
      192.0.2.1 State: Prune Flags: SR Timeout: 201
      Uptime: 00:04:15 Time since last Prune: 00:00:08
      Assert Winner: 192.0.2.1 Metric: 0 Pref: 0 Timeout: 179
    Interface: Pseudo-VXLAN
    Number of downstream interfaces: 2

Instance: PIM.master Family: INET6
R = Rendezvous Point Tree, S = Sparse, W = Wildcard

```

Meaning

The device reachability using PIM is working as configured.

Verifying VXLAN Reachability

Purpose

Verify the connectivity between the VTEPs in the different data centers.

Action

From the operational mode, run the **show l2-learning vxlan-tunnel-end-point source**, **show l2-learning vxlan-tunnel-end-point remote**, and **show interfaces vtep** commands.

```
user@PE1> show l2-learning vxlan-tunnel-end-point source
```

Logical System Name	Id	SVTEP-IP	IFL	L3-Idx
<default>	0	203.1.113.11	lo0.0	7
L2-RTT	Bridge Domain		VNID	MC-Group-IP
evpn10	__evpn10__		10	172.16.1.10

```
user@PE2> show l2-learning vxlan-tunnel-end-point source
```

Logical System Name	Id	SVTEP-IP	IFL	L3-Idx
<default>	0	203.1.113.12	lo0.0	7
L2-RTT	Bridge Domain		VNID	MC-Group-IP
evpn10	__evpn10__		10	192.168.1.10

```
user@PE1> show l2-learning vxlan-tunnel-end-point remote
```

Logical System Name	Id	SVTEP-IP	IFL	L3-Idx
<default>	0	203.1.113.11	lo0.0	7
RVTEP-IP	IFL-Idx	NH-Id		
10.255.181.43	2684275660	2684275660		
VNID	MC-Group-IP			
10	172.16.1.10			

```
user@PE2> show l2-learning vxlan-tunnel-end-point remote
```

Logical System Name	Id	SVTEP-IP	IFL	L3-Idx
<default>	0	203.1.113.12	lo0.0	7
RVTEP-IP	IFL-Idx	NH-Id		
10.255.181.98	351	661		

VNID	MC-Group-IP
10	192.168.1.10

```

user@PE1> show interfaces vtep
Physical interface: vtep, Enabled, Physical link is Up
  Interface index: 133, SNMP ifIndex: 508
  Type: Software-Pseudo, Link-level type: VxLAN-Tunnel-Endpoint, MTU: 1600, Speed: Unlimited
  Device flags   : Present Running
  Interface flags: SNMP-Traps
  Link type      : Full-Duplex
  Link flags     : None
  Last flapped   : Never
    Input packets : 0
    Output packets: 0

  Logical interface vtep.32768 (Index 339) (SNMP ifIndex 560)
    Flags: Up SNMP-Traps Encapsulation: ENET2
    Ethernet segment value: 00:00:00:00:00:00:00:00:00:00, Mode: Single-homed, Multi-homed
status: Forwarding
    VXLAN Endpoint Type: Source, VXLAN Endpoint Address: 203.1.113.11, L2 Routing Instance:
evpn10, L3 Routing Instance: vrf
    Input packets : 0
    Output packets: 0

  Logical interface vtep.32769 (Index 341) (SNMP ifIndex 567)
    Flags: Up SNMP-Traps Encapsulation: ENET2
    VXLAN Endpoint Type: Remote, VXLAN Endpoint Address: 10.255.181.43, L2 Routing Instance:
evpn10, L3 Routing Instance: vrf
    Input packets : 143746
    Output packets: 95828
    Protocol bridge, MTU: 1600
    Flags: Trunk-Mode

```

Meaning

The output shows the correct tunnel source IP address (assigned to the loopback interface), VLAN, and multicast group for the VXLAN. Device PE1 is reachable because its IP address (the address assigned to the loopback interface) appears in the output. The output also shows that the VXLAN (VNI 10) and corresponding multicast group are configured correctly on the remote VTEP, Device PE2.

RELATED DOCUMENTATION

| [VXLAN Data Center Interconnect Using EVPN Overview](#) | 1123

Interconnecting EVPN-VXLAN Data Centers Through an EVPN-MPLS WAN

IN THIS CHAPTER

- [EVPN-VXLAN Data Center Interconnect Through EVPN-MPLS WAN Overview | 1163](#)
- [Example: Interconnecting EVPN-VXLAN Data Center Networks Through a WAN Running EVPN-based MPLS | 1175](#)
- [Overview of EVPN-VXLAN Interconnects through EVPN MPLS-MPLS WAN Using Gateways | 1356](#)

EVPN-VXLAN Data Center Interconnect Through EVPN-MPLS WAN Overview

IN THIS SECTION

- [Interconnection of Data Center Networks Through WAN Overview | 1164](#)
- [Multi-homing on Data Center Gateways | 1168](#)
- [EVPN Designated Forwarder \(DF\) Election | 1168](#)
- [Split Horizon | 1169](#)
- [Aliasing | 1169](#)
- [VLAN-Aware Bundle Service | 1170](#)

You can interconnect different data center networks running Ethernet VPN (EVPN) with Virtual extensible LAN (VXLAN) encapsulation through a WAN running MPLS-based EVPN.

The following sections describe the technology and implementation overview of interconnecting data center networks running EVPN-VXLAN through a WAN running EVPN-MPLS to be used as a data center interconnect (DCI) solution.

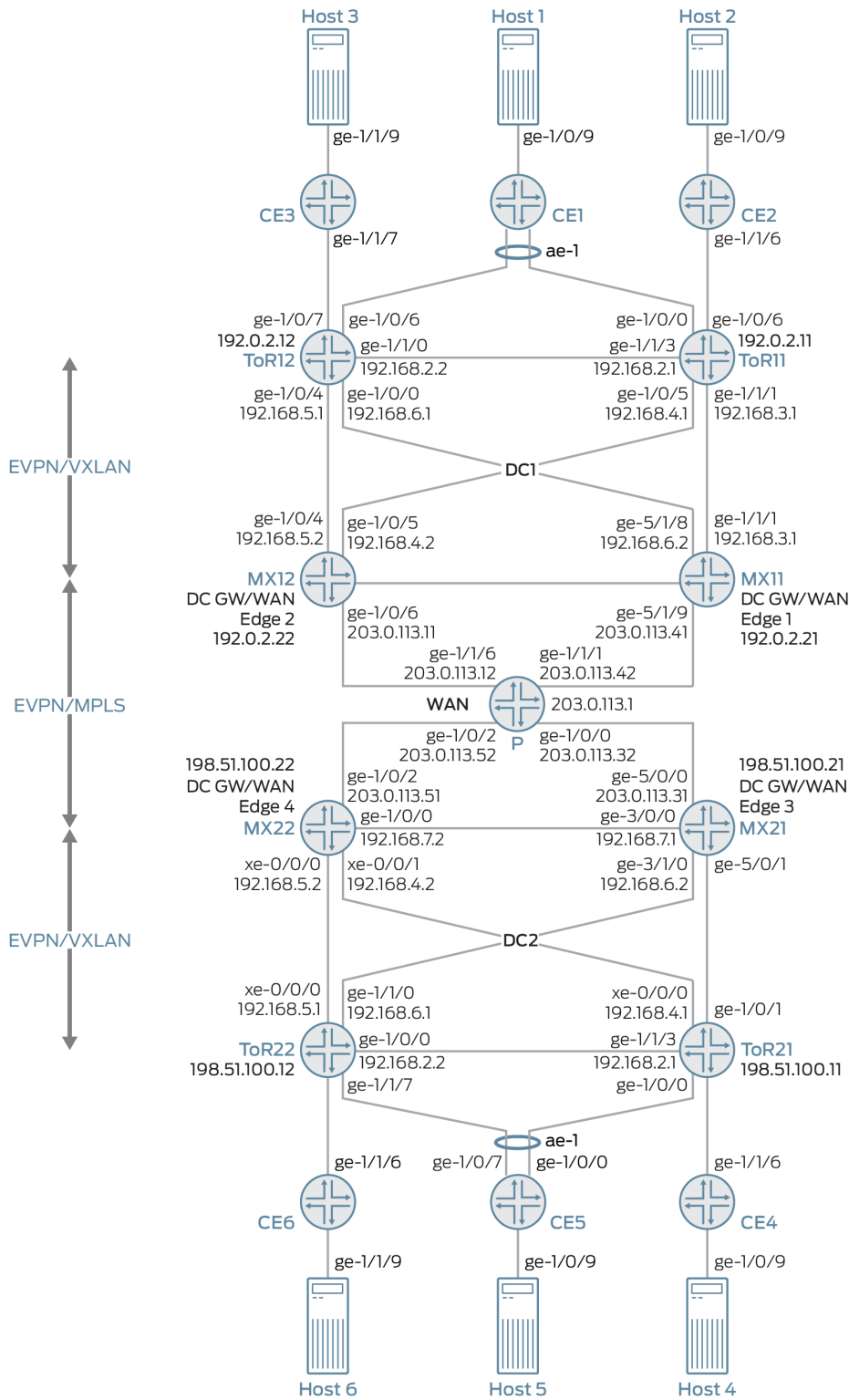
Interconnection of Data Center Networks Through WAN Overview

The following provides a conceptual overview of interconnecting different data center networks running Ethernet VPN (EVPN) with Virtual extensible LAN (VXLAN) encapsulation through a WAN running MPLS-based EVPN using the logical tunnel (It-) interface. You can:

- Connect data center edge routers over a WAN network running MPLS-based EVPN to achieve data center interconnection.
- Interconnect EVPN-VXLAN and EVPN-MPLS using the logical tunnel (It-) interface configured on the data center edge routers.

[Figure 131 on page 1166](#) illustration shows the interconnection of two data center networks (DC1 and DC2) running EVPN-VXLAN encapsulation through a WAN running MPLS-based EVPN:

Figure 131: EVPN-VXLAN Data Center Interconnect Through WAN Running EVPN-MPLS



In this illustration,

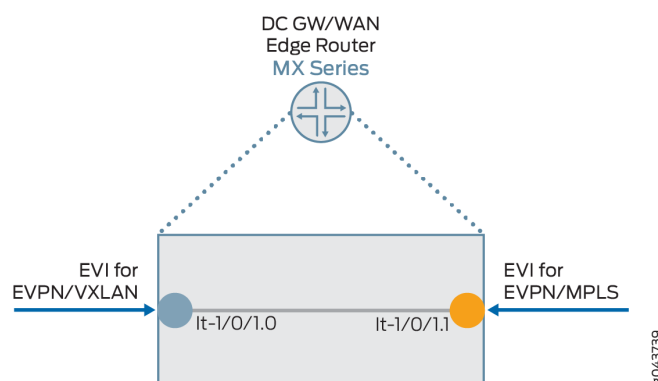
- The following devices are a part of the data center EVPN-VXLAN overlay network 1 (DC1):
 - Customer edge devices (CE1, CE2, and CE3) connected to the data center network.
 - VLAN hosts connected to each CE device.
 - MX routers playing the role of top-of-rack (ToR11 and ToR12) routers.
 - MX routers playing the role of the data center gateway router in the EVPN-VXLAN network and as a WAN edge router running MPLS-based EVPN (MX11 and MX12).
- The following devices are a part of the data center EVPN-VXLAN overlay network 2 (DC2):
 - Customer edge devices (CE4, CE5, and CE6) connected to the data center network.
 - VLAN hosts connected to each CE device.
 - MX routers playing the role of top-of-rack (ToR21 and ToR22) routers.
 - MX routers playing the role of the data center gateway router in the EVPN-VXLAN network and as a WAN edge router running MPLS-based EVPN (MX21 and MX22).

The interconnection of the data center network is realized on the data center gateway router through a pair of logical tunnel (It-) interface.

On the data center gateway router, you need to configure a pair of logical tunnel (It-) interface to interconnect the data center EVPN-VXLAN instance and the WAN MPLS-based EVPN instance: one logical tunnel (It-) interface is configured as the access interface for EVPN-VXLAN network and the other logical tunnel (It-) interface as the access interface for MPLS-based EVPN network as shown in the [Figure 132 on page 1168](#).

The support for active-active multi-homing is provided at the data center gateway routers for interconnection.

Figure 132: Logical Tunnel (It-) Interface of DC GW/WAN Edge Router Configured to Interconnect EVPN-VXLAN and EVPN-MPLS Instances



To configure EVPN-VXLAN and MPLS-based EVPN instances on the logical tunnel (It-) interface of the data center gateway router, see ["Example: Interconnecting EVPN-VXLAN Data Center Networks Through a WAN Running EVPN-based MPLS"](#) on page 1175.

To configure EVPN-VXLAN Data Center interconnection with EVPN-MPLS in WAN using an Inter-AS gateway, but without a logical tunnel interface, see *Example: Interconnecting EVPN-VXLAN Data Center with WAN Running EVPN-based MPLS Using a Gateway Model*.

Multi-homing on Data Center Gateways

You can configure redundant data center gateways and active-active multi-homing of EVPN-VXLAN network to a WAN running MPLS-based EVPN and active-active multi-homing of MPLS-based EVPN network to EVPN-VXLAN. This allows redundancy between the interconnection of EVPN-VXLAN network and MPLS-based EVPN WAN network. This also enables load-balancing of unicast traffic among the redundant data center gateways in both directions (from EVPN-VXLAN to EVPN-MPLS, as well as from EVPN-MPLS to EVPN-VXLAN). Broadcast, unknown unicast, and multicast (BUM) traffic is forwarded out of the data center by one of the data center gateways.

EVPN Designated Forwarder (DF) Election

To achieve active-active EVPN-VXLAN to EVPN-MPLS interconnect instance and active-active EVPN-MPLS to EVPN-VXLAN instance, the logical tunnel (It-) interface on the data center gateway router is configured with a non-zero Ethernet Segment Identifier (ESI). The ESI, a 10-octet value that must be unique across the entire network, is configured on a per port basis for the logical tunnel (It-) interface. As

per the EVPN multi-homing procedure defined in the RFC7432, the following routes are advertised for an EVPN instance (EVI):

- Advertise an Ethernet segment route
- Advertise an ESI auto-discovery route with a valid split-horizon label and mode set to multi-homing

The standard EVPN DF election procedure described in RFC7432 is considered. The DF election is based on per Ethernet segment for each EVI. The EVPN-VXLAN and EVPN-MPLS run its DF election process independently.

Split Horizon

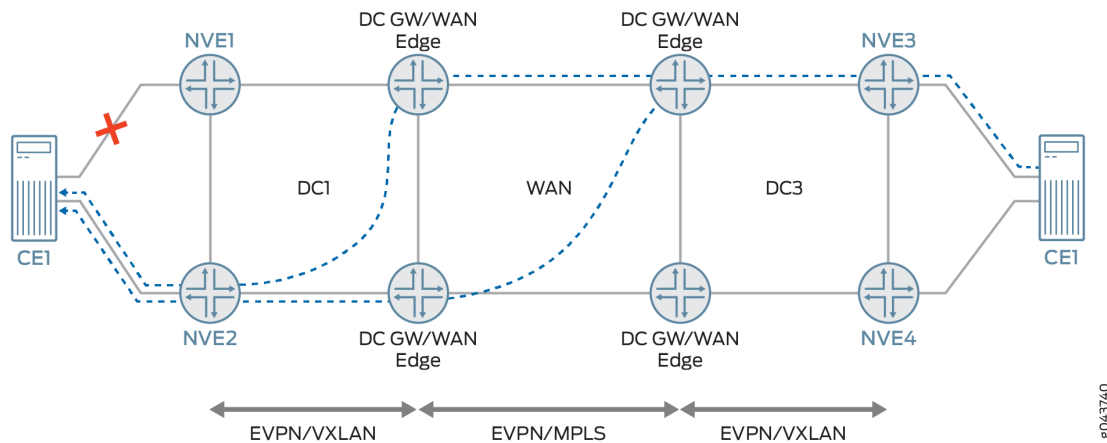
Split horizon prevents the looping of BUM traffic in a network, see RFC7432. For BUM traffic from core to the data center gateway (EVPN PE) direction, DF floods the BUM traffic to the access (It- interface) router and non-DF blocks the BUM traffic. When a DF or non-DF receives the BUM traffic coming from access (It- interface) router, it gets flooded to the core, but DF does not flood BUM traffic received from its non-DF to the access router based on split horizon rules. For a given BUM packet, only one copy is flooded to the access router (It- interface) and then to the EVPN core through one of the data center gateway routers because EVPN is multi-homed to another EVPN network. The DF filter rule from the first EVPN instance guarantees that only one copy of BUM traffic is forwarded from the DF to the It-interface before it re-enters the second EVPN instance.

Aliasing

When redundancy is configured in data center gateways, the traffic is load-balanced among redundant data center gateway routers on a per flow basis. MAC address is learned through the data plane using a pair of logical tunnel (It-) interfaces configured for EVPN-VXLAN instance and EVPN-MPLS instance for data center interconnect. However, the MAC owned by a host is always accessible by all the redundant data center gateways due to the nature of active-active multi-homing and full-mesh of EVPN PEs in both EVPN-VXLAN network and in WAN running EVPN-MPLS. Each EVPN instance on the data center gateway router declares the support of aliasing function for the ESI configured on the logical tunnel (It-) interface by advertising per EVI auto-discovery route. The aliasing functionality support is defined in the RFC7432.

Figure 133 on page 1170 illustrates a link failure between CE1 and NVE1 but CE1 is still reachable by both data center gateway routers within the data center network (DC1).

Figure 133: Load Balancing Among Redundant DC GW/WAN Edge Routers



A link failure between a host and its top-of-rack (TOR) devices does not impact the aliasing functionality declared by the data center gateway router as the data center network itself is active-active to the WAN running EVPN-MPLS. As long as the host is connected to another ToR device in the data center network, the host is still accessible by all the other redundant data center gateway routers, so the aliasing functionality applies.

VLAN-Aware Bundle Service

In Junos OS for MX Series, both EVPN-VXLAN and EVPN-MPLS instances support VLAN-aware bundle service with one or more bridge domains. To connect two EVIs with VLAN-aware bundle service through a pair of logical tunnel (lt-) interface, it needs trunk interface support on the logical tunnel (lt-) interface, as well as trunk interface support for both EVPN-VXLAN and EVPN-MPLS instances. A trunk interface on Junos OS MX Series allows a logical interface to accept packets tagged with any VLAN ID specified in a VLAN ID list.

When the trunk mode is used for the logical tunnel (lt-) interface, the frames going out of the logical tunnel (lt-) interface trunk port from the first EVPN virtual switch are tagged with the appropriate VLAN tag; going through its peer logical tunnel (lt-) interface, the incoming frames to the second virtual switch are inspected and forwarded based on the VLAN tag found within the frame.

The following is a sample configuration to use trunk mode on logical tunnel (lt-) interface to support VLAN-aware bundle service for interconnection of EVPN-VXLAN with a WAN running MPLS-based EVPN:

```

interfaces lt-1/0/10 {
esi {
    36:36:36:36:36:36:36:36:36:36;
    all-active;
}
unit 3 {
    encapsulation ethernet-bridge;
    peer-unit 4;
    family bridge {
        interface-mode trunk;
        vlan-id-list [ 51 52 ];
    }
}
unit 4 {
    encapsulation ethernet-bridge;
    peer-unit 3;
    family bridge {
        interface-mode trunk;
        vlan-id-list [ 51 52 ];
    }
}
}

```

The following is a sample configuration of trunk port support for EVPN-VXLAN and EVPN-MPLS:

```

routing-instances evpn-mpls {
vtep-source-interface lo0.0;
    instance-type virtual-switch;
    interface lt-1/0/10.4;
    route-distinguisher 101:2;
    vrf-target target:2:2;
    protocols {
        evpn {
            extended-vlan-list 51-52;
        }
    }
}
bridge-domains {

```

```

    bd1 {
        domain-type bridge;
        vlan-id 51;
    }
    bd2 {
        domain-type bridge;
        vlan-id 52;
    }
}
}
routing-instances red {
vtep-source-interface lo0.0;
    instance-type virtual-switch;
    interface lt-1/0/10.4
    route-distinguisher 101:1;
    vrf-target target:1:1;
    protocols {
        evpn {
            encapsulation vxlan;
            extended-vni-list all;
        }
    }
}
bridge-domains {
    bd1 {
        domain-type bridge;
        vlan-id 51;
        routing-interface irb.0;
        vxlan {
            vni 51;
            encapsulate-inner-vlan;
            decapsulate-accept-inner-vlan;
        }
    }
    bd2 {
        domain-type bridge;
        vlan-id 52;
        routing-interface irb.1;
        vxlan {
            vni 52;
            encapsulate-inner-vlan;
            decapsulate-accept-inner-vlan;
        }
    }
}

```

```
}
}
```

Data Center Network Design and Considerations

Before designing a data center network, you need to decide whether to use IGP, or iBGP, or eBGP protocols in the data center network for IP underlay. Another important factor to consider is the AS assignment. The ToR device in the data center network is required to have an AS number that is different than the AS number used in the WAN edge router.

For the overlay network, you need to decide whether to use iBGP, or eBGP, or a combination of both iBGP and eBGP.

Figure 134: Data Center Network Design

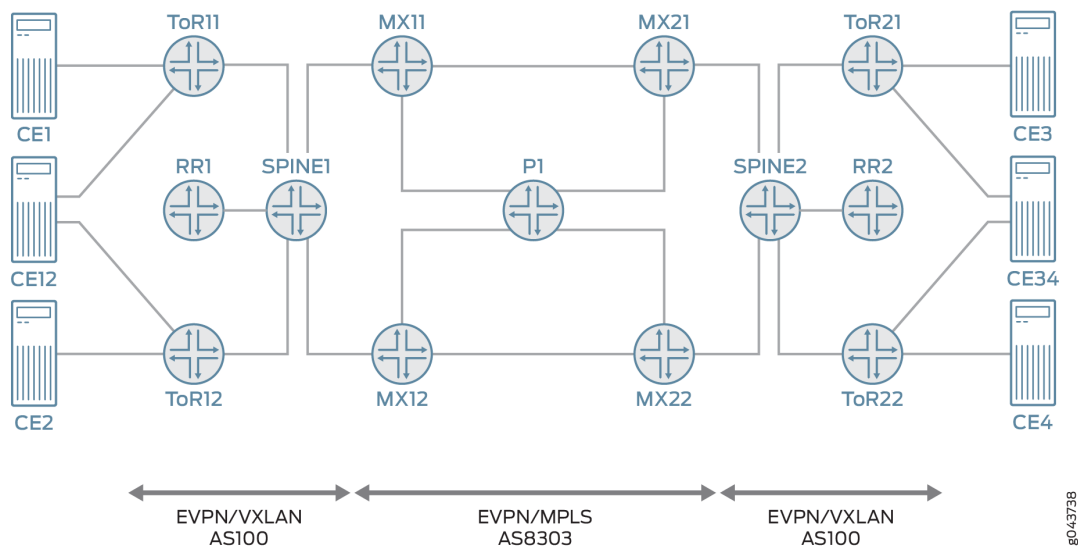


Figure 134 on page 1173 illustrates the MX routers (MX11, MX12, MX21 and MX22) as the data center gateway and WAN edge routers that interconnects EVPN-VXLAN to EVPN-MPLS. Spine switches provide connection for east and west traffic among ToRs so that traffic that does not need to be Layer 3 routed does not go through the MX routers. From a network design perspective, to provide an end-to-end EVPN solution, the following requirements must be met:

Isolate IGP Between EVPN-VXLAN and EVPN-MPLS Segments

When IGP is used in the data center network, you need to isolate the IP network in EVPN-VXLAN from the IP network in the WAN. When IGP is used in the data center, one option is not to run IGP protocol on the interfaces that connects spine switches and MX routers. Instead, an eBGP session with address

family inet unicast is used between spine switches and MX routers such that through IGP/eBGP/policy you can leak loopback addresses of ToR and MX routers to each other and still maintain the isolation of IP network in the data center from WAN. In the EVPN-VXLAN segment, IGP is between spine switches and ToRs only. In the EVPN-MPLS segment, IGP is between all the MX routers.

Using iBGP for IP Underlay in the Data Center Network

If the requirement is to not use IGP in the IP underlay in the data center, iBGP with address family inet unicast can be used to replace OSPF between spine switches and ToRs. Between spine switches and data center gateways, you still need to use eBGP for advertising loopback IP.

Using eBGP for the IP Underlay in the Data Center Network

If the requirement is to use eBGP only in the data center, you need to use eBGP with address family inet unicast for the IP underlay. In this case, it is a typical 2 stage CLOS network without spine aggregation layer. Each ToR and data center gateway is assigned a unique AS number. ToR establishes eBGP session with data center gateway routers directly.

Different Autonomous Systems (AS) in EVPN-VXLAN and EVPN-MPLS Network

The following is support for different AS in EVPN-VXLAN and EVPN-MPLS network running iBGP, or eBGP for the IP overlay.

Runing iBGP/eBGP for the Overlay

ToRs and spine switches are in the same AS100 and all MX Series routers are in AS8303. Among ToRs, its EVPN Network Layer Reachability Information (NLRI) is exchanged through iBGP session. A BGP route reflector (RR) is used and each ToR establishes iBGP session with the RR. Data traffic between ToRs that belongs to the same bridge domain goes through spine switch only and it is always 2 hops away. Since the ToRs and spine switches are in the same AS and the MX edge routers are in the different AS, the MX edge router establishes eBGP session to either RR or each ToR directly. By default, route learned from iBGP session (ToRs) are re-advertised to the eBGP (MX routers) and vice versa. BGP next-hop unchanged is enforced when BGP re-advertises EVPN NLRI among iBGP and eBGP session.

Running eBGP only for the Overlay

If the requirement is to run eBGP only in the data center, each ToR is assigned a unique AS number. Each data center gateway router uses a unique AS number on the data center facing side. For the WAN facing side, the same AS number is used, but the AS number would be different from the AS number used for the data center facing side. The AS number may also be reused in each data center.

To prevent an EVPN route in the data center to be advertised to the data center gateway routers in another data center, you must turn on the route constrain in the EVPN-MPLS network. To make BGP route constrain to work, different route target is used for EVPN-VXLAN and EVPN-MPLS network, respectively.

RELATED DOCUMENTATION

[Example: Interconnecting EVPN-VXLAN Data Center Networks Through a WAN Running EVPN-based MPLS | 1175](#)

Example: Interconnecting EVPN-VXLAN Data Center Networks Through a WAN Running EVPN-based MPLS

IN THIS SECTION

- [Requirements | 1175](#)
- [Overview | 1176](#)
- [Configuration | 1180](#)
- [Verification | 1275](#)

This example shows how to interconnect EVPN-VXLAN data center networks through a WAN running EVPN-MPLS to leverage the benefits of EVPN as a Data Center Interconnect (DCI) solution.

Requirements

This example uses the following hardware and software components:


- Four Juniper Networks MX Series routers to be configured as data center gateways and WAN edge routers.
- Four Juniper Networks MX Series routers to be configured as top-of-rack (ToR) routers.
- Six customer edge (CE) devices.
- Six host devices connected to each CE device that has the capability to configure multiple VLANs.
- One provider (P) router part of the EVPN-MPLS WAN network.

- Junos OS Release 17.2 or later.

Overview

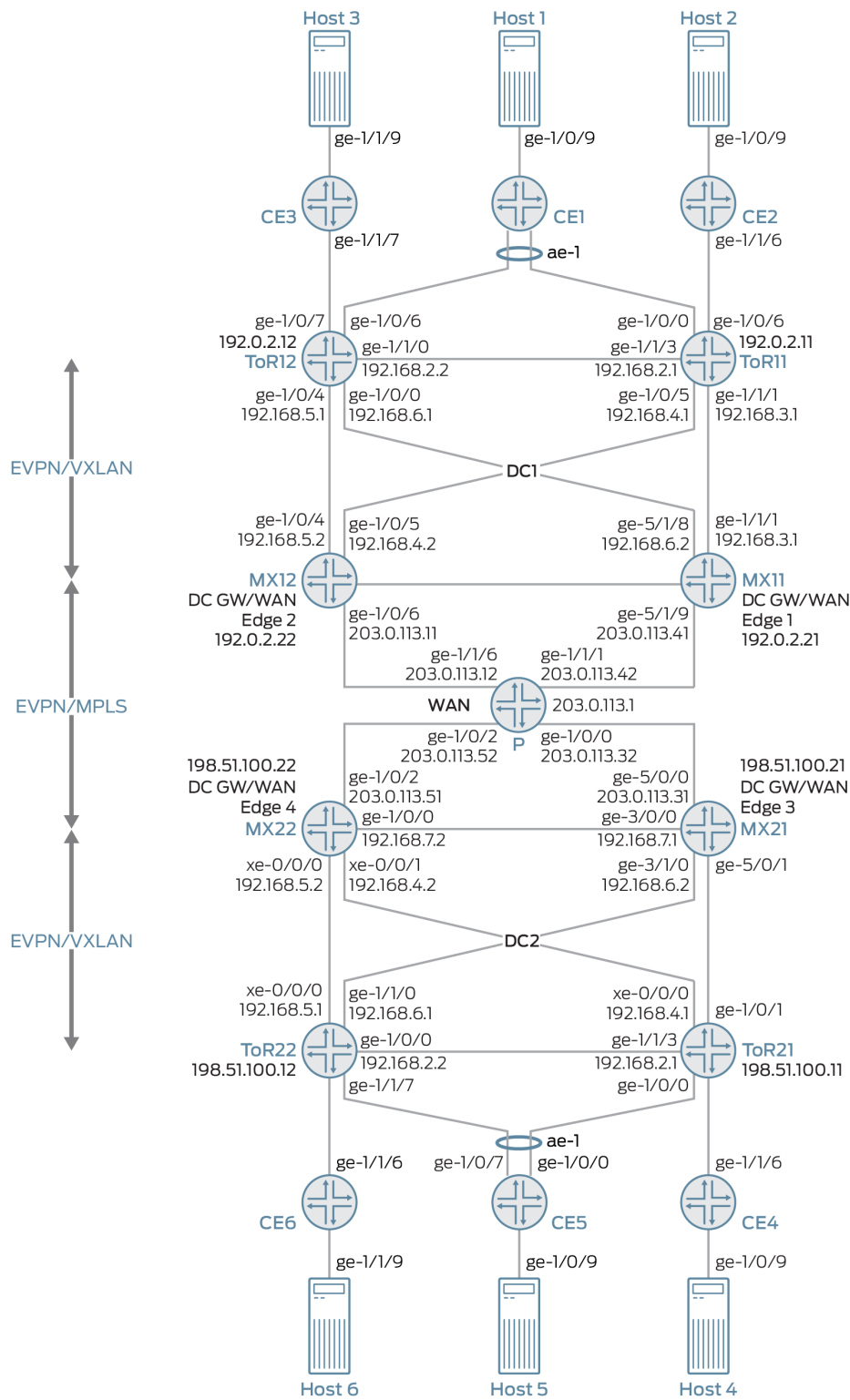
You can interconnect different data center networks running Ethernet VPN (EVPN) with Virtual extensible LAN (VXLAN) encapsulation through a WAN running MPLS-based EVPN using the logical tunnel (lt-) interface.

[Figure 135 on page 1178](#) illustrates the interconnection of data center networks running EVPN with VXLAN encapsulation through a WAN running MPLS-based EVPN. For the purposes of this example, the MX Series routers acting as data center gateways and as WAN edge routers are named MX11, MX12, MX21, and MX22. The MX Series routers acting as top-of-rack (ToR) routers are named ToR11, ToR12, ToR21, and ToR22. The customer edge (CE) devices connected to the data center network 1 (DC1) are named CE1, CE2, and CE3. The customer edge (CE) devices connected to the data center network 2 (DC2) are named CE4, CE5, and CE6. The host devices connected to each CE device should be able to configure multiple host VLANs. The WAN provider router is named P.



NOTE: CE devices are part of the logical system of ToR devices.

Figure 135: EVPN-VXLAN Data Center Interconnect Through WAN Running EVPN-MPLS



For the MX Series routers acting as data center gateways and WAN edge routers, configure the following information:

- IRB interfaces, virtual gateway addresses, and loopback logical interfaces.
- Multiprotocol external BGP (MP-EBGP) underlay connectivity between gateway and ToR routers, EVPN as the signaling protocol.
- Routing policies to allow specific routes into the virtual-switch tables.
- Routing instances (Layer 3 VRFs) for each virtual network, including a unique route distinguisher, and a vrf-target value.
- Virtual-switch instances (Layer 2 MAC-VRFs) for each virtual network, the VTEP source interface (always lo0.0), route distinguisher, and vrf-import policy.
- EVPN protocol, encapsulation method, VNI list, and BUM traffic forwarding method for each virtual switch.
- Bridge domain within each virtual switch that maps VNIDs to VLAN IDs, an IRB (Layer 3) interface, and the BUM forwarding method.

For the MX Series routers acting as top-of-rack (ToR) routers, configure the following information:

- Host facing interfaces with VLANs, VLAN IDs, and loopback logical interfaces.
- Link Aggregation Control Protocol (LACP)-enabled link aggregation group (LAG), Ethernet Segment ID (ESI), and all-active mode.
- Multiprotocol external BGP (MP-EBGP) overlays between ToR and gateway routers using EVPN as the signaling protocol.
- EVPN with VXLAN as the encapsulation method, extended-vni-list, multicast mode, and route targets for each VNI.
- Vrf-imp policy, vtep-source-interface, route-distinguisher, and vrf import and target information.
- VLANs, with VLAN IDs mapped to globally significant VNIs.

NOTE: You can set the virtual gateway address as the default IPv4 or IPv6 gateway address for end hosts (virtual machines or servers).

Configuration

IN THIS SECTION

- [CLI Quick Configuration | 1180](#)
- [Configuring ToR11 | 1212](#)
- [Configuring ToR12 | 1218](#)
- [Configuring Data Center Gateway and WAN Edge 1 Router \(MX11\) | 1225](#)
- [Configuring Data Center Gateway and WAN Edge 2 Router \(MX12\) | 1234](#)
- [Configuring Data Center Gateway and WAN Edge 3 Router \(MX21\) | 1245](#)
- [Configuring Data Center Gateway and WAN Edge 4 Router \(MX22\) | 1253](#)
- [Configuring ToR21 | 1262](#)
- [Configuring ToR22 | 1268](#)

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

ToR11

```
set system host-name ToR11
set logical-systems CE-2 interfaces ge-1/0/9 unit 0 description "CONNECTED TO Host-2"
set logical-systems CE-2 interfaces ge-1/0/9 unit 0 family bridge interface-mode trunk
set logical-systems CE-2 interfaces ge-1/0/9 unit 0 family bridge vlan-id-list 1-5
set logical-systems CE-2 interfaces ge-1/1/6 unit 0 description "CONNECTED TO ToR11"
set logical-systems CE-2 interfaces ge-1/1/6 unit 0 family bridge interface-mode trunk
set logical-systems CE-2 interfaces ge-1/1/6 unit 0 family bridge vlan-id-list 1-5
set logical-systems CE-2 bridge-domains BD-1 domain-type bridge
set logical-systems CE-2 bridge-domains BD-1 vlan-id 1
set logical-systems CE-2 bridge-domains BD-2 domain-type bridge
set logical-systems CE-2 bridge-domains BD-2 vlan-id 2
set logical-systems CE-2 bridge-domains BD-3 domain-type bridge
set logical-systems CE-2 bridge-domains BD-3 vlan-id 3
set logical-systems CE-2 bridge-domains BD-4 domain-type bridge
```

```

set logical-systems CE-2 bridge-domains BD-4 vlan-id 4
set logical-systems CE-2 bridge-domains BD-5 domain-type bridge
set logical-systems CE-2 bridge-domains BD-5 vlan-id 5
set chassis aggregated-devices ethernet device-count 1
set interfaces traceoptions file R0-DCD.log
set interfaces traceoptions file size 10m
set interfaces traceoptions flag all
set interfaces ge-1/0/5 unit 0 description "CONNECTED TO MX-12"
set interfaces ge-1/0/5 unit 0 family inet address 192.168.4.1/24
set interfaces ge-1/0/6 unit 0 description "CONNECTED TO CE-2"
set interfaces ge-1/0/6 unit 0 family bridge interface-mode trunk
set interfaces ge-1/0/6 unit 0 family bridge vlan-id-list 1-5
set interfaces ge-1/1/0 description "CONNECTED TO CE-1"
set interfaces ge-1/1/0 gigether-options 802.3ad ae0
set interfaces ge-1/1/1 unit 0 description "CONNECTED TO MX-11"
set interfaces ge-1/1/1 unit 0 family inet address 192.168.3.1/24
set interfaces ge-1/1/3 unit 0 description "CONNECTED TO ToR12"
set interfaces ge-1/1/3 unit 0 family inet address 192.168.2.1/24
set interfaces ae0 esi 00:11:11:11:11:11:11:11:11:11
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp periodic fast
set interfaces ae0 aggregated-ether-options lacp system-id 11:11:11:11:11:11
set interfaces ae0 unit 0 family bridge interface-mode trunk
set interfaces ae0 unit 0 family bridge vlan-id-list 1-5
set interfaces lo0 unit 81 family inet address 192.0.2.11/32
set routing-options router-id 192.0.2.11
set routing-options autonomous-system 100
set routing-options forwarding-table export evpn-pplb
set protocols bgp local-as 100
set protocols bgp group MX11 type external
set protocols bgp group MX11 local-address 192.168.3.1
set protocols bgp group MX11 export L0
set protocols bgp group MX11 export TEST
set protocols bgp group MX11 peer-as 400
set protocols bgp group MX11 neighbor 192.168.3.2 family inet unicast
set protocols bgp group MX12 type external
set protocols bgp group MX12 local-address 192.168.4.1
set protocols bgp group MX12 export L0
set protocols bgp group MX12 export TEST
set protocols bgp group MX12 peer-as 500
set protocols bgp group MX12 neighbor 192.168.4.2 family inet unicast
set protocols bgp group ToR12 type external

```



```

set protocols bgp group ToR12 local-address 192.168.2.1
set protocols bgp group ToR12 export L0
set protocols bgp group ToR12 export TEST
set protocols bgp group ToR12 peer-as 200
set protocols bgp group ToR12 local-as 100
set protocols bgp group ToR12 neighbor 192.168.2.2 family inet unicast
set protocols bgp group MX11-EVPN type external
set protocols bgp group MX11-EVPN multihop ttl 2
set protocols bgp group MX11-EVPN multihop no-nexthop-change
set protocols bgp group MX11-EVPN local-address 192.0.2.11
set protocols bgp group MX11-EVPN export TEST
set protocols bgp group MX11-EVPN peer-as 400
set protocols bgp group MX11-EVPN local-as 100
set protocols bgp group MX11-EVPN neighbor 192.0.2.21 family evpn signaling
set protocols bgp group MX12-EVPN type external
set protocols bgp group MX12-EVPN multihop ttl 2
set protocols bgp group MX12-EVPN multihop no-nexthop-change
set protocols bgp group MX12-EVPN local-address 192.0.2.11
set protocols bgp group MX12-EVPN export TEST
set protocols bgp group MX12-EVPN peer-as 500
set protocols bgp group MX12-EVPN local-as 100
set protocols bgp group MX12-EVPN neighbor 192.0.2.22 family evpn signaling
set protocols bgp group ToR12-EVPN type external
set protocols bgp group ToR12-EVPN multihop ttl 2
set protocols bgp group ToR12-EVPN multihop no-nexthop-change
set protocols bgp group ToR12-EVPN local-address 192.0.2.11
set protocols bgp group ToR12-EVPN export TEST
set protocols bgp group ToR12-EVPN peer-as 200
set protocols bgp group ToR12-EVPN local-as 100
set protocols bgp group ToR12-EVPN neighbor 192.0.2.12 family evpn signaling
set protocols l2-learning traceoptions file TOR11-L2ALD.log
set protocols l2-learning traceoptions file size 10m
set protocols l2-learning traceoptions level all
set protocols l2-learning traceoptions flag all
set policy-options policy-statement L0 term 1 from protocol direct
set policy-options policy-statement L0 term 1 from route-filter 192.0.2.11/32 exact
set policy-options policy-statement L0 term 1 then accept
set policy-options policy-statement TEST then community add NO-EXPORT
set policy-options policy-statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pplb then load-balance per-packet
set policy-options community NO-EXPORT members no-advertise
set policy-options community NO-EXPORT members no-export
set policy-options community NO-EXPORT members no-export-subconfed

```

```

set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.81
set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
set routing-instances EVPN-VXLAN-1 interface ge-1/0/6.0
set routing-instances EVPN-VXLAN-1 interface ae0.0
set routing-instances EVPN-VXLAN-1 route-distinguisher 192.0.2.11:1
set routing-instances EVPN-VXLAN-1 vrf-target target:1:1
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file TOR11-EVPN-VXLAN-1.log
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file size 10m
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions flag all
set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation vxlan
set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list 1-5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id 5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni 5

```

ToR12

```

set system host-name ToR12
set logical-systems CE-1 interfaces ge-1/0/9 unit 0 description "CONNECTED TO Host 1"
set logical-systems CE-1 interfaces ge-1/0/9 unit 0 family bridge interface-mode trunk
set logical-systems CE-1 interfaces ge-1/0/9 unit 0 family bridge vlan-id-list 1-5
set logical-systems CE-1 interfaces ae1 unit 0 description "CONNECTED TO ToR12"
set logical-systems CE-1 interfaces ae1 unit 0 family bridge interface-mode trunk
set logical-systems CE-1 interfaces ae1 unit 0 family bridge vlan-id-list 1-5
set logical-systems CE-1 bridge-domains BD-1 domain-type bridge
set logical-systems CE-1 bridge-domains BD-1 vlan-id 1
set logical-systems CE-1 bridge-domains BD-2 domain-type bridge
set logical-systems CE-1 bridge-domains BD-2 vlan-id 2
set logical-systems CE-1 bridge-domains BD-3 domain-type bridge

```

```

set logical-systems CE-1 bridge-domains BD-3 vlan-id 3
set logical-systems CE-1 bridge-domains BD-4 domain-type bridge
set logical-systems CE-1 bridge-domains BD-4 vlan-id 4
set logical-systems CE-1 bridge-domains BD-5 domain-type bridge
set logical-systems CE-1 bridge-domains BD-5 vlan-id 5
set logical-systems CE-3 interfaces ge-1/1/7 unit 0 description "CONNECTED TO ToR12"
set logical-systems CE-3 interfaces ge-1/1/7 unit 0 family bridge interface-mode trunk
set logical-systems CE-3 interfaces ge-1/1/7 unit 0 family bridge vlan-id-list 1-5
set logical-systems CE-3 interfaces ge-1/1/9 unit 0 description "CONNECTED TO Host 3"
set logical-systems CE-3 interfaces ge-1/1/9 unit 0 family bridge interface-mode trunk
set logical-systems CE-3 interfaces ge-1/1/9 unit 0 family bridge vlan-id-list 1-5
set logical-systems CE-3 bridge-domains BD-1 domain-type bridge
set logical-systems CE-3 bridge-domains BD-1 vlan-id 1
set logical-systems CE-3 bridge-domains BD-2 domain-type bridge
set logical-systems CE-3 bridge-domains BD-2 vlan-id 2
set logical-systems CE-3 bridge-domains BD-3 domain-type bridge
set logical-systems CE-3 bridge-domains BD-3 vlan-id 3
set logical-systems CE-3 bridge-domains BD-4 domain-type bridge
set logical-systems CE-3 bridge-domains BD-4 vlan-id 4
set logical-systems CE-3 bridge-domains BD-5 domain-type bridge
set logical-systems CE-3 bridge-domains BD-5 vlan-id 5
set chassis aggregated-devices ethernet device-count 2
set interfaces traceoptions file R1-DCD.log
set interfaces traceoptions file size 10m
set interfaces traceoptions flag all
set interfaces ge-1/0/0 unit 0 description "CONNECTED TO MX-11"
set interfaces ge-1/0/0 unit 0 family inet address 192.168.6.1/24
set interfaces ge-1/0/4 unit 0 description "CONNECTED TO MX12"
set interfaces ge-1/0/4 unit 0 family inet address 192.168.5.1/24
set interfaces ge-1/0/6 description "CONNECTED TO CE-1"
set interfaces ge-1/0/6 gigether-options 802.3ad ae0
set interfaces ge-1/0/7 unit 0 description "CONNECTED TO CE-3"
set interfaces ge-1/0/7 unit 0 family bridge interface-mode trunk
set interfaces ge-1/0/7 unit 0 family bridge vlan-id-list 1-5
set interfaces ge-1/1/0 description "CONNECTED TO ToR11"
set interfaces ge-1/1/0 gigether-options 802.3ad ae1
set interfaces ge-1/1/3 unit 0 description "CONNECTED TO ToR11"
set interfaces ge-1/1/3 unit 0 family inet address 192.168.2.2/24
set interfaces ge-1/1/6 description "CONNECTED TO ToR12"
set interfaces ge-1/1/6 gigether-options 802.3ad ae1
set interfaces ae0 esi 00:11:11:11:11:11:11:11:11:11
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp system-id 11:11:11:11:11:11

```

```
set interfaces ae0 unit 0 family bridge interface-mode trunk
set interfaces ae0 unit 0 family bridge vlan-id-list 1-5
set interfaces ae1 aggregated-ether-options lacp active
set interfaces ae1 aggregated-ether-options lacp periodic fast
set interfaces lo0 unit 82 family inet address 192.0.2.12/32
set routing-options router-id 192.0.2.12
set routing-options autonomous-system 200
set routing-options forwarding-table export evpn-pplb
set protocols bgp local-as 200
set protocols bgp group MX11 type external
set protocols bgp group MX11 local-address 192.168.6.1
set protocols bgp group MX11 export L0
set protocols bgp group MX11 export TEST
set protocols bgp group MX11 peer-as 400
set protocols bgp group MX11 local-as 200
set protocols bgp group MX11 neighbor 192.168.6.2 family inet unicast
set protocols bgp group MX12 type external
set protocols bgp group MX12 local-address 192.168.5.1
set protocols bgp group MX12 export L0
set protocols bgp group MX12 export TEST
set protocols bgp group MX12 peer-as 500
set protocols bgp group MX12 local-as 200
set protocols bgp group MX12 neighbor 192.168.5.2 family inet unicast
set protocols bgp group ToR11 type external
set protocols bgp group ToR11 local-address 192.168.2.2
set protocols bgp group ToR11 export L0
set protocols bgp group ToR11 export TEST
set protocols bgp group ToR11 peer-as 100
set protocols bgp group ToR11 local-as 200
set protocols bgp group ToR11 neighbor 192.168.2.1 family inet unicast
set protocols bgp group MX11-EVPN type external
set protocols bgp group MX11-EVPN multihop ttl 2
set protocols bgp group MX11-EVPN multihop no-nexthop-change
set protocols bgp group MX11-EVPN local-address 192.0.2.12
set protocols bgp group MX11-EVPN export TEST
set protocols bgp group MX11-EVPN peer-as 400
set protocols bgp group MX11-EVPN local-as 200
set protocols bgp group MX11-EVPN neighbor 192.0.2.21 family evpn signaling
set protocols bgp group MX12-EVPN type external
set protocols bgp group MX12-EVPN multihop ttl 2
set protocols bgp group MX12-EVPN multihop no-nexthop-change
set protocols bgp group MX12-EVPN local-address 192.0.2.12
set protocols bgp group MX12-EVPN export TEST
```

```

set protocols bgp group MX12-EVPN peer-as 500
set protocols bgp group MX12-EVPN local-as 200
set protocols bgp group MX12-EVPN neighbor 192.0.2.22 family evpn signaling
set protocols bgp group Tor11-EVPN type external
set protocols bgp group Tor11-EVPN multihop ttl 2
set protocols bgp group Tor11-EVPN multihop no-nexthop-change
set protocols bgp group Tor11-EVPN local-address 192.0.2.12
set protocols bgp group Tor11-EVPN export TEST
set protocols bgp group Tor11-EVPN peer-as 100
set protocols bgp group Tor11-EVPN local-as 200
set protocols bgp group Tor11-EVPN neighbor 192.0.2.11 family evpn signaling
set protocols bgp group Tor12-EVPN export TEST
set protocols l2-learning traceoptions file TOR12-L2ALD.log
set protocols l2-learning traceoptions file size 10m
set protocols l2-learning traceoptions level all
set protocols l2-learning traceoptions flag all
set policy-options policy-statement L0 term 1 from protocol direct
set policy-options policy-statement L0 term 1 from route-filter 192.0.2.12/32 exact
set policy-options policy-statement L0 term 1 then accept
set policy-options policy-statement TEST then community add NO-EXPORT
set policy-options policy-statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pplb then load-balance per-packet
set policy-options community NO-EXPORT members no-advertise
set policy-options community NO-EXPORT members no-export
set policy-options community NO-EXPORT members no-export-subconfed
set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.82
set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
set routing-instances EVPN-VXLAN-1 interface ge-1/0/7.0
set routing-instances EVPN-VXLAN-1 interface ae0.0
set routing-instances EVPN-VXLAN-1 route-distinguisher 192.0.2.12:1
set routing-instances EVPN-VXLAN-1 vrf-target target:1:1
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file TOR12-EVPN-VXLAN-1.log
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file size 10m
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions flag all
set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation vxlan
set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list 1-5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type bridge

```

```

set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id 5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni 5

```

Data Center Gateway and WAN Edge 1 Router (MX11)

```

set system host-name MX11
set interfaces ge-5/1/0 unit 0 description "CONNECTED TO MX21"
set interfaces ge-5/1/0 unit 0 family inet address 192.168.7.1/24
set interfaces lt-5/1/0 esi 00:22:22:22:22:22:22:22:22:22
set interfaces lt-5/1/0 esi all-active
set interfaces lt-5/1/0 unit 0 peer-unit 1
set interfaces lt-5/1/0 unit 0 family bridge interface-mode trunk
set interfaces lt-5/1/0 unit 0 family bridge vlan-id-list 1-5
set interfaces lt-5/1/0 unit 1 peer-unit 0
set interfaces lt-5/1/0 unit 1 family bridge interface-mode trunk
set interfaces lt-5/1/0 unit 1 family bridge vlan-id-list 1-5
set interfaces ge-5/1/1 unit 0 description "CONNECTED TO ToR11"
set interfaces ge-5/1/1 unit 0 family inet address 192.168.3.2/24
set interfaces ge-5/1/8 unit 0 description "CONNECTED TO ToR12"
set interfaces ge-5/1/8 unit 0 family inet address 192.168.6.2/24
set interfaces ge-5/1/9 unit 0 description "CONNECTED TO P"
set interfaces ge-5/1/9 unit 0 family inet address 203.0.1.1/24
set interfaces ge-5/1/9 unit 0 family mpls
set interfaces irb unit 1 proxy-macip-advertisement
set interfaces irb unit 1 virtual-gateway-esi 00:11:aa:aa:aa:aa:aa:aa:aa:aa
set interfaces irb unit 1 virtual-gateway-esi all-active
set interfaces irb unit 1 family inet address 10.11.1.12/24 virtual-gateway-address 10.11.1.10
set interfaces irb unit 2 proxy-macip-advertisement
set interfaces irb unit 2 virtual-gateway-esi 00:11:bb:bb:bb:bb:bb:bb:bb:bb
set interfaces irb unit 2 virtual-gateway-esi all-active
set interfaces irb unit 2 family inet address 10.12.1.12/24 virtual-gateway-address 10.12.1.10
set interfaces irb unit 3 proxy-macip-advertisement
set interfaces irb unit 3 virtual-gateway-esi 00:11:cc:cc:cc:cc:cc:cc:cc:cc
set interfaces irb unit 3 virtual-gateway-esi all-active
set interfaces irb unit 3 family inet address 10.13.1.12/24 virtual-gateway-address 10.13.1.10

```

```

set interfaces irb unit 4 proxy-macip-advertisement
set interfaces irb unit 4 virtual-gateway-esi 00:11:dd:dd:dd:dd:dd:dd:dd
set interfaces irb unit 4 virtual-gateway-esi all-active
set interfaces irb unit 4 family inet address 10.14.1.12/24 virtual-gateway-address 10.14.1.10
set interfaces irb unit 5 proxy-macip-advertisement
set interfaces irb unit 5 virtual-gateway-esi 00:11:ee:ee:ee:ee:ee:ee:ee
set interfaces irb unit 5 virtual-gateway-esi all-active
set interfaces irb unit 5 family inet address 10.15.1.12/24 virtual-gateway-address 10.15.1.10
set interfaces lo0 unit 84 family inet address 192.0.2.21/32
set interfaces lo0 unit 84 family mpls
set routing-options router-id 192.0.2.21
set routing-options autonomous-system 300
set routing-options forwarding-table export evpn-pplb
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path MX11-T0-MX12 to 192.0.2.22
set protocols mpls label-switched-path MX11-T0-P to 203.0.113.1
set protocols mpls label-switched-path MX11-T0-MX21 to 198.51.100.21
set protocols mpls label-switched-path MX11-T0-MX22 to 198.51.100.22
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp local-address 192.0.2.21
set protocols bgp local-as 300
set protocols bgp group INT type internal
set protocols bgp group INT local-address 192.0.2.21
set protocols bgp group INT family evpn signaling
set protocols bgp group INT export TEST
set protocols bgp group INT neighbor 203.0.113.1
set protocols bgp group ToR11 type external
set protocols bgp group ToR11 local-address 192.168.3.2
set protocols bgp group ToR11 import TEST
set protocols bgp group ToR11 export TEST
set protocols bgp group ToR11 export LO
set protocols bgp group ToR11 peer-as 100
set protocols bgp group ToR11 local-as 400
set protocols bgp group ToR11 neighbor 192.168.3.1 family inet unicast
set protocols bgp group ToR12 type external
set protocols bgp group ToR12 local-address 192.168.6.2
set protocols bgp group ToR12 export TEST
set protocols bgp group ToR12 export LO
set protocols bgp group ToR12 peer-as 200
set protocols bgp group ToR12 local-as 400
set protocols bgp group ToR12 neighbor 192.168.6.1 family inet unicast

```

```
set protocols bgp group MX12 type external
set protocols bgp group MX12 local-address 192.168.7.1
set protocols bgp group MX12 export TEST
set protocols bgp group MX12 export L0
set protocols bgp group MX12 peer-as 500
set protocols bgp group MX12 local-as 400
set protocols bgp group MX12 neighbor 192.168.7.2 family inet unicast
set protocols bgp group ToR11-EVPN type external
set protocols bgp group ToR11-EVPN multihop ttl 2
set protocols bgp group ToR11-EVPN multihop no-nexthop-change
set protocols bgp group ToR11-EVPN local-address 192.0.2.21
set protocols bgp group ToR11-EVPN export TEST
set protocols bgp group ToR11-EVPN peer-as 100
set protocols bgp group ToR11-EVPN local-as 400
set protocols bgp group ToR11-EVPN neighbor 192.0.2.11 family evpn signaling
set protocols bgp group ToR12-EVPN type external
set protocols bgp group ToR12-EVPN multihop ttl 2
set protocols bgp group ToR12-EVPN multihop no-nexthop-change
set protocols bgp group ToR12-EVPN local-address 192.0.2.21
set protocols bgp group ToR12-EVPN export TEST
set protocols bgp group ToR12-EVPN peer-as 200
set protocols bgp group ToR12-EVPN local-as 400
set protocols bgp group ToR12-EVPN neighbor 192.0.2.12 family evpn signaling
set protocols bgp group MX12-EVPN type external
set protocols bgp group MX12-EVPN multihop ttl 2
set protocols bgp group MX12-EVPN multihop no-nexthop-change
set protocols bgp group MX12-EVPN local-address 192.0.2.21
set protocols bgp group MX12-EVPN export TEST
set protocols bgp group MX12-EVPN peer-as 500
set protocols bgp group MX12-EVPN local-as 400
set protocols bgp group MX12-EVPN neighbor 192.0.2.22 family evpn signaling
set protocols bgp group MX11-EVPN export TEST
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-5/1/9.0
set protocols ospf area 0.0.0.0 interface lo0.84 passive
set protocols l2-learning traceoptions file MX11-L2ALD.log
set protocols l2-learning traceoptions file size 10m
set protocols l2-learning traceoptions level all
set protocols l2-learning traceoptions flag all
set policy-options policy-statement L0 term 1 from protocol direct
set policy-options policy-statement L0 term 1 from route-filter 192.0.2.21/32 exact
set policy-options policy-statement L0 term 1 then accept
set policy-options policy-statement L0 from protocol direct
```



```

set policy-options policy-statement LO from route-filter 192.0.2.21/32 exact
set policy-options policy-statement LO then accept
set policy-options policy-statement TEST then community add NO-EXPORT
set policy-options policy-statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pplb then load-balance per-packet
set policy-options community NO-EXPORT members no-advertise
set policy-options community NO-EXPORT members no-export
set policy-options community NO-EXPORT members no-export-subconfed
set routing-instances EVPN-MPLS-1 instance-type virtual-switch
set routing-instances EVPN-MPLS-1 interface lt-5/1/0.0
set routing-instances EVPN-MPLS-1 route-distinguisher 192.0.2.21:100
set routing-instances EVPN-MPLS-1 vrf-target target:1:2
set routing-instances EVPN-MPLS-1 protocols evpn traceoptions file MX11-EVPN-MPLS-1.log
set routing-instances EVPN-MPLS-1 protocols evpn traceoptions file size 10m
set routing-instances EVPN-MPLS-1 protocols evpn traceoptions flag all
set routing-instances EVPN-MPLS-1 protocols evpn extended-vlan-list 1-5
set routing-instances EVPN-MPLS-1 protocols evpn default-gateway no-gateway-community
set routing-instances EVPN-MPLS-1 bridge-domains BD-1 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-1 vlan-id 1
set routing-instances EVPN-MPLS-1 bridge-domains BD-2 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-2 vlan-id 2
set routing-instances EVPN-MPLS-1 bridge-domains BD-3 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-3 vlan-id 3
set routing-instances EVPN-MPLS-1 bridge-domains BD-4 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-4 vlan-id 4
set routing-instances EVPN-MPLS-1 bridge-domains BD-5 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-5 vlan-id 5
set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.84
set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
set routing-instances EVPN-VXLAN-1 interface lt-5/1/0.1
set routing-instances EVPN-VXLAN-1 route-distinguisher 192.0.2.21:1
set routing-instances EVPN-VXLAN-1 vrf-target target:1:1
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file MX11-EVPN-VXLAN-1.log
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file size 10m
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions flag all
set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation vxlan
set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list 1-5
set routing-instances EVPN-VXLAN-1 protocols evpn default-gateway no-gateway-community
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 routing-interface irb.1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type bridge

```

```

set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 routing-interface irb.2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 routing-interface irb.3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 routing-interface irb.4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id 5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 routing-interface irb.5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni 5
set routing-instances VRF instance-type vrf
set routing-instances VRF interface irb.1
set routing-instances VRF interface irb.2
set routing-instances VRF interface irb.3
set routing-instances VRF interface irb.4
set routing-instances VRF interface irb.5
set routing-instances VRF route-distinguisher 1:1
set routing-instances VRF vrf-target target:10:10

```

Data Center Gateway and WAN Edge 2 Router (MX12)

```

set system host-name MX12
set logical-systems P interfaces ge-1/0/0 unit 0 description "CONNECTED TO MX21"
set logical-systems P interfaces ge-1/0/0 unit 0 family inet address 203.0.4.2/24
set logical-systems P interfaces ge-1/0/0 unit 0 family mpls
set logical-systems P interfaces ge-1/0/2 unit 0 description "CONNECTED TO MX22"
set logical-systems P interfaces ge-1/0/2 unit 0 family inet address 203.0.3.2/24
set logical-systems P interfaces ge-1/0/2 unit 0 family mpls
set logical-systems P interfaces ge-1/1/1 unit 0 description "CONNECTED TO MX11"
set logical-systems P interfaces ge-1/1/1 unit 0 family inet address 203.0.1.2/24
set logical-systems P interfaces ge-1/1/1 unit 0 family mpls
set logical-systems P interfaces ge-1/1/6 unit 0 description "CONNECTED TO MX12"
set logical-systems P interfaces ge-1/1/6 unit 0 family inet address 203.0.2.2/24
set logical-systems P interfaces ge-1/1/6 unit 0 family mpls
set logical-systems P interfaces lo0 unit 86 family inet address 203.0.113.1/32
set logical-systems P interfaces lo0 unit 86 family mpls

```

```

set logical-systems P protocols rsvp interface all
set logical-systems P protocols mpls label-switched-path P-T0-MX11 from 203.0.113.1
set logical-systems P protocols mpls label-switched-path P-T0-MX11 to 192.0.2.21
set logical-systems P protocols mpls label-switched-path P-T0-MX12 to 192.0.2.22
set logical-systems P protocols mpls label-switched-path P-T0-MX21 to 198.51.100.21
set logical-systems P protocols mpls label-switched-path P-T0-MX22 to 198.51.100.22
set logical-systems P protocols mpls interface all
set logical-systems P protocols bgp local-address 203.0.113.1
set logical-systems P protocols bgp local-as 300
set logical-systems P protocols bgp group INT type internal
set logical-systems P protocols bgp group INT import BLOCK-VXLAN-ROUTES-FROM-CORE
set logical-systems P protocols bgp group INT family evpn signaling
set logical-systems P protocols bgp group INT cluster 203.0.113.1
set logical-systems P protocols bgp group INT neighbor 192.0.2.21
set logical-systems P protocols bgp group INT neighbor 192.0.2.22
set logical-systems P protocols bgp group INT neighbor 198.51.100.21
set logical-systems P protocols bgp group INT neighbor 198.51.100.22
set logical-systems P protocols ospf traffic-engineering
set logical-systems P protocols ospf area 0.0.0.0 interface all
set logical-systems P protocols ospf area 0.0.0.0 interface lo0.86
set logical-systems P policy-options policy-statement BLOCK-VXLAN-ROUTES-FROM-CORE term 1 from
protocol bgp
set logical-systems P policy-options policy-statement BLOCK-VXLAN-ROUTES-FROM-CORE term 1 from
community RT-CORE
set logical-systems P policy-options policy-statement BLOCK-VXLAN-ROUTES-FROM-CORE term 1 then
accept
set logical-systems P policy-options policy-statement BLOCK-VXLAN-ROUTES-FROM-CORE term 2 from
protocol bgp
set logical-systems P policy-options policy-statement BLOCK-VXLAN-ROUTES-FROM-CORE term 2 from
community RT-DC1
set logical-systems P policy-options policy-statement BLOCK-VXLAN-ROUTES-FROM-CORE term 2 then
reject
set logical-systems P policy-options policy-statement BLOCK-VXLAN-ROUTES-FROM-CORE term 3 from
protocol bgp
set logical-systems P policy-options policy-statement BLOCK-VXLAN-ROUTES-FROM-CORE term 3 from
community RT-DC2
set logical-systems P policy-options policy-statement BLOCK-VXLAN-ROUTES-FROM-CORE term 3 then
reject
set logical-systems P policy-options community RT-CORE members target:1:2
set logical-systems P policy-options community RT-DC1 members target:1:1
set logical-systems P policy-options community RT-DC2 members target:1:3
set logical-systems P routing-options router-id 203.0.113.1
set logical-systems P routing-options autonomous-system 300

```

```

set chassis fpc 1 pic 0 tunnel-services
set chassis network-services enhanced-ip
set interfaces traceoptions file R3-DCD.log
set interfaces traceoptions file size 10m
set interfaces traceoptions flag all
set interfaces lt-1/0/0 esi 00:22:22:22:22:22:22:22
set interfaces lt-1/0/0 esi all-active
set interfaces lt-1/0/0 unit 0 peer-unit 1
set interfaces lt-1/0/0 unit 0 family bridge interface-mode trunk
set interfaces lt-1/0/0 unit 0 family bridge vlan-id-list 1-5
set interfaces lt-1/0/0 unit 1 peer-unit 0
set interfaces lt-1/0/0 unit 1 family bridge interface-mode trunk
set interfaces lt-1/0/0 unit 1 family bridge vlan-id-list 1-5
set interfaces ge-1/0/4 unit 0 description "CONNECTED TO ToR12"
set interfaces ge-1/0/4 unit 0 family inet address 192.168.5.2/24
set interfaces ge-1/0/5 unit 0 description "CONNECTED TO TOR11"
set interfaces ge-1/0/5 unit 0 family inet address 192.168.4.2/24
set interfaces ge-1/0/6 unit 0 description "CONNECTED TO P"
set interfaces ge-1/0/6 unit 0 family inet address 203.0.2.1/24
set interfaces ge-1/0/6 unit 0 family mpls
set interfaces ge-1/1/0 unit 0 description "CONNECTED TO MX11"
set interfaces ge-1/1/0 unit 0 family inet address 192.168.7.2/24
set interfaces irb unit 1 proxy-macip-advertisement
set interfaces irb unit 1 virtual-gateway-esi 00:11:aa:aa:aa:aa:aa:aa
set interfaces irb unit 1 virtual-gateway-esi all-active
set interfaces irb unit 1 family inet address 10.11.1.13/24 virtual-gateway-address 10.11.1.10
set interfaces irb unit 2 proxy-macip-advertisement
set interfaces irb unit 2 virtual-gateway-esi 00:11:bb:bb:bb:bb:bb:bb
set interfaces irb unit 2 virtual-gateway-esi all-active
set interfaces irb unit 2 family inet address 10.12.1.13/24 virtual-gateway-address 10.12.1.10
set interfaces irb unit 3 proxy-macip-advertisement
set interfaces irb unit 3 virtual-gateway-esi 00:11:cc:cc:cc:cc:cc:cc
set interfaces irb unit 3 virtual-gateway-esi all-active
set interfaces irb unit 3 family inet address 10.13.1.13/24 virtual-gateway-address 10.13.1.10
set interfaces irb unit 4 proxy-macip-advertisement
set interfaces irb unit 4 virtual-gateway-esi 00:11:dd:dd:dd:dd:dd:dd
set interfaces irb unit 4 virtual-gateway-esi all-active
set interfaces irb unit 4 family inet address 10.14.1.13/24 virtual-gateway-address 10.14.1.10
set interfaces irb unit 5 proxy-macip-advertisement
set interfaces irb unit 5 virtual-gateway-esi 00:11:ee:ee:ee:ee:ee:ee
set interfaces irb unit 5 virtual-gateway-esi all-active
set interfaces irb unit 5 family inet address 10.15.1.13/24 virtual-gateway-address 10.15.1.10
set interfaces lo0 unit 85 family inet address 192.0.2.22/32

```

```
set interfaces lo0 unit 85 family mpls
set routing-options router-id 192.0.2.22
set routing-options autonomous-system 300
set routing-options forwarding-table export evpn-pplb
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path MX12-T0-MX11 to 192.0.2.21
set protocols mpls label-switched-path MX12-T0-P to 203.0.113.1
set protocols mpls label-switched-path MX12-T0-MX21 to 198.51.100.21
set protocols mpls label-switched-path MX12-T0-MX22 to 198.51.100.22
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp local-address 192.0.2.22
set protocols bgp local-as 300
set protocols bgp group INT type internal
set protocols bgp group INT family evpn signaling
set protocols bgp group INT export TEST
set protocols bgp group INT neighbor 203.0.113.1
set protocols bgp group ToR11 type external
set protocols bgp group ToR11 local-address 192.168.4.2
set protocols bgp group ToR11 export TEST
set protocols bgp group ToR11 export L0
set protocols bgp group ToR11 peer-as 100
set protocols bgp group ToR11 local-as 500
set protocols bgp group ToR11 neighbor 192.168.4.1 family inet unicast
set protocols bgp group ToR12 type external
set protocols bgp group ToR12 local-address 192.168.5.2
set protocols bgp group ToR12 export TEST
set protocols bgp group ToR12 export L0
set protocols bgp group ToR12 peer-as 200
set protocols bgp group ToR12 local-as 500
set protocols bgp group ToR12 neighbor 192.168.5.1 family inet unicast
set protocols bgp group MX11 type external
set protocols bgp group MX11 local-address 192.168.7.2
set protocols bgp group MX11 export TEST
set protocols bgp group MX11 export L0
set protocols bgp group MX11 peer-as 400
set protocols bgp group MX11 local-as 500
set protocols bgp group MX11 neighbor 192.168.7.1 family inet unicast
set protocols bgp group ToR11-EVPN type external
set protocols bgp group ToR11-EVPN multihop ttl 2
set protocols bgp group ToR11-EVPN multihop no-next-hop-change
set protocols bgp group ToR11-EVPN local-address 192.0.2.22
```

```
set protocols bgp group ToR11-EVPN export TEST
set protocols bgp group ToR11-EVPN peer-as 100
set protocols bgp group ToR11-EVPN local-as 500
set protocols bgp group ToR11-EVPN neighbor 192.0.2.11 family evpn signaling
set protocols bgp group ToR12-EVPN type external
set protocols bgp group ToR12-EVPN multihop ttl 2
set protocols bgp group ToR12-EVPN multihop no-nexthop-change
set protocols bgp group ToR12-EVPN local-address 192.0.2.22
set protocols bgp group ToR12-EVPN export TEST
set protocols bgp group ToR12-EVPN peer-as 200
set protocols bgp group ToR12-EVPN local-as 500
set protocols bgp group ToR12-EVPN neighbor 192.0.2.12 family evpn signaling
set protocols bgp group MX11-EVPN type external
set protocols bgp group MX11-EVPN multihop ttl 2
set protocols bgp group MX11-EVPN multihop no-nexthop-change
set protocols bgp group MX11-EVPN local-address 192.0.2.22
set protocols bgp group MX11-EVPN export TEST
set protocols bgp group MX11-EVPN peer-as 400
set protocols bgp group MX11-EVPN local-as 500
set protocols bgp group MX11-EVPN neighbor 192.0.2.21 family evpn signaling
set protocols bgp group MX12-EVPN export TEST
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-1/0/6.0
set protocols ospf area 0.0.0.0 interface lo0.85 passive
set protocols l2-learning traceoptions file MX12-L2ALD.log
set protocols l2-learning traceoptions file size 10m
set protocols l2-learning traceoptions level all
set protocols l2-learning traceoptions flag all
set policy-options policy-statement L0 term 1 from protocol direct
set policy-options policy-statement L0 term 1 from route-filter 192.0.2.22/32 exact
set policy-options policy-statement L0 term 1 then accept
set policy-options policy-statement L0 from protocol direct
set policy-options policy-statement L0 from route-filter 192.0.2.22/32 exact
set policy-options policy-statement L0 then accept
set policy-options policy-statement TEST from protocol bgp
set policy-options policy-statement TEST from protocol evpn
set policy-options policy-statement TEST then community add NO-EXPORT
set policy-options policy-statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pplb then load-balance per-packet
set policy-options community NO-EXPORT members no-advertise
set policy-options community NO-EXPORT members no-export
set policy-options community NO-EXPORT members no-export-subconfed
set routing-instances EVPN-MPLS-1 instance-type virtual-switch
```

```

set routing-instances EVPN-MPLS-1 interface lt-1/0/0.0
set routing-instances EVPN-MPLS-1 route-distinguisher 192.0.2.22:100
set routing-instances EVPN-MPLS-1 vrf-target target:1:2
set routing-instances EVPN-MPLS-1 protocols evpn traceoptions file MX12-EVPN-MPLS-1.log
set routing-instances EVPN-MPLS-1 protocols evpn traceoptions file size 10m
set routing-instances EVPN-MPLS-1 protocols evpn traceoptions flag all
set routing-instances EVPN-MPLS-1 protocols evpn extended-vlan-list 1-5
set routing-instances EVPN-MPLS-1 protocols evpn default-gateway no-gateway-community
set routing-instances EVPN-MPLS-1 bridge-domains BD-1 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-1 vlan-id 1
set routing-instances EVPN-MPLS-1 bridge-domains BD-2 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-2 vlan-id 2
set routing-instances EVPN-MPLS-1 bridge-domains BD-3 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-3 vlan-id 3
set routing-instances EVPN-MPLS-1 bridge-domains BD-4 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-4 vlan-id 4
set routing-instances EVPN-MPLS-1 bridge-domains BD-5 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-5 vlan-id 5
set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.85
set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
set routing-instances EVPN-VXLAN-1 interface lt-1/0/0.1
set routing-instances EVPN-VXLAN-1 route-distinguisher 192.0.2.22:1
set routing-instances EVPN-VXLAN-1 vrf-target target:1:1
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file MX12-EVPN-VXLAN-1.log
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file size 10m
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions flag all
set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation vxlan
set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list 1-4
set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list 5
set routing-instances EVPN-VXLAN-1 protocols evpn default-gateway no-gateway-community
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 routing-interface irb.1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 routing-interface irb.2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 routing-interface irb.3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type bridge

```

```

set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 routing-interface irb.4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id 5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 routing-interface irb.5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni 5
set routing-instances VRF instance-type vrf
set routing-instances VRF interface irb.1
set routing-instances VRF interface irb.2
set routing-instances VRF interface irb.3
set routing-instances VRF interface irb.4
set routing-instances VRF interface irb.5
set routing-instances VRF route-distinguisher 1:1
set routing-instances VRF vrf-target target:10:10

```

Data Center Gateway and WAN Edge 3 Router (MX21)

```

set system host-name MX21
set interfaces ge-3/0/0 unit 0 description "CONNECTED TO MX21"
set interfaces ge-3/0/0 unit 0 family inet address 192.168.13.1/24
set interfaces ge-3/1/0 unit 0 description "CONNECTED TO ToR22"
set interfaces ge-3/1/0 unit 0 family inet address 192.168.8.1/24
set interfaces ge-5/0/0 unit 0 description "CONNECTED TO P"
set interfaces ge-5/0/0 unit 0 family inet address 203.0.4.1/24
set interfaces ge-5/0/0 unit 0 family mpls
set interfaces lt-5/0/0 esi 00:33:33:33:33:33:33:33:33:33
set interfaces lt-5/0/0 esi all-active
set interfaces lt-5/0/0 unit 0 peer-unit 1
set interfaces lt-5/0/0 unit 0 family bridge interface-mode trunk
set interfaces lt-5/0/0 unit 0 family bridge vlan-id-list 1-5
set interfaces lt-5/0/0 unit 1 peer-unit 0
set interfaces lt-5/0/0 unit 1 family bridge interface-mode trunk
set interfaces lt-5/0/0 unit 1 family bridge vlan-id-list 1-5
set interfaces ge-5/0/1 unit 0 description "CONNECTED TO ToR21"
set interfaces ge-5/0/1 unit 0 family inet address 192.168.9.1/24
set interfaces irb unit 1 proxy-macip-advertisement
set interfaces irb unit 1 virtual-gateway-esi 00:22:aa:aa:aa:aa:aa:aa:aa:aa
set interfaces irb unit 1 virtual-gateway-esi all-active
set interfaces irb unit 1 family inet address 10.11.1.14/24 virtual-gateway-address 10.11.1.11
set interfaces irb unit 2 proxy-macip-advertisement

```



```

set interfaces irb unit 2 virtual-gateway-esi 00:22:bb:bb:bb:bb:bb:bb:bb
set interfaces irb unit 2 virtual-gateway-esi all-active
set interfaces irb unit 2 family inet address 10.12.1.14/24 virtual-gateway-address 10.12.1.11
set interfaces irb unit 3 proxy-macip-advertisement
set interfaces irb unit 3 virtual-gateway-esi 00:22:cc:cc:cc:cc:cc:cc:cc
set interfaces irb unit 3 virtual-gateway-esi all-active
set interfaces irb unit 3 family inet address 10.13.1.14/24 virtual-gateway-address 10.13.1.11
set interfaces irb unit 4 proxy-macip-advertisement
set interfaces irb unit 4 virtual-gateway-esi 00:22:dd:dd:dd:dd:dd:dd:dd
set interfaces irb unit 4 virtual-gateway-esi all-active
set interfaces irb unit 4 family inet address 10.14.1.14/24 virtual-gateway-address 10.14.1.11
set interfaces irb unit 5 proxy-macip-advertisement
set interfaces irb unit 5 virtual-gateway-esi 00:22:ee:ee:ee:ee:ee:ee:ee
set interfaces irb unit 5 virtual-gateway-esi all-active
set interfaces irb unit 5 family inet address 10.15.1.14/24 virtual-gateway-address 10.15.1.11
set interfaces lo0 unit 87 family inet address 198.51.100.21/32
set interfaces lo0 unit 87 family mpls
set routing-options router-id 198.51.100.21
set routing-options autonomous-system 300
set routing-options forwarding-table export evpn-pplb
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path MX21-T0-MX11 to 192.0.2.21
set protocols mpls label-switched-path MX21-T0-MX12 to 192.0.2.22
set protocols mpls label-switched-path MX21-T0-P to 203.0.113.1
set protocols mpls label-switched-path MX21-T0-MX22 to 198.51.100.22
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp local-address 198.51.100.21
set protocols bgp export TEST
set protocols bgp local-as 300
set protocols bgp group INT type internal
set protocols bgp group INT local-address 198.51.100.21
set protocols bgp group INT family evpn signaling
set protocols bgp group INT export TEST1
set protocols bgp group INT neighbor 203.0.113.1
set protocols bgp group ToR21 type external
set protocols bgp group ToR21 local-address 192.168.9.1
set protocols bgp group ToR21 export TEST
set protocols bgp group ToR21 export L0
set protocols bgp group ToR21 peer-as 600
set protocols bgp group ToR21 local-as 800
set protocols bgp group ToR21 neighbor 192.168.9.2 family inet unicast

```

```
set protocols bgp group ToR22 type external
set protocols bgp group ToR22 local-address 192.168.8.1
set protocols bgp group ToR22 export TEST
set protocols bgp group ToR22 export L0
set protocols bgp group ToR22 peer-as 700
set protocols bgp group ToR22 local-as 800
set protocols bgp group ToR22 neighbor 192.168.8.2 family inet unicast
set protocols bgp group MX22 type external
set protocols bgp group MX22 local-address 192.168.13.1
set protocols bgp group MX22 export TEST
set protocols bgp group MX22 export L0
set protocols bgp group MX22 peer-as 900
set protocols bgp group MX22 local-as 800
set protocols bgp group MX22 neighbor 10.115.15.2 family inet unicast
set protocols bgp group ToR21-EVPN type external
set protocols bgp group ToR21-EVPN multihop ttl 2
set protocols bgp group ToR21-EVPN multihop no-nexthop-change
set protocols bgp group ToR21-EVPN local-address 198.51.100.21
set protocols bgp group ToR21-EVPN peer-as 600
set protocols bgp group ToR21-EVPN local-as 800
set protocols bgp group ToR21-EVPN neighbor 198.51.100.11 family evpn signaling
set protocols bgp group ToR22-EVPN type external
set protocols bgp group ToR22-EVPN multihop ttl 2
set protocols bgp group ToR22-EVPN multihop no-nexthop-change
set protocols bgp group ToR22-EVPN local-address 198.51.100.21
set protocols bgp group ToR22-EVPN peer-as 700
set protocols bgp group ToR22-EVPN local-as 800
set protocols bgp group ToR22-EVPN neighbor 198.51.100.12 family evpn signaling
set protocols bgp group MX22-EVPN type external
set protocols bgp group MX22-EVPN multihop ttl 2
set protocols bgp group MX22-EVPN multihop no-nexthop-change
set protocols bgp group MX22-EVPN local-address 198.51.100.21
set protocols bgp group MX22-EVPN peer-as 900
set protocols bgp group MX22-EVPN local-as 800
set protocols bgp group MX22-EVPN neighbor 198.51.100.22 family evpn signaling
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-5/0/0.0
set protocols ospf area 0.0.0.0 interface lo0.87 passive
set protocols l2-learning traceoptions file MX21-L2ALD.log
set protocols l2-learning traceoptions file size 10m
set protocols l2-learning traceoptions level all
set protocols l2-learning traceoptions flag all
set policy-options policy-statement L0 from protocol direct
```

```

set policy-options policy-statement L0 from route-filter 198.51.100.21/32 exact
set policy-options policy-statement L0 then accept
set policy-options policy-statement TEST then community add NO-EXPORT
set policy-options policy-statement TEST1 term 1 from protocol bgp
set policy-options policy-statement TEST1 term 1 from external
set policy-options policy-statement TEST1 term 1 then reject
set policy-options policy-statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pplb then load-balance per-packet
set policy-options community NO-EXPORT members no-advertise
set policy-options community NO-EXPORT members no-export
set policy-options community NO-EXPORT members no-export-subconfed
set routing-instances EVPN-MPLS-1 instance-type virtual-switch
set routing-instances EVPN-MPLS-1 interface lt-5/0/0.0
set routing-instances EVPN-MPLS-1 route-distinguisher 198.51.100.21:100
set routing-instances EVPN-MPLS-1 vrf-target target:1:2
set routing-instances EVPN-MPLS-1 protocols evpn traceoptions file MX21-EVPN-MPLS-1.log
set routing-instances EVPN-MPLS-1 protocols evpn traceoptions file size 10m
set routing-instances EVPN-MPLS-1 protocols evpn traceoptions flag all
set routing-instances EVPN-MPLS-1 protocols evpn extended-vlan-list 1-5
set routing-instances EVPN-MPLS-1 protocols evpn default-gateway no-gateway-community
set routing-instances EVPN-MPLS-1 bridge-domains BD-1 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-1 vlan-id 1
set routing-instances EVPN-MPLS-1 bridge-domains BD-2 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-2 vlan-id 2
set routing-instances EVPN-MPLS-1 bridge-domains BD-3 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-3 vlan-id 3
set routing-instances EVPN-MPLS-1 bridge-domains BD-4 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-4 vlan-id 4
set routing-instances EVPN-MPLS-1 bridge-domains BD-5 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-5 vlan-id 5
set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.87
set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
set routing-instances EVPN-VXLAN-1 interface lt-5/0/0.1
set routing-instances EVPN-VXLAN-1 route-distinguisher 198.51.100.21:1
set routing-instances EVPN-VXLAN-1 vrf-target target:1:3
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file MX21-EVPN-VXLAN-1.log
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file size 10m
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions flag all
set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation vxlan
set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list 1-5
set routing-instances EVPN-VXLAN-1 protocols evpn default-gateway no-gateway-community
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id 1

```

```

set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 routing-interface irb.1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 routing-interface irb.2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 routing-interface irb.3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 routing-interface irb.4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id 5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 routing-interface irb.5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni 5
set routing-instances VRF instance-type vrf
set routing-instances VRF interface irb.1
set routing-instances VRF interface irb.2
set routing-instances VRF interface irb.3
set routing-instances VRF interface irb.4
set routing-instances VRF interface irb.5
set routing-instances VRF route-distinguisher 1:1
set routing-instances VRF vrf-target target:10:10

```

Data Center Gateway and WAN Edge 4 Router (MX22)

```

set system host-name MX22
set interfaces xe-0/0/0 unit 0 description "CONNECTED TO ToR22"
set interfaces xe-0/0/0 unit 0 family inet address 192.168.11.1/24
set interfaces xe-0/0/1 unit 0 description "CONNECTED TO ToR21"
set interfaces xe-0/0/1 unit 0 family inet address 192.168.10.1/24
set interfaces ge-1/0/0 unit 0 description "CONNECTED TO MX21"
set interfaces ge-1/0/0 unit 0 family inet address 10.115.15.2/24
set interfaces lt-1/0/0 esi 00:33:33:33:33:33:33:33:33:33
set interfaces lt-1/0/0 esi all-active
set interfaces lt-1/0/0 unit 0 peer-unit 1
set interfaces lt-1/0/0 unit 0 family bridge interface-mode trunk
set interfaces lt-1/0/0 unit 0 family bridge vlan-id-list 1-5

```

```

set interfaces lt-1/0/0 unit 1 peer-unit 0
set interfaces lt-1/0/0 unit 1 family bridge interface-mode trunk
set interfaces lt-1/0/0 unit 1 family bridge vlan-id-list 1-5
set interfaces ge-1/0/2 unit 0 description "CONNECTED TO P"
set interfaces ge-1/0/2 unit 0 family inet address 203.0.3.1/24
set interfaces ge-1/0/2 unit 0 family mpls
set interfaces irb unit 1 proxy-macip-advertisement
set interfaces irb unit 1 virtual-gateway-esi 00:22:aa:aa:aa:aa:aa:aa:aa
set interfaces irb unit 1 virtual-gateway-esi all-active
set interfaces irb unit 1 family inet address 10.11.1.15/24 virtual-gateway-address 10.11.1.11
set interfaces irb unit 2 proxy-macip-advertisement
set interfaces irb unit 2 virtual-gateway-esi 00:22:bb:bb:bb:bb:bb:bb:bb
set interfaces irb unit 2 virtual-gateway-esi all-active
set interfaces irb unit 2 family inet address 10.12.1.15/24 virtual-gateway-address 10.12.1.11
set interfaces irb unit 3 proxy-macip-advertisement
set interfaces irb unit 3 virtual-gateway-esi 00:22:cc:cc:cc:cc:cc:cc:cc
set interfaces irb unit 3 virtual-gateway-esi all-active
set interfaces irb unit 3 family inet address 10.13.1.15/24 virtual-gateway-address 10.13.1.11
set interfaces irb unit 4 proxy-macip-advertisement
set interfaces irb unit 4 virtual-gateway-esi 00:22:dd:dd:dd:dd:dd:dd:dd
set interfaces irb unit 4 virtual-gateway-esi all-active
set interfaces irb unit 4 family inet address 10.14.1.15/24 virtual-gateway-address 10.14.1.11
set interfaces irb unit 5 proxy-macip-advertisement
set interfaces irb unit 5 virtual-gateway-esi 00:22:ee:ee:ee:ee:ee:ee:ee
set interfaces irb unit 5 virtual-gateway-esi all-active
set interfaces irb unit 5 family inet address 10.15.1.15/24 virtual-gateway-address 10.15.1.11
set interfaces lo0 unit 88 family inet address 198.51.100.22/32
set routing-options router-id 198.51.100.22
set routing-options autonomous-system 300
set routing-options forwarding-table export evpn-pplb
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path MX22-T0-MX11 to 192.0.2.21
set protocols mpls label-switched-path MX22-T0-MX12 to 192.0.2.22
set protocols mpls label-switched-path MX22-T0-P to 203.0.113.1
set protocols mpls label-switched-path MX22-T0-MX21 to 198.51.100.21
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp local-address 198.51.100.22
set protocols bgp export TEST
set protocols bgp local-as 300
set protocols bgp group INT type internal
set protocols bgp group INT family evpn signaling

```

```
set protocols bgp group INT export TEST1
set protocols bgp group INT neighbor 203.0.113.1
set protocols bgp group ToR21 type external
set protocols bgp group ToR21 local-address 192.168.10.1
set protocols bgp group ToR21 export TEST
set protocols bgp group ToR21 export L0
set protocols bgp group ToR21 peer-as 600
set protocols bgp group ToR21 local-as 900
set protocols bgp group ToR21 neighbor 10.102.2.1 family inet unicast
set protocols bgp group ToR22 type external
set protocols bgp group ToR22 local-address 192.168.11.1
set protocols bgp group ToR22 export TEST
set protocols bgp group ToR22 export L0
set protocols bgp group ToR22 peer-as 700
set protocols bgp group ToR22 local-as 900
set protocols bgp group ToR22 neighbor 192.168.11.2 family inet unicast
set protocols bgp group MX21 type external
set protocols bgp group MX21 local-address 10.115.15.2
set protocols bgp group MX21 export TEST
set protocols bgp group MX21 export L0
set protocols bgp group MX21 peer-as 800
set protocols bgp group MX21 local-as 900
set protocols bgp group MX21 neighbor 192.168.13.1 family inet unicast
set protocols bgp group ToR21-EVPN type external
set protocols bgp group ToR21-EVPN multihop ttl 2
set protocols bgp group ToR21-EVPN multihop no-nexthop-change
set protocols bgp group ToR21-EVPN local-address 198.51.100.22
set protocols bgp group ToR21-EVPN peer-as 600
set protocols bgp group ToR21-EVPN local-as 900
set protocols bgp group ToR21-EVPN neighbor 198.51.100.11 family evpn signaling
set protocols bgp group ToR22-EVPN type external
set protocols bgp group ToR22-EVPN multihop ttl 2
set protocols bgp group ToR22-EVPN multihop no-nexthop-change
set protocols bgp group ToR22-EVPN local-address 198.51.100.22
set protocols bgp group ToR22-EVPN peer-as 700
set protocols bgp group ToR22-EVPN local-as 900
set protocols bgp group ToR22-EVPN neighbor 198.51.100.12 family evpn signaling
set protocols bgp group MX21-EVPN type external
set protocols bgp group MX21-EVPN multihop ttl 2
set protocols bgp group MX21-EVPN multihop no-nexthop-change
set protocols bgp group MX21-EVPN local-address 198.51.100.22
set protocols bgp group MX21-EVPN peer-as 800
set protocols bgp group MX21-EVPN local-as 900
```

```
set protocols bgp group MX21-EVPN neighbor 198.51.100.21 family evpn signaling
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-1/0/2.0
set protocols ospf area 0.0.0.0 interface lo0.88 passive
set protocols l2-learning traceoptions file MX22-L2ALD.log
set protocols l2-learning traceoptions file size 10m
set protocols l2-learning traceoptions level all
set protocols l2-learning traceoptions flag all
set policy-options policy-statement L0 from protocol direct
set policy-options policy-statement L0 from route-filter 198.51.100.22/32 exact
set policy-options policy-statement L0 then accept
set policy-options policy-statement TEST then community add NO-EXPORT
set policy-options policy-statement TEST1 term 1 from protocol bgp
set policy-options policy-statement TEST1 term 1 from external
set policy-options policy-statement TEST1 term 1 then reject
set policy-options policy-statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pplb then load-balance per-packet
set policy-options community NO-EXPORT members no-advertise
set policy-options community NO-EXPORT members no-export
set policy-options community NO-EXPORT members no-export-subconfed
set routing-instances EVPN-MPLS-1 instance-type virtual-switch
set routing-instances EVPN-MPLS-1 interface lt-1/0/0.0
set routing-instances EVPN-MPLS-1 route-distinguisher 198.51.100.22:100
set routing-instances EVPN-MPLS-1 vrf-target target:1:2
set routing-instances EVPN-MPLS-1 protocols evpn traceoptions file MX22-EVPN-MPLS-1.log
set routing-instances EVPN-MPLS-1 protocols evpn traceoptions file size 10m
set routing-instances EVPN-MPLS-1 protocols evpn traceoptions flag all
set routing-instances EVPN-MPLS-1 protocols evpn extended-vlan-list 1-5
set routing-instances EVPN-MPLS-1 protocols evpn default-gateway no-gateway-community
set routing-instances EVPN-MPLS-1 bridge-domains BD-1 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-1 vlan-id 1
set routing-instances EVPN-MPLS-1 bridge-domains BD-2 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-2 vlan-id 2
set routing-instances EVPN-MPLS-1 bridge-domains BD-3 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-3 vlan-id 3
set routing-instances EVPN-MPLS-1 bridge-domains BD-4 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-4 vlan-id 4
set routing-instances EVPN-MPLS-1 bridge-domains BD-5 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-5 vlan-id 5
set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.88
set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
set routing-instances EVPN-VXLAN-1 interface lt-1/0/0.1
set routing-instances EVPN-VXLAN-1 route-distinguisher 198.51.100.22:1
```

```

set routing-instances EVPN-VXLAN-1 vrf-target target:1:3
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file MX22-EVPN-VXLAN-1.log
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file size 10m
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions flag all
set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation vxlan
set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list 1-5
set routing-instances EVPN-VXLAN-1 protocols evpn default-gateway no-gateway-community
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 routing-interface irb.1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 routing-interface irb.2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 routing-interface irb.3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 routing-interface irb.4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id 5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 routing-interface irb.5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni 5
set routing-instances VRF instance-type vrf
set routing-instances VRF interface irb.1
set routing-instances VRF interface irb.2
set routing-instances VRF interface irb.3
set routing-instances VRF interface irb.4
set routing-instances VRF interface irb.5
set routing-instances VRF route-distinguisher 1:1
set routing-instances VRF vrf-target target:10:10

```

ToR-21

```

set system host-name ToR21
set logical-systems CE-4 interfaces ge-1/0/9 unit 0 description "CONNECTED TO Host 4"
set logical-systems CE-4 interfaces ge-1/0/9 unit 0 family bridge interface-mode trunk

```



```

set logical-systems CE-4 interfaces ge-1/0/9 unit 0 family bridge vlan-id-list 1-5
set logical-systems CE-4 interfaces ge-1/1/6 unit 0 description "CONNECTED TO ToR21"
set logical-systems CE-4 interfaces ge-1/1/6 unit 0 family bridge interface-mode trunk
set logical-systems CE-4 interfaces ge-1/1/6 unit 0 family bridge vlan-id-list 1-5
set logical-systems CE-4 bridge-domains BD-1 domain-type bridge
set logical-systems CE-4 bridge-domains BD-1 vlan-id 1
set logical-systems CE-4 bridge-domains BD-2 domain-type bridge
set logical-systems CE-4 bridge-domains BD-2 vlan-id 2
set logical-systems CE-4 bridge-domains BD-3 domain-type bridge
set logical-systems CE-4 bridge-domains BD-3 vlan-id 3
set logical-systems CE-4 bridge-domains BD-4 domain-type bridge
set logical-systems CE-4 bridge-domains BD-4 vlan-id 4
set logical-systems CE-4 bridge-domains BD-5 domain-type bridge
set logical-systems CE-4 bridge-domains BD-5 vlan-id 5
set chassis aggregated-devices ethernet device-count 1
set interfaces traceoptions file R6-DCD.log
set interfaces traceoptions file size 10m
set interfaces traceoptions flag all
set interfaces xe-0/0/0 unit 0 description "CONNECTED TO MX22"
set interfaces xe-0/0/0 unit 0 family inet address 192.168.10.2/24
set interfaces ge-1/0/0 description "CONNECTED TO CE-5"
set interfaces ge-1/0/0 gigether-options 802.3ad ae0
set interfaces ge-1/0/1 unit 0 description "CONNECTED TO MX21"
set interfaces ge-1/0/1 unit 0 family inet address 192.168.101.1/24
set interfaces ge-1/0/6 unit 0 description "CONNECTED TO CE-4"
set interfaces ge-1/0/6 unit 0 family bridge interface-mode trunk
set interfaces ge-1/0/6 unit 0 family bridge vlan-id-list 1-5
set interfaces ge-1/1/3 unit 0 description "CONNECTED TO ToR22"
set interfaces ge-1/1/3 unit 0 family inet address 192.168.12.1/24
set interfaces ae0 esi 00:44:44:44:44:44:44:44:44
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp periodic fast
set interfaces ae0 aggregated-ether-options lacp system-id 22:22:22:22:22:22
set interfaces ae0 unit 0 family bridge interface-mode trunk
set interfaces ae0 unit 0 family bridge vlan-id-list 1-5
set interfaces lo0 unit 90 family inet address 198.51.100.11/32
set routing-options router-id 198.51.100.11
set routing-options autonomous-system 600
set routing-options forwarding-table export evpn-pplb
set protocols bgp export TEST
set protocols bgp local-as 600
set protocols bgp group MX21 type external

```

```
set protocols bgp group MX21 local-address 192.168.9.2
set protocols bgp group MX21 export L0
set protocols bgp group MX21 export TEST
set protocols bgp group MX21 peer-as 800
set protocols bgp group MX21 local-as 600
set protocols bgp group MX21 neighbor 192.168.9.1 family inet unicast
set protocols bgp group MX22 type external
set protocols bgp group MX22 local-address 10.102.2.1
set protocols bgp group MX22 export L0
set protocols bgp group MX22 export TEST
set protocols bgp group MX22 peer-as 900
set protocols bgp group MX22 local-as 600
set protocols bgp group MX22 neighbor 192.168.10.1 family inet unicast
set protocols bgp group ToR22 type external
set protocols bgp group ToR22 local-address 10.105.5.1
set protocols bgp group ToR22 export L0
set protocols bgp group ToR22 export TEST
set protocols bgp group ToR22 peer-as 700
set protocols bgp group ToR22 local-as 600
set protocols bgp group ToR22 neighbor 192.168.12.2 family inet unicast
set protocols bgp group MX21-EVPN type external
set protocols bgp group MX21-EVPN multihop ttl 2
set protocols bgp group MX21-EVPN multihop no-nexthop-change
set protocols bgp group MX21-EVPN local-address 198.51.100.11
set protocols bgp group MX21-EVPN peer-as 800
set protocols bgp group MX21-EVPN local-as 600
set protocols bgp group MX21-EVPN neighbor 198.51.100.21 family evpn signaling
set protocols bgp group MX22-EVPN type external
set protocols bgp group MX22-EVPN multihop ttl 2
set protocols bgp group MX22-EVPN multihop no-nexthop-change
set protocols bgp group MX22-EVPN local-address 198.51.100.11
set protocols bgp group MX22-EVPN peer-as 900
set protocols bgp group MX22-EVPN local-as 600
set protocols bgp group MX22-EVPN neighbor 198.51.100.22 family evpn signaling
set protocols bgp group ToR22-EVPN type external
set protocols bgp group ToR22-EVPN multihop ttl 2
set protocols bgp group ToR22-EVPN multihop no-nexthop-change
set protocols bgp group ToR22-EVPN local-address 198.51.100.11
set protocols bgp group ToR22-EVPN peer-as 700
set protocols bgp group ToR22-EVPN local-as 600
set protocols bgp group ToR22-EVPN neighbor 198.51.100.12 family evpn signaling
set protocols l2-learning traceoptions file TOR21-L2ALD.log
set protocols l2-learning traceoptions file size 10m
```

```

set protocols l2-learning traceoptions level all
set protocols l2-learning traceoptions flag all
set policy-options policy-statement L0 term 1 from protocol direct
set policy-options policy-statement L0 term 1 from route-filter 198.51.100.11/32 exact
set policy-options policy-statement L0 term 1 then accept
set policy-options policy-statement TEST then community add NO-EXPORT
set policy-options policy-statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pplb then load-balance per-packet
set policy-options community NO-EXPORT members no-advertise
set policy-options community NO-EXPORT members no-export
set policy-options community NO-EXPORT members no-export-subconfed
set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.90
set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
set routing-instances EVPN-VXLAN-1 interface ge-1/0/6.0
set routing-instances EVPN-VXLAN-1 interface ae0.0
set routing-instances EVPN-VXLAN-1 route-distinguisher 198.51.100.11:1
set routing-instances EVPN-VXLAN-1 vrf-target target:1:3
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file TOR21-EVPN-VXLAN-1.log
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file size 10m
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions flag all
set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation vxlan
set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list 1-5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id 5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni 5

```

ToR-22

```

set system host-name ToR22

```

```

set logical-systems CE-5 interfaces ge-1/0/9 unit 0 description "CONNECTED TO Host 5"
set logical-systems CE-5 interfaces ge-1/0/9 unit 0 family bridge interface-mode trunk
set logical-systems CE-5 interfaces ge-1/0/9 unit 0 family bridge vlan-id-list 1-5
set logical-systems CE-5 interfaces ae1 unit 0 description "CONNECTED TO ToR21"
set logical-systems CE-5 interfaces ae1 unit 0 family bridge interface-mode trunk
set logical-systems CE-5 interfaces ae1 unit 0 family bridge vlan-id-list 1-5
set logical-systems CE-5 bridge-domains BD-1 domain-type bridge
set logical-systems CE-5 bridge-domains BD-1 vlan-id 1
set logical-systems CE-5 bridge-domains BD-2 domain-type bridge
set logical-systems CE-5 bridge-domains BD-2 vlan-id 2
set logical-systems CE-5 bridge-domains BD-3 domain-type bridge
set logical-systems CE-5 bridge-domains BD-3 vlan-id 3
set logical-systems CE-5 bridge-domains BD-4 domain-type bridge
set logical-systems CE-5 bridge-domains BD-4 vlan-id 4
set logical-systems CE-5 bridge-domains BD-5 domain-type bridge
set logical-systems CE-5 bridge-domains BD-5 vlan-id 5
set logical-systems CE-6 interfaces ge-1/1/6 unit 0 description "CONNECTED TO ToR22"
set logical-systems CE-6 interfaces ge-1/1/6 unit 0 family bridge interface-mode trunk
set logical-systems CE-6 interfaces ge-1/1/6 unit 0 family bridge vlan-id-list 1-5
set logical-systems CE-6 interfaces ge-1/1/9 unit 0 description "CONNECTED TO Host 6"
set logical-systems CE-6 interfaces ge-1/1/9 unit 0 family bridge interface-mode trunk
set logical-systems CE-6 interfaces ge-1/1/9 unit 0 family bridge vlan-id-list 1-5
set logical-systems CE-6 bridge-domains BD-1 domain-type bridge
set logical-systems CE-6 bridge-domains BD-1 vlan-id 1
set logical-systems CE-6 bridge-domains BD-2 domain-type bridge
set logical-systems CE-6 bridge-domains BD-2 vlan-id 2
set logical-systems CE-6 bridge-domains BD-3 domain-type bridge
set logical-systems CE-6 bridge-domains BD-3 vlan-id 3
set logical-systems CE-6 bridge-domains BD-4 domain-type bridge
set logical-systems CE-6 bridge-domains BD-4 vlan-id 4
set logical-systems CE-6 bridge-domains BD-5 domain-type bridge
set logical-systems CE-6 bridge-domains BD-5 vlan-id 5
set chassis aggregated-devices ethernet device-count 2
set interfaces traceoptions file R7-DCD.log
set interfaces traceoptions file size 10m
set interfaces traceoptions flag all
set interfaces xe-0/0/0 unit 0 description "CONNECTED TO MX22"
set interfaces xe-0/0/0 unit 0 family inet address 192.168.11.2/24
set interfaces ge-1/0/0 description "CONNECTED TO ToR21"
set interfaces ge-1/0/0 gigether-options 802.3ad ae1
set interfaces ge-1/0/6 unit 0 description "CONNECTED TO CE-6"
set interfaces ge-1/0/6 unit 0 family bridge interface-mode trunk
set interfaces ge-1/0/6 unit 0 family bridge vlan-id-list 1-5

```

```

set interfaces ge-1/0/7 description "CONNECTED TO ToR22"
set interfaces ge-1/0/7 gigether-options 802.3ad ae1
set interfaces ge-1/1/0 unit 0 description "CONNECTED TO MX21"
set interfaces ge-1/1/0 unit 0 family inet address 192.168.8.2/24
set interfaces ge-1/1/3 unit 0 description "CONNECTED TO ToR21"
set interfaces ge-1/1/3 unit 0 family inet address 192.168.12.2/24
set interfaces ge-1/1/7 description "CONNECTED TO CE-5"
set interfaces ge-1/1/7 gigether-options 802.3ad ae0
set interfaces ae0 esi 00:44:44:44:44:44:44:44:44
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp periodic fast
set interfaces ae0 aggregated-ether-options lacp system-id 22:22:22:22:22:22
set interfaces ae0 unit 0 family bridge interface-mode trunk
set interfaces ae0 unit 0 family bridge vlan-id-list 1-5
set interfaces ae1 aggregated-ether-options lacp active
set interfaces ae1 aggregated-ether-options lacp periodic fast
set interfaces ae1 aggregated-ether-options lacp system-id 22:22:22:22:22:22
set interfaces lo0 unit 92 family inet address 198.51.100.12/32
set routing-options router-id 198.51.100.12
set routing-options autonomous-system 700
set routing-options forwarding-table export evpn-pplb
set protocols bgp export TEST
set protocols bgp local-as 700
set protocols bgp group MX21 type external
set protocols bgp group MX21 local-address 192.168.8.2
set protocols bgp group MX21 export L0
set protocols bgp group MX21 export TEST
set protocols bgp group MX21 peer-as 800
set protocols bgp group MX21 local-as 700
set protocols bgp group MX21 neighbor 192.168.8.1 family inet unicast
set protocols bgp group MX22 type external
set protocols bgp group MX22 local-address 192.168.11.2
set protocols bgp group MX22 export L0
set protocols bgp group MX22 export TEST
set protocols bgp group MX22 peer-as 900
set protocols bgp group MX22 local-as 700
set protocols bgp group MX22 neighbor 192.168.11.1 family inet unicast
set protocols bgp group ToR21 type external
set protocols bgp group ToR21 local-address 192.168.12.2
set protocols bgp group ToR21 export L0
set protocols bgp group ToR21 export TEST
set protocols bgp group ToR21 peer-as 600

```

```
set protocols bgp group ToR21 local-as 700
set protocols bgp group ToR21 neighbor 10.105.5.1 family inet unicast
set protocols bgp group MX21-EVPN type external
set protocols bgp group MX21-EVPN multihop ttl 2
set protocols bgp group MX21-EVPN multihop no-nexthop-change
set protocols bgp group MX21-EVPN local-address 198.51.100.12
set protocols bgp group MX21-EVPN peer-as 800
set protocols bgp group MX21-EVPN local-as 700
set protocols bgp group MX21-EVPN neighbor 198.51.100.21 family evpn signaling
set protocols bgp group MX22-EVPN type external
set protocols bgp group MX22-EVPN multihop ttl 2
set protocols bgp group MX22-EVPN multihop no-nexthop-change
set protocols bgp group MX22-EVPN local-address 198.51.100.12
set protocols bgp group MX22-EVPN peer-as 900
set protocols bgp group MX22-EVPN local-as 700
set protocols bgp group MX22-EVPN neighbor 198.51.100.22 family evpn signaling
set protocols bgp group ToR21-EVPN type external
set protocols bgp group ToR21-EVPN multihop ttl 2
set protocols bgp group ToR21-EVPN multihop no-nexthop-change
set protocols bgp group ToR21-EVPN local-address 198.51.100.12
set protocols bgp group ToR21-EVPN peer-as 600
set protocols bgp group ToR21-EVPN local-as 700
set protocols bgp group ToR21-EVPN neighbor 198.51.100.11 family evpn signaling
set protocols l2-learning traceoptions file TOR22-L2ALD.log
set protocols l2-learning traceoptions file size 10m
set protocols l2-learning traceoptions level all
set protocols l2-learning traceoptions flag all
set policy-options policy-statement L0 term 1 from protocol direct
set policy-options policy-statement L0 term 1 from route-filter 198.51.100.12/32 exact
set policy-options policy-statement L0 term 1 then accept
set policy-options policy-statement TEST then community add NO-EXPORT
set policy-options policy-statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pplb then load-balance per-packet
set policy-options community NO-EXPORT members no-advertise
set policy-options community NO-EXPORT members no-export
set policy-options community NO-EXPORT members no-export-subconfed
set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.92
set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
set routing-instances EVPN-VXLAN-1 interface ge-1/0/6.0
set routing-instances EVPN-VXLAN-1 interface ae0.0
set routing-instances EVPN-VXLAN-1 route-distinguisher 198.51.100.12:1
set routing-instances EVPN-VXLAN-1 vrf-target target:1:3
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file TOR22-EVPN-VXLAN-1.log
```

```

set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file size 10m
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions flag all
set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation vxlan
set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list 1-5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id 5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni 5

```

Configuring ToR11

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

Configure the MX router as ToR11:

1. Set the system hostname.

```

[edit]
user@ToR11# set system host-name ToR11

```

2. Configure the interfaces and bridge domains on the CE2 device to enable Layer 2 connectivity.

```

[edit]

user@ce2# set logical-systems CE-2 interfaces ge-1/0/9 unit 0 description "CONNECTED TO
Host 2"

```

```

user@ce2# set logical-systems CE-2 interfaces ge-1/0/9 unit 0 family bridge interface-mode
trunk
user@ce2# set logical-systems CE-2 interfaces ge-1/0/9 unit 0 family bridge vlan-id-list 1-5
user@ce2# set logical-systems CE-2 interfaces ge-1/1/6 unit 0 description "CONNECTED TO
ToR11"
user@ce2# set logical-systems CE-2 interfaces ge-1/1/6 unit 0 family bridge interface-mode
trunk
user@ce2# set logical-systems CE-2 interfaces ge-1/1/6 unit 0 family bridge vlan-id-list 1-5
user@ce2# set logical-systems CE-2 bridge-domains BD-1 domain-type bridge
user@ce2# set logical-systems CE-2 bridge-domains BD-1 vlan-id 1
user@ce2# set logical-systems CE-2 bridge-domains BD-2 domain-type bridge
user@ce2# set logical-systems CE-2 bridge-domains BD-2 vlan-id 2
user@ce2# set logical-systems CE-2 bridge-domains BD-3 domain-type bridge
user@ce2# set logical-systems CE-2 bridge-domains BD-3 vlan-id 3
user@ce2# set logical-systems CE-2 bridge-domains BD-4 domain-type bridge
user@ce2# set logical-systems CE-2 bridge-domains BD-4 vlan-id 4
user@ce2# set logical-systems CE-2 bridge-domains BD-5 domain-type bridge
user@ce2# set logical-systems CE-2 bridge-domains BD-5 vlan-id 5

```

3. Configure trace options for the interfaces to enable trace logs.

```

[edit]

user@ce2# set interfaces traceoptions file R0-DCD.log
user@ce2# set interfaces traceoptions file size 10m
user@ce2# set interfaces traceoptions flag all

```

4. Set the number of aggregated Ethernet interfaces.

```

[edit]

user@ToR11# set chassis aggregated-devices ethernet device-count 1

```

5. Configure the interfaces on the ToR11 device to connect to the MX12, CE-2, CE-1, ToR12, and MX11 devices to enable underlay connectivity.

```

[edit]

user@ToR11# set interfaces ge-1/0/5 unit 0 description "CONNECTED TO MX12"
user@ToR11# set interfaces ge-1/0/5 unit 0 family inet address 192.168.4.1/24

```



```

user@ToR11# set interfaces ge-1/0/6 unit 0 description "CONNECTED TO CE-2"
user@ToR11# set interfaces ge-1/0/6 unit 0 family bridge interface-mode trunk
user@ToR11# set interfaces ge-1/0/6 unit 0 family bridge vlan-id-list 1-5
user@ToR11# set interfaces ge-1/1/0 description "CONNECTED TO CE-1"
user@ToR11# set interfaces ge-1/1/0 gigether-options 802.3ad ae0
user@ToR11# set interfaces ge-1/1/1 unit 0 description "CONNECTED TO MX11"
user@ToR11# set interfaces ge-1/1/1 unit 0 family inet address 192.168.3.1/24
user@ToR11# set interfaces ge-1/1/3 unit 0 description "CONNECTED TO ToR12"
user@ToR11# set interfaces ge-1/1/3 unit 0 family inet address 192.168.2.1/24

```

6. Configure a Link Aggregation Control Protocol (LACP)-enabled link aggregation group (LAG) interface towards the CE-1 end host device. The ESI value is globally unique across the entire EVPN domain. The all-active configuration enables ToR11 and ToR12 to forward traffic to, and from the CE devices, such that all CE links are actively used.

```

[edit]

user@ToR11# set interfaces ae0 esi 00:11:11:11:11:11:11:11
user@ToR11# set interfaces ae0 esi all-active
user@ToR11# set interfaces ae0 aggregated-ether-options lacp active
user@ToR11# set interfaces ae0 aggregated-ether-options lacp periodic fast
user@ToR11# set interfaces ae0 aggregated-ether-options lacp system-id 11:11:11:11:11:11
user@ToR11# set interfaces ae0 unit 0 family bridge interface-mode trunk
user@ToR11# set interfaces ae0 unit 0 family bridge vlan-id-list 1-5

```

7. Configure the loopback interface address and routing options.

```

[edit]

user@ToR11# set interfaces lo0 unit 81 family inet address 192.0.2.11/32
user@ToR11# set routing-options router-id 192.0.2.11
user@ToR11# set routing-options autonomous-system 100

```

8. Configure load balancing on ToR11.

```

[edit]

user@ToR11# set routing-options forwarding-table export evpn-pplb

```

9. Configure a multiprotocol external BGP (MP-EBGP) underlay connectivity between the ToR (ToR11 and ToR12) and gateway routers (MX11 and MX12).

[edit]

```

user@ToR11# set protocols bgp local-as 100
user@ToR11# set protocols bgp group MX11 type external
user@ToR11# set protocols bgp group MX11 local-address 192.168.3.1
user@ToR11# set protocols bgp group MX11 export L0
user@ToR11# set protocols bgp group MX11 export TEST
user@ToR11# set protocols bgp group MX11 peer-as 400
user@ToR11# set protocols bgp group MX11 neighbor 192.168.3.2 family inet unicast
user@ToR11# set protocols bgp group MX12 type external
user@ToR11# set protocols bgp group MX12 local-address 192.168.4.1
user@ToR11# set protocols bgp group MX12 export L0
user@ToR11# set protocols bgp group MX12 export TEST
user@ToR11# set protocols bgp group MX12 peer-as 500
user@ToR11# set protocols bgp group MX12 neighbor 192.168.4.2 family inet unicast
user@ToR11# set protocols bgp group ToR12 type external
user@ToR11# set protocols bgp group ToR12 local-address 192.168.2.1
user@ToR11# set protocols bgp group ToR12 export L0
user@ToR11# set protocols bgp group ToR12 export TEST
user@ToR11# set protocols bgp group ToR12 peer-as 200
user@ToR11# set protocols bgp group ToR12 local-as 100
user@ToR11# set protocols bgp group ToR12 neighbor 192.168.2.2 family inet unicast

```

10. Configure a multiprotocol external BGP (MP-EBGP) overlay between the ToR (ToR11 and ToR12) and gateway routers (MX11 and MX12) and set EVPN as the signaling protocol.

Step-by-Step Procedure

- a. Configure a MP-EBGP overlay to connect between ToR11 and MX11 using EVPN signaling.

[edit]

```

user@ToR11# set protocols bgp group MX11-EVPN type external
user@ToR11# set protocols bgp group MX11-EVPN multihop ttl 2
user@ToR11# set protocols bgp group MX11-EVPN multihop no-nexthop-change
user@ToR11# set protocols bgp group MX11-EVPN local-address 192.0.2.11
user@ToR11# set protocols bgp group MX11-EVPN export TEST

```

```

user@ToR11# set protocols bgp group MX11-EVPN peer-as 400
user@ToR11# set protocols bgp group MX11-EVPN local-as 100
user@ToR11# set protocols bgp group MX11-EVPN neighbor 192.0.2.21 family evpn signaling

```

- b.** Configure a MP-EBGP overlay to connect between ToR11 and MX12 using EVPN signaling.

```

[edit]

user@ToR11# set protocols bgp group MX12-EVPN type external
user@ToR11# set protocols bgp group MX12-EVPN multihop ttl 2
user@ToR11# set protocols bgp group MX12-EVPN multihop no-nexthop-change
user@ToR11# set protocols bgp group MX12-EVPN local-address 192.0.2.11
user@ToR11# set protocols bgp group MX12-EVPN export TEST
user@ToR11# set protocols bgp group MX12-EVPN peer-as 500
user@ToR11# set protocols bgp group MX12-EVPN local-as 100
user@ToR11# set protocols bgp group MX12-EVPN neighbor 192.0.2.22 family evpn signaling

```

- c.** Configure a MP-EBGP overlay to connect between ToR11 and ToR12 using EVPN signaling.

```

[edit]

user@ToR11# set protocols bgp group ToR12-EVPN type external
user@ToR11# set protocols bgp group ToR12-EVPN multihop ttl 2
user@ToR11# set protocols bgp group ToR12-EVPN multihop no-nexthop-change
user@ToR11# set protocols bgp group ToR12-EVPN local-address 192.0.2.11
user@ToR11# set protocols bgp group ToR12-EVPN export TEST
user@ToR11# set protocols bgp group ToR12-EVPN peer-as 200
user@ToR11# set protocols bgp group ToR12-EVPN local-as 100
user@ToR11# set protocols bgp group ToR12-EVPN neighbor 192.0.2.12 family evpn signaling

```

- 11.** Configure trace operations to track all Layer 2 address learning and forwarding properties.

```

[edit]

user@ToR11# set protocols l2-learning traceoptions file TOR11-L2ALD.log
user@ToR11# set protocols l2-learning traceoptions file size 10m
user@ToR11# set protocols l2-learning traceoptions level all
user@ToR11# set protocols l2-learning traceoptions flag all

```

12. Configure routing policies to accept the direct loopback address route and redirect it into BGP.

```
[edit]

user@ToR11# set policy-options policy-statement L0 term 1 from protocol direct
user@ToR11# set policy-options policy-statement L0 term 1 from route-filter 192.0.2.11/32
exact
user@ToR11# set policy-options policy-statement L0 term 1 then accept
```

13. Configure community policy options.

```
[edit]

user@ToR11# set policy-options community NO-EXPORT members no-advertise
user@ToR11# set policy-options community NO-EXPORT members no-export
user@ToR11# set policy-options community NO-EXPORT members no-export-subconfed
```

14. Apply load balance.

```
[edit]

user@ToR11# set policy-options policy-statement TEST then community add NO-EXPORT
user@ToR11# set policy-options policy-statement evpn-pplb from protocol evpn
user@ToR11# set policy-options policy-statement evpn-pplb then load-balance per-packet
```

15. Configure EVPN routing instances for each virtual network. Define the VTEP source interface, route distinguisher (used to identify and advertise EVPN routes), and vrf-target (exports and tags all routes for that local VRF using the defined route target). Configure the EVPN protocol, encapsulation method, VNI list, and BUM traffic forwarding method. Finally, configure a bridge domain for each virtual router that maps VNIDs to VLAN IDs, and identify the BUM forwarding method.

```
[edit]

user@ToR11# set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.81
user@ToR11# set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
user@ToR11# set routing-instances EVPN-VXLAN-1 interface ge-1/0/6.0
user@ToR11# set routing-instances EVPN-VXLAN-1 interface ae0.0
user@ToR11# set routing-instances EVPN-VXLAN-1 route-distinguisher 192.0.2.11:1
```

```

user@ToR11# set routing-instances EVPN-VXLAN-1 vrf-target target:1:1
user@ToR11# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file TOR11-EVPN-
VXLAN-1.log
user@ToR11# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file size 10m
user@ToR11# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions flag all
user@ToR11# set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation vxlan
user@ToR11# set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list 1-5
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type bridge
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id 1
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni 1
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type bridge
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id 2
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni 2
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type bridge
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id 3
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni 3
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type bridge
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id 4
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni 4
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type bridge
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id 5
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni 5

```

Configuring ToR12

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

Configure the MX router ToR12:

1. Set the system hostname.

```

[edit]
user@ToR12# set system host-name ToR12

```

2. Configure the interfaces and bridge domains on the CE-1 device to enable Layer 2 connectivity.

[edit]

```
user@ce1# set logical-systems CE-1 interfaces ge-1/0/9 unit 0 description "CONNECTED TO
Host 1"
user@ce1# set logical-systems CE-1 interfaces ge-1/0/9 unit 0 family bridge interface-mode
trunk
user@ce1# set logical-systems CE-1 interfaces ge-1/0/9 unit 0 family bridge vlan-id-list 1-5
user@ce1# set logical-systems CE-1 interfaces ae1 unit 0 description "CONNECTED TO ToR12"
user@ce1# set logical-systems CE-1 interfaces ae1 unit 0 family bridge interface-mode trunk
user@ce1# set logical-systems CE-1 interfaces ae1 unit 0 family bridge vlan-id-list 1-5
user@ce1# set logical-systems CE-1 bridge-domains BD-1 domain-type bridge
user@ce1# set logical-systems CE-1 bridge-domains BD-1 vlan-id 1
user@ce1# set logical-systems CE-1 bridge-domains BD-2 domain-type bridge
user@ce1# set logical-systems CE-1 bridge-domains BD-2 vlan-id 2
user@ce1# set logical-systems CE-1 bridge-domains BD-3 domain-type bridge
user@ce1# set logical-systems CE-1 bridge-domains BD-3 vlan-id 3
user@ce1# set logical-systems CE-1 bridge-domains BD-4 domain-type bridge
user@ce1# set logical-systems CE-1 bridge-domains BD-4 vlan-id 4
user@ce1# set logical-systems CE-1 bridge-domains BD-5 domain-type bridge
user@ce1# set logical-systems CE-1 bridge-domains BD-5 vlan-id 5
```

3. Configure the interfaces and bridge domains on the CE-3 device to enable Layer 2 connectivity.

[edit]

```
user@ce3# set logical-systems CE-3 interfaces ge-1/1/7 unit 0 description "CONNECTED TO
ToR12"
user@ce3# set logical-systems CE-3 interfaces ge-1/1/7 unit 0 family bridge interface-mode
trunk
user@ce3# set logical-systems CE-3 interfaces ge-1/1/7 unit 0 family bridge vlan-id-list 1-5
user@ce3# set logical-systems CE-3 interfaces ge-1/1/9 unit 0 description "CONNECTED TO
Host 3"
user@ce3# set logical-systems CE-3 interfaces ge-1/1/9 unit 0 family bridge interface-mode
trunk
user@ce3# set logical-systems CE-3 interfaces ge-1/1/9 unit 0 family bridge vlan-id-list 1-5
user@ce3# set logical-systems CE-3 bridge-domains BD-1 domain-type bridge
user@ce3# set logical-systems CE-3 bridge-domains BD-1 vlan-id 1
user@ce3# set logical-systems CE-3 bridge-domains BD-2 domain-type bridge
user@ce3# set logical-systems CE-3 bridge-domains BD-2 vlan-id 2
```

```

user@ce3# set logical-systems CE-3 bridge-domains BD-3 domain-type bridge
user@ce3# set logical-systems CE-3 bridge-domains BD-3 vlan-id 3
user@ce3# set logical-systems CE-3 bridge-domains BD-4 domain-type bridge
user@ce3# set logical-systems CE-3 bridge-domains BD-4 vlan-id 4
user@ce3# set logical-systems CE-3 bridge-domains BD-5 domain-type bridge
user@ce3# set logical-systems CE-3 bridge-domains BD-5 vlan-id 5

```

4. Configure trace options for the interfaces to enable trace logs.

```

[edit]

user@ce3# set interfaces traceoptions file R1-DCD.log
user@ce3# set interfaces traceoptions file size 10m
user@ce3# set interfaces traceoptions flag all

```

5. Set the number of aggregated Ethernet interfaces.

```

[edit]

user@ToR12# set chassis aggregated-devices ethernet device-count 2

```

6. Configure the interfaces on the ToR12 device to connect to the MX12, CE-2, CE-3, ToR11, and MX11 devices to enable underlay connectivity.

```

[edit]

user@ToR12# set interfaces ge-1/0/0 unit 0 description "CONNECTED TO MX11"
user@ToR12# set interfaces ge-1/0/0 unit 0 family inet address 192.168.6.1/24
user@ToR12# set interfaces ge-1/0/4 unit 0 description "CONNECTED TO MX12"
user@ToR12# set interfaces ge-1/0/4 unit 0 family inet address 192.168.5.1/24
user@ToR12# set interfaces ge-1/0/6 description "CONNECTED TO CE-1"
user@ToR12# set interfaces ge-1/0/6 gigether-options 802.3ad ae0
user@ToR12# set interfaces ge-1/0/7 unit 0 description "CONNECTED TO CE-3"
user@ToR12# set interfaces ge-1/0/7 unit 0 family bridge interface-mode trunk
user@ToR12# set interfaces ge-1/0/7 unit 0 family bridge vlan-id-list 1-5
user@ToR12# set interfaces ge-1/1/0 description "CONNECTED TO ToR11"
user@ToR12# set interfaces ge-1/1/0 gigether-options 802.3ad ae1
user@ToR12# set interfaces ge-1/1/3 unit 0 description "CONNECTED TO ToR11"
user@ToR12# set interfaces ge-1/1/3 unit 0 family inet address 192.168.2.2/24

```

```

user@ToR12# set interfaces ge-1/1/6 description "CONNECTED TO ToR12"
user@ToR12# set interfaces ge-1/1/6 gigether-options 802.3ad ae1

```

7. Configure a Link Aggregation Control Protocol (LACP)-enabled link aggregation group (LAG) interface towards the CE-1 end host device. The ESI value is globally unique across the entire EVPN domain. The all-active configuration enables ToR11 and ToR12 to forward traffic to, and from the CE devices, such that all CE links are actively used.

```

[edit]

user@ToR12# set interfaces ae0 esi 00:11:11:11:11:11:11:11:11
user@ToR12# set interfaces ae0 esi all-active
user@ToR12# set interfaces ae0 aggregated-ether-options lacp system-id 11:11:11:11:11:11
user@ToR12# set interfaces ae0 unit 0 family bridge interface-mode trunk
user@ToR12# set interfaces ae0 unit 0 family bridge vlan-id-list 1-5
user@ToR12# set interfaces ae1 aggregated-ether-options lacp active
user@ToR12# set interfaces ae1 aggregated-ether-options lacp periodic fast

```

8. Configure the loopback interface address and routing options.

```

[edit]

user@ToR12# set interfaces lo0 unit 82 family inet address 192.0.2.12/32
user@ToR12# set routing-options router-id 192.0.2.12
user@ToR12# set routing-options autonomous-system 200

```

9. Configure load balancing on ToR12.

```

[edit]

user@ToR12# set routing-options forwarding-table export evpn-pplb

```

10. Configure a multiprotocol external BGP (MP-EBGP) underlay connectivity between the ToR (ToR12 and ToR11) and gateway routers (MX11 and MX12).

```

[edit]

user@ToR12# set protocols bgp local-as 200
user@ToR12# set protocols bgp group MX11 type external

```



```

user@ToR12# set protocols bgp group MX11 local-address 192.168.6.1
user@ToR12# set protocols bgp group MX11 export L0
user@ToR12# set protocols bgp group MX11 export TEST
user@ToR12# set protocols bgp group MX11 peer-as 400
user@ToR12# set protocols bgp group MX11 local-as 200
user@ToR12# set protocols bgp group MX11 neighbor 192.168.6.2 family inet unicast
user@ToR12# set protocols bgp group MX12 type external
user@ToR12# set protocols bgp group MX12 local-address 192.168.5.1
user@ToR12# set protocols bgp group MX12 export L0
user@ToR12# set protocols bgp group MX12 export TEST
user@ToR12# set protocols bgp group MX12 peer-as 500
user@ToR12# set protocols bgp group MX12 local-as 200
user@ToR12# set protocols bgp group MX12 neighbor 192.168.5.2 family inet unicast
user@ToR12# set protocols bgp group ToR11 type external
user@ToR12# set protocols bgp group ToR11 local-address 192.168.2.2
user@ToR12# set protocols bgp group ToR11 export L0
user@ToR12# set protocols bgp group ToR11 export TEST
user@ToR12# set protocols bgp group ToR11 peer-as 100
user@ToR12# set protocols bgp group ToR11 local-as 200
user@ToR12# set protocols bgp group ToR11 neighbor 192.168.2.1 family inet unicast

```

11. Configure a multiprotocol external BGP (MP-EBGP) overlay between the ToR (ToR12 and ToR11) and gateway routers (MX11 and MX12) and set EVPN as the signaling protocol.

Step-by-Step Procedure

- a. Configure a MP-EBGP overlay to connect between ToR12 and MX11 using EVPN signaling.

[edit]

```

user@ToR12# set protocols bgp group MX11-EVPN type external
user@ToR12# set protocols bgp group MX11-EVPN multihop ttl 2
user@ToR12# set protocols bgp group MX11-EVPN multihop no-nexthop-change
user@ToR12# set protocols bgp group MX11-EVPN local-address 192.0.2.12
user@ToR12# set protocols bgp group MX11-EVPN export TEST
user@ToR12# set protocols bgp group MX11-EVPN peer-as 400
user@ToR12# set protocols bgp group MX11-EVPN local-as 200
user@ToR12# set protocols bgp group MX11-EVPN neighbor 192.0.2.21 family evpn signaling

```

- b. Configure a MP-EBGP overlay to connect between ToR12 and MX12 using EVPN signaling.

```
[edit]

user@ToR12# set protocols bgp group MX12-EVPN type external
user@ToR12# set protocols bgp group MX12-EVPN multihop ttl 2
user@ToR12# set protocols bgp group MX12-EVPN multihop no-nexthop-change
user@ToR12# set protocols bgp group MX12-EVPN local-address 192.0.2.12
user@ToR12# set protocols bgp group MX12-EVPN export TEST
user@ToR12# set protocols bgp group MX12-EVPN peer-as 500
user@ToR12# set protocols bgp group MX12-EVPN local-as 200
user@ToR12# set protocols bgp group MX12-EVPN neighbor 192.0.2.22 family evpn signaling
```

- c. Configure a MP-EBGP overlay to connect between ToR12 and ToR11 using EVPN signaling.

```
[edit]

user@ToR12# set protocols bgp group ToR11-EVPN type external
user@ToR12# set protocols bgp group ToR11-EVPN multihop ttl 2
user@ToR12# set protocols bgp group ToR11-EVPN multihop no-nexthop-change
user@ToR12# set protocols bgp group ToR11-EVPN local-address 192.0.2.12
user@ToR12# set protocols bgp group ToR11-EVPN export TEST
user@ToR12# set protocols bgp group ToR11-EVPN peer-as 100
user@ToR12# set protocols bgp group ToR11-EVPN local-as 200
user@ToR12# set protocols bgp group ToR11-EVPN neighbor 192.0.2.11 family evpn signaling
user@ToR12# set protocols bgp group ToR12-EVPN export TEST
```

12. Configure trace operations to track all Layer 2 address learning and forwarding properties.

```
[edit]

user@ToR12# set protocols l2-learning traceoptions file TOR12-L2ALD.log
user@ToR12# set protocols l2-learning traceoptions file size 10m
user@ToR12# set protocols l2-learning traceoptions level all
user@ToR12# set protocols l2-learning traceoptions flag all
```

13. Configure routing policies to accept the direct loopback address route and redirect it into BGP.

```
[edit]

user@ToR12# set policy-options policy-statement L0 term 1 from protocol direct
user@ToR12# set policy-options policy-statement L0 term 1 from route-filter 192.0.2.12/32
exact
user@ToR12# set policy-options policy-statement L0 term 1 then accept
```

14. Configure community policy options.

```
[edit]

user@ToR12# set policy-options community NO-EXPORT members no-advertise
user@ToR12# set policy-options community NO-EXPORT members no-export
user@ToR12# set policy-options community NO-EXPORT members no-export-subconfed
```

15. Apply load balance.

```
[edit]

user@ToR12# set policy-options policy-statement TEST then community add NO-EXPORT
user@ToR12# set policy-options policy-statement evpn-pplb from protocol evpn
user@ToR12# set policy-options policy-statement evpn-pplb then load-balance per-packet
```

16. Configure EVPN routing instances for each virtual network. Define the VTEP source interface, route distinguisher (used to identify and advertise EVPN routes), and vrf-target (exports and tags all routes for that local VRF using the defined route target). Configure the EVPN protocol, encapsulation method, VNI list, and BUM traffic forwarding method. Finally, configure a bridge domain for each virtual router that maps VNIDs to VLAN IDs, and identify the BUM forwarding method.

```
[edit]

user@ToR12# set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.82
user@ToR12# set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
user@ToR12# set routing-instances EVPN-VXLAN-1 interface ge-1/0/7.0
user@ToR12# set routing-instances EVPN-VXLAN-1 interface ae0.0
user@ToR12# set routing-instances EVPN-VXLAN-1 route-distinguisher 192.0.2.12:1
```

```

user@ToR12# set routing-instances EVPN-VXLAN-1 vrf-target target:1:1
user@ToR12# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file TOR12-EVPN-
VXLAN-1.log
user@ToR12# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file size 10m
user@ToR12# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions flag all
user@ToR12# set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation vxlan
user@ToR12# set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list 1-5
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type bridge
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id 1
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni 1
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type bridge
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id 2
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni 2
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type bridge
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id 3
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni 3
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type bridge
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id 4
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni 4
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type bridge
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id 5
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni 5

```

Configuring Data Center Gateway and WAN Edge 1 Router (MX11)

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

Configure an MX Series router as the data center gateway and WAN edge router and name it as MX11:

1. Set the system hostname.

```

[edit]
user@MX11# set system host-name MX11

```

2. Configure the interfaces on the MX11 router (DC GW/WAN Edge1) to enable the underlay connectivity to the MX12, ToR11, ToR12, and P devices, which is the EVPN-VXLAN part of DC1 network.

[edit]

```
user@MX11# set interfaces ge-5/1/0 unit 0 description "CONNECTED TO MX12"
user@MX11# set interfaces ge-5/1/0 unit 0 family inet address 192.168.7.1/24
user@MX11# set interfaces ge-5/1/1 unit 0 description "CONNECTED TO ToR11"
user@MX11# set interfaces ge-5/1/1 unit 0 family inet address 192.168.3.2/24
user@MX11# set interfaces ge-5/1/8 unit 0 description "CONNECTED TO ToR12"
user@MX11# set interfaces ge-5/1/8 unit 0 family inet address 192.168.6.2/24
user@MX11# set interfaces ge-5/1/9 unit 0 description "CONNECTED TO P"
user@MX11# set interfaces ge-5/1/9 unit 0 family inet address 203.0.1.1/24
user@MX11# set interfaces ge-5/1/9 unit 0 family mpls
```

3. Configure a multiprotocol external BGP (MP-EBGP) underlay connectivity between the gateway routers (MX11 and MX12) and ToR (ToR11 and ToR12).

[edit]

```
user@MX11# set protocols bgp group ToR11 type external
user@MX11# set protocols bgp group ToR11 local-address 192.168.3.2
user@MX11# set protocols bgp group ToR11 import TEST
user@MX11# set protocols bgp group ToR11 export TEST
user@MX11# set protocols bgp group ToR11 export L0
user@MX11# set protocols bgp group ToR11 peer-as 100
user@MX11# set protocols bgp group ToR11 local-as 400
user@MX11# set protocols bgp group ToR11 neighbor 192.168.3.1 family inet unicast
user@MX11# set protocols bgp group ToR12 type external
user@MX11# set protocols bgp group ToR12 local-address 192.168.6.2
user@MX11# set protocols bgp group ToR12 export TEST
user@MX11# set protocols bgp group ToR12 export L0
user@MX11# set protocols bgp group ToR12 peer-as 200
user@MX11# set protocols bgp group ToR12 local-as 400
user@MX11# set protocols bgp group ToR12 neighbor 192.168.6.1 family inet unicast
user@MX11# set protocols bgp group MX12 type external
user@MX11# set protocols bgp group MX12 local-address 192.168.7.1
user@MX11# set protocols bgp group MX12 export TEST
user@MX11# set protocols bgp group MX12 export L0
user@MX11# set protocols bgp group MX12 peer-as 500
```

```

user@MX11# set protocols bgp group MX12 local-as 400
user@MX11# set protocols bgp group MX12 neighbor 192.168.7.2 family inet unicast

```

4. Configure a multiprotocol external BGP (MP-EBGP) overlay connectivity between the gateway routers (MX11 and MX12) and ToR (ToR11 and ToR12) and set EVPN as the signaling protocol.

```

[edit]

user@MX11# set protocols bgp group ToR11-EVPN type external
user@MX11# set protocols bgp group ToR11-EVPN multihop ttl 2
user@MX11# set protocols bgp group ToR11-EVPN multihop no-nexthop-change
user@MX11# set protocols bgp group ToR11-EVPN local-address 192.0.2.21
user@MX11# set protocols bgp group ToR11-EVPN export TEST
user@MX11# set protocols bgp group ToR11-EVPN peer-as 100
user@MX11# set protocols bgp group ToR11-EVPN local-as 400
user@MX11# set protocols bgp group ToR11-EVPN neighbor 192.0.2.11 family evpn signaling
user@MX11# set protocols bgp group ToR12-EVPN type external
user@MX11# set protocols bgp group ToR12-EVPN multihop ttl 2
user@MX11# set protocols bgp group ToR12-EVPN multihop no-nexthop-change
user@MX11# set protocols bgp group ToR12-EVPN local-address 192.0.2.21
user@MX11# set protocols bgp group ToR12-EVPN export TEST
user@MX11# set protocols bgp group ToR12-EVPN peer-as 200
user@MX11# set protocols bgp group ToR12-EVPN local-as 400
user@MX11# set protocols bgp group ToR12-EVPN neighbor 192.0.2.12 family evpn signaling
user@MX11# set protocols bgp group MX12-EVPN type external
user@MX11# set protocols bgp group MX12-EVPN multihop ttl 2
user@MX11# set protocols bgp group MX12-EVPN multihop no-nexthop-change
user@MX11# set protocols bgp group MX12-EVPN local-address 192.0.2.21
user@MX11# set protocols bgp group MX12-EVPN export TEST
user@MX11# set protocols bgp group MX12-EVPN peer-as 500
user@MX11# set protocols bgp group MX12-EVPN local-as 400
user@MX11# set protocols bgp group MX12-EVPN neighbor 192.0.2.22 family evpn signaling

```

5. Configure integrated routing and bridging (IRB) interfaces that advertises the MAC and IP routes (MAC+IP type 2 routes) for hosts in the topology. The IRB configuration is the gateway for the VLANs on the hosts.

Step-by-Step Procedure

- a. The following is the IRB gateway configuration for the VLAN-1 on Host (which is the host part of VLAN-1):

```
[edit]

user@MX11# set interfaces irb unit 1 proxy-macip-advertisement
user@MX11# set interfaces irb unit 1 virtual-gateway-esi 00:11:aa:aa:aa:aa:aa:aa:aa:aa
user@MX11# set interfaces irb unit 1 virtual-gateway-esi all-active
user@MX11# set interfaces irb unit 1 family inet address 10.11.1.12/24 virtual-gateway-
address 10.11.1.10
```

- b. The following is the IRB gateway configuration for the VLAN-2 on Host (which is the host part of VLAN-2):

```
[edit]

user@MX11# set interfaces irb unit 2 proxy-macip-advertisement
user@MX11# set interfaces irb unit 2 virtual-gateway-esi 00:11:bb:bb:bb:bb:bb:bb:bb:bb
user@MX11# set interfaces irb unit 2 virtual-gateway-esi all-active
user@MX11# set interfaces irb unit 2 family inet address 10.12.1.12/24 virtual-gateway-
address 10.12.1.10
```

- c. The following is the IRB gateway configuration for the VLAN-3 on Host (which is the host part of VLAN-3):

```
[edit]

user@MX11# set interfaces irb unit 3 proxy-macip-advertisement
user@MX11# set interfaces irb unit 3 virtual-gateway-esi 00:11:cc:cc:cc:cc:cc:cc:cc:cc
user@MX11# set interfaces irb unit 3 virtual-gateway-esi all-active
user@MX11# set interfaces irb unit 3 family inet address 10.13.1.12/24 virtual-gateway-
address 10.13.1.10
```

- d. The following is the IRB gateway configuration for the VLAN-4 on Host (which is the host part of VLAN-4):

```
[edit]

user@MX11# set interfaces irb unit 4 proxy-macip-advertisement
```

```

user@MX11# set interfaces irb unit 4 virtual-gateway-esi 00:11:dd:dd:dd:dd:dd:dd:dd
user@MX11# set interfaces irb unit 4 virtual-gateway-esi all-active
user@MX11# set interfaces irb unit 4 family inet address 10.14.1.12/24 virtual-gateway-
address 10.14.1.10

```

- e. The following is the IRB gateway configuration for the VLAN-5 on Host (which is the host part of VLAN-5):

```

[edit]

user@MX11# set interfaces irb unit 5 proxy-macip-advertisement
user@MX11# set interfaces irb unit 5 virtual-gateway-esi 00:11:ee:ee:ee:ee:ee:ee:ee
user@MX11# set interfaces irb unit 5 virtual-gateway-esi all-active
user@MX11# set interfaces irb unit 5 family inet address 10.15.1.12/24 virtual-gateway-
address 10.15.1.10

```

6. Configure trace operations to track all Layer 2 address learning and forwarding properties.

```

[edit]

user@MX11# set protocols l2-learning traceoptions file MX11-L2ALD.log
user@MX11# set protocols l2-learning traceoptions file size 10m
user@MX11# set protocols l2-learning traceoptions level all
user@MX11# set protocols l2-learning traceoptions flag all

```

7. Configure routing policies to accept the direct loopback address route and redirect it into BGP.

```

[edit]

user@MX11# set policy-options policy-statement L0 term 1 from protocol direct
user@MX11# set policy-options policy-statement L0 term 1 from route-filter 192.0.2.21/32
exact
user@MX11# set policy-options policy-statement L0 term 1 then accept
user@MX11# set policy-options policy-statement L0 from protocol direct
user@MX11# set policy-options policy-statement L0 from route-filter 192.0.2.21/32 exact
user@MX11# set policy-options policy-statement L0 then accept

```


8. Configure community policy options.

```
[edit]

user@MX11# set policy-options community NO-EXPORT members no-advertise
user@MX11# set policy-options community NO-EXPORT members no-export
user@MX11# set policy-options community NO-EXPORT members no-export-subconfed
```

9. Apply load balance.

```
[edit]

user@MX11# set policy-options policy-statement TEST then community add NO-EXPORT
user@MX11# set policy-options policy-statement evpn-pplb from protocol evpn
user@MX11# set policy-options policy-statement evpn-pplb then load-balance per-packet
```

10. Configure an ESI value on the logical tunnel interface. Use the same ESI value on all other gateway/WAN edge routers in the DC1 network.

```
[edit]

user@MX11# set interfaces lt-5/1/0 esi 00:22:22:22:22:22:22:22:22
```

11. Configure active-active multihoming on the logical tunnel interface by including the all-active statement.

```
[edit]

user@MX11# set interfaces lt-5/1/0 esi all-active
```

12. Configure a pair of logical tunnel (lt-) interface on the MX11 gateway router to interconnect the EVPN-VXLAN instance of the data center network with the MPLS-based EVPN instance of the WAN. One logical tunnel (lt-) interface is configured as the access interface for EVPN-VXLAN and the other logical tunnel (lt-) interface is configured as the access interface for MPLS-based EVPN.

```
[edit]

user@MX11# set interfaces lt-5/1/0 unit 0 peer-unit 1
```

```

user@MX11# set interfaces lt-5/1/0 unit 0 family bridge interface-mode trunk
user@MX11# set interfaces lt-5/1/0 unit 0 family bridge vlan-id-list 1-5
user@MX11# set interfaces lt-5/1/0 unit 1 peer-unit 0
user@MX11# set interfaces lt-5/1/0 unit 1 family bridge interface-mode trunk
user@MX11# set interfaces lt-5/1/0 unit 1 family bridge vlan-id-list 1-5

```

13. Configure the loopback interface address and routing options.

```

[edit]

user@MX11# set interfaces lo0 unit 84 family inet address 192.0.2.21/32
user@MX11# set interfaces lo0 unit 84 family mpls
user@MX11# set routing-options router-id 192.0.2.21
user@MX11# set routing-options autonomous-system 300

```

14. Configure load balancing on MX11.

```

[edit]

user@MX11# set routing-options forwarding-table export evpn-pplb

```

15. Enable RSVP, MPLS, BGP, and OSPF protocols on the core interfaces. Create MPLS LSPs and specify the address of the other gateway and WAN edge routers (MX12, P, MX21, MX22).

```

[edit]

user@MX11# set protocols rsvp interface all
user@MX11# set protocols rsvp interface fxp0.0 disable
user@MX11# set protocols mpls label-switched-path MX11-T0-MX12 to 192.0.2.22
user@MX11# set protocols mpls label-switched-path MX11-T0-P to 203.0.113.1
user@MX11# set protocols mpls label-switched-path MX11-T0-MX21 to 198.51.100.21
user@MX11# set protocols mpls label-switched-path MX11-T0-MX22 to 198.51.100.22
user@MX11# set protocols mpls interface all
user@MX11# set protocols mpls interface fxp0.0 disable
user@MX11# set protocols bgp local-address 192.0.2.21
user@MX11# set protocols bgp local-as 300
user@MX11# set protocols bgp group INT type internal
user@MX11# set protocols bgp group INT local-address 192.0.2.21
user@MX11# set protocols bgp group INT family evpn signaling
user@MX11# set protocols bgp group INT export TEST

```

```

user@MX11# set protocols bgp group INT neighbor 203.0.113.1
user@MX11# set protocols ospf traffic-engineering
user@MX11# set protocols ospf area 0.0.0.0 interface ge-5/1/9.0
user@MX11# set protocols ospf area 0.0.0.0 interface lo0.84 passive

```

16. Configure EVPN-based MPLS routing instances on the MX11 router for each virtual network. Define the route distinguisher (used to identify and advertise EVPN-MPLS routes) and vrf-target (exports and tags all routes for that local VRF using the defined route target). Configure a bridge domain for each virtual router that maps VLAN IDs.

```

[edit]

user@MX11# set routing-instances EVPN-MPLS-1 instance-type virtual-switch
user@MX11# set routing-instances EVPN-MPLS-1 interface lt-5/1/0.0
user@MX11# set routing-instances EVPN-MPLS-1 route-distinguisher 192.0.2.21:100
user@MX11# set routing-instances EVPN-MPLS-1 vrf-target target:1:2
user@MX11# set routing-instances EVPN-MPLS-1 protocols evpn traceoptions file MX11-EVPN-
MPLS-1.log
user@MX11# set routing-instances EVPN-MPLS-1 protocols evpn traceoptions file size 10m
user@MX11# set routing-instances EVPN-MPLS-1 protocols evpn traceoptions flag all
user@MX11# set routing-instances EVPN-MPLS-1 protocols evpn extended-vlan-list 1-5
user@MX11# set routing-instances EVPN-MPLS-1 protocols evpn default-gateway no-gateway-
community
user@MX11# set routing-instances EVPN-MPLS-1 bridge-domains BD-1 domain-type bridge
user@MX11# set routing-instances EVPN-MPLS-1 bridge-domains BD-1 vlan-id 1
user@MX11# set routing-instances EVPN-MPLS-1 bridge-domains BD-2 domain-type bridge
user@MX11# set routing-instances EVPN-MPLS-1 bridge-domains BD-2 vlan-id 2
user@MX11# set routing-instances EVPN-MPLS-1 bridge-domains BD-3 domain-type bridge
user@MX11# set routing-instances EVPN-MPLS-1 bridge-domains BD-3 vlan-id 3
user@MX11# set routing-instances EVPN-MPLS-1 bridge-domains BD-4 domain-type bridge
user@MX11# set routing-instances EVPN-MPLS-1 bridge-domains BD-4 vlan-id 4
user@MX11# set routing-instances EVPN-MPLS-1 bridge-domains BD-5 domain-type bridge
user@MX11# set routing-instances EVPN-MPLS-1 bridge-domains BD-5 vlan-id 5

```

17. Configure EVPN-VXLAN routing instances on the MX11 router for each virtual network. Define the VTEP source interface, route distinguisher (used to identify and advertise EVPN routes), and vrf-target (exports and tags all routes for that local VRF using the defined route target). Configure the EVPN protocol, encapsulation method, VNI list, and BUM traffic forwarding method. Finally,

configure a bridge domain for each virtual router that maps VNIDs to VLAN IDs, and identify the BUM forwarding method.

[edit]

```

user@MX11# set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.84
user@MX11# set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
user@MX11# set routing-instances EVPN-VXLAN-1 interface lt-5/1/0.1
user@MX11# set routing-instances EVPN-VXLAN-1 route-distinguisher 192.0.2.21:1
user@MX11# set routing-instances EVPN-VXLAN-1 vrf-target target:1:1
user@MX11# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file MX11-EVPN-
VXLAN-1.log
user@MX11# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file size 10m
user@MX11# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions flag all
user@MX11# set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation vxlan
user@MX11# set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list 1-5
user@MX11# set routing-instances EVPN-VXLAN-1 protocols evpn default-gateway no-gateway-
community
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type bridge
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id 1
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 routing-interface irb.1
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni 1
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type bridge
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id 2
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 routing-interface irb.2
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni 2
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type bridge
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id 3
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 routing-interface irb.3
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni 3
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type bridge
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id 4
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 routing-interface irb.4
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni 4
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type bridge
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id 5
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 routing-interface irb.5
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni 5
user@MX11# set routing-instances VRF instance-type vrf
user@MX11# set routing-instances VRF interface irb.1
user@MX11# set routing-instances VRF interface irb.2
user@MX11# set routing-instances VRF interface irb.3

```

```

user@MX11# set routing-instances VRF interface irb.4
user@MX11# set routing-instances VRF interface irb.5
user@MX11# set routing-instances VRF route-distinguisher 1:1
user@MX11# set routing-instances VRF vrf-target target:10:10

```

Configuring Data Center Gateway and WAN Edge 2 Router (MX12)

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

Configure an MX Series router as the data center gateway and WAN edge router and name it as MX12:

1. Set the system hostname.

```

[edit]
user@MX12# set system host-name MX12

```

2. Configure the P device as the logical system of MX12 data center gateway and WAN edge router.

Step-by-Step Procedure

- a. Configure the P device to operate in the enhanced-ip mode because the EVPN active-active functionality is supported on routers with MPCs and MIC interfaces only. A system reboot is required on committing this configuration.

```

[edit chassis]
user@P# set network-services enhanced-ip

```

- b. Configure the interfaces of the P device.

```

[edit]

user@P# set interfaces ge-1/0/4 unit 0 description "CONNECTED TO ToR12"
user@P# set interfaces ge-1/0/4 unit 0 family inet address 192.168.5.2/24
user@P# set interfaces ge-1/0/5 unit 0 description "CONNECTED TO TOR11"
user@P# set interfaces ge-1/0/5 unit 0 family inet address 192.168.4.2/24

```

```

user@P# set interfaces ge-1/0/6 unit 0 description "CONNECTED TO P"
user@P# set interfaces ge-1/0/6 unit 0 family inet address 203.0.2.1/24
user@P# set interfaces ge-1/0/6 unit 0 family mpls
user@P# set interfaces ge-1/1/0 unit 0 description "CONNECTED TO MX11"
user@P# set interfaces ge-1/1/0 unit 0 family inet address 192.168.7.2/24

```

- c. Enable RSVP, MPLS, BGP, and OSPF protocols on the core interfaces of the P device. Create MPLS LSPs and specify the address of the other gateway and WAN edge routers (MX11, MX12, MX21, MX22).

```

[edit]

user@P# set logical-systems P protocols rsvp interface all
user@P# set logical-systems P protocols mpls label-switched-path P-T0-MX11 from
203.0.113.1
user@P# set logical-systems P protocols mpls label-switched-path P-T0-MX11 to 192.0.2.21
user@P# set logical-systems P protocols mpls label-switched-path P-T0-MX12 to 192.0.2.22
user@P# set logical-systems P protocols mpls label-switched-path P-T0-MX21 to
198.51.100.21
user@P# set logical-systems P protocols mpls label-switched-path P-T0-MX22 to
198.51.100.22
user@P# set logical-systems P protocols mpls interface all
user@P# set logical-systems P protocols bgp local-address 203.0.113.1
user@P# set logical-systems P protocols bgp local-as 300
user@P# set logical-systems P protocols bgp group INT type internal
user@P# set logical-systems P protocols bgp group INT import BLOCK-VXLAN-ROUTES-FROM-CORE
user@P# set logical-systems P protocols bgp group INT family evpn signaling
user@P# set logical-systems P protocols bgp group INT cluster 203.0.113.1
user@P# set logical-systems P protocols bgp group INT neighbor 192.0.2.21
user@P# set logical-systems P protocols bgp group INT neighbor 192.0.2.22
user@P# set logical-systems P protocols bgp group INT neighbor 198.51.100.21
user@P# set logical-systems P protocols bgp group INT neighbor 198.51.100.22
user@P# set logical-systems P protocols ospf traffic-engineering
user@P# set logical-systems P protocols ospf area 0.0.0.0 interface all
user@P# set logical-systems P protocols ospf area 0.0.0.0 interface lo0.86

```

- d. Configure the loopback interface address and routing options.

```

[edit]

user@P# set logical-systems P interfaces lo0 unit 86 family inet address 203.0.113.1/32

```

```

user@P# set logical-systems P interfaces lo0 unit 86 family mpls
user@P# set logical-systems P routing-options router-id 203.0.113.1
user@P# set logical-systems P routing-options autonomous-system 300

```

- e. Configure routing policies to accept the direct loopback address route and redirect it into BGP.

```

[edit]

user@P# set logical-systems P policy-options policy-statement BLOCK-VXLAN-ROUTES-FROM-
CORE term 1 from protocol bgp
user@P# set logical-systems P policy-options policy-statement BLOCK-VXLAN-ROUTES-FROM-
CORE term 1 from community RT-CORE
user@P# set logical-systems P policy-options policy-statement BLOCK-VXLAN-ROUTES-FROM-
CORE term 1 then accept
user@P# set logical-systems P policy-options policy-statement BLOCK-VXLAN-ROUTES-FROM-
CORE term 2 from protocol bgp
user@P# set logical-systems P policy-options policy-statement BLOCK-VXLAN-ROUTES-FROM-
CORE term 2 from community RT-DC1
user@P# set logical-systems P policy-options policy-statement BLOCK-VXLAN-ROUTES-FROM-
CORE term 2 then reject
user@P# set logical-systems P policy-options policy-statement BLOCK-VXLAN-ROUTES-FROM-
CORE term 3 from protocol bgp
user@P# set logical-systems P policy-options policy-statement BLOCK-VXLAN-ROUTES-FROM-
CORE term 3 from community RT-DC2
user@P# set logical-systems P policy-options policy-statement BLOCK-VXLAN-ROUTES-FROM-
CORE term 3 then reject

```

- f. Configure community policy options.

```

[edit]

user@P# set logical-systems P policy-options community RT-CORE members target:1:2
user@P# set logical-systems P policy-options community RT-DC1 members target:1:1
user@P# set logical-systems P policy-options community RT-DC2 members target:1:3

```

- g. Configure trace options for the interfaces to enable trace logs.

```

[edit]

user@P# set interfaces traceoptions file R3-DCD.log

```

```
user@P# set interfaces traceoptions file size 10m
user@P# set interfaces traceoptions flag all
```

3. Configure the interfaces on the MX12 router (DC GW/WAN Edge 2) to enable the underlay connectivity to the MX11, ToR12, ToR11, and P devices, which is the EVPN-VXLAN part of DC1 network.

[edit]

```
user@MX12# set interfaces ge-1/0/4 unit 0 description "CONNECTED TO ToR12"
user@MX12# set interfaces ge-1/0/4 unit 0 family inet address 192.168.5.2/24
user@MX12# set interfaces ge-1/0/5 unit 0 description "CONNECTED TO TOR11"
user@MX12# set interfaces ge-1/0/5 unit 0 family inet address 192.168.4.2/24
user@MX12# set interfaces ge-1/0/6 unit 0 description "CONNECTED TO P"
user@MX12# set interfaces ge-1/0/6 unit 0 family inet address 203.0.2.1/24
user@MX12# set interfaces ge-1/0/6 unit 0 family mpls
user@MX12# set interfaces ge-1/1/0 unit 0 description "CONNECTED TO MX11"
user@MX12# set interfaces ge-1/1/0 unit 0 family inet address 192.168.7.2/24
```

4. Configure a multiprotocol external BGP (MP-EBGP) underlay connectivity between the gateway routers (MX11 and MX12) and ToR (ToR11 and ToR12).

[edit]

```
user@MX12# set protocols bgp group ToR11 type external
user@MX12# set protocols bgp group ToR11 local-address 192.168.4.2
user@MX12# set protocols bgp group ToR11 export TEST
user@MX12# set protocols bgp group ToR11 export L0
user@MX12# set protocols bgp group ToR11 peer-as 100
user@MX12# set protocols bgp group ToR11 local-as 500
user@MX12# set protocols bgp group ToR11 neighbor 192.168.4.1 family inet unicast
user@MX12# set protocols bgp group ToR12 type external
user@MX12# set protocols bgp group ToR12 local-address 192.168.5.2
user@MX12# set protocols bgp group ToR12 export TEST
user@MX12# set protocols bgp group ToR12 export L0
user@MX12# set protocols bgp group ToR12 peer-as 200
user@MX12# set protocols bgp group ToR12 local-as 500
user@MX12# set protocols bgp group ToR12 neighbor 192.168.5.1 family inet unicast
user@MX12# set protocols bgp group MX11 type external
user@MX12# set protocols bgp group MX11 local-address 192.168.7.2
user@MX12# set protocols bgp group MX11 export TEST
```



```

user@MX12# set protocols bgp group MX11 export L0
user@MX12# set protocols bgp group MX11 peer-as 400
user@MX12# set protocols bgp group MX11 local-as 500
user@MX12# set protocols bgp group MX11 neighbor 192.168.7.1 family inet unicast

```

5. Configure a multiprotocol external BGP (MP-EBGP) overlay connectivity between the gateway routers (MX11 and MX12) and ToR (ToR11 and ToR12) and set EVPN as the signaling protocol.

```

[edit]

user@MX12# set protocols bgp group ToR11-EVPN type external
user@MX12# set protocols bgp group ToR11-EVPN multihop ttl 2
user@MX12# set protocols bgp group ToR11-EVPN multihop no-nexthop-change
user@MX12# set protocols bgp group ToR11-EVPN local-address 192.0.2.22
user@MX12# set protocols bgp group ToR11-EVPN export TEST
user@MX12# set protocols bgp group ToR11-EVPN peer-as 100
user@MX12# set protocols bgp group ToR11-EVPN local-as 500
user@MX12# set protocols bgp group ToR11-EVPN neighbor 192.0.2.11 family evpn signaling
user@MX12# set protocols bgp group ToR12-EVPN type external
user@MX12# set protocols bgp group ToR12-EVPN multihop ttl 2
user@MX12# set protocols bgp group ToR12-EVPN multihop no-nexthop-change
user@MX12# set protocols bgp group ToR12-EVPN local-address 192.0.2.22
user@MX12# set protocols bgp group ToR12-EVPN export TEST
user@MX12# set protocols bgp group ToR12-EVPN peer-as 200
user@MX12# set protocols bgp group ToR12-EVPN local-as 500
user@MX12# set protocols bgp group ToR12-EVPN neighbor 192.0.2.12 family evpn signaling
user@MX12# set protocols bgp group MX11-EVPN type external
user@MX12# set protocols bgp group MX11-EVPN multihop ttl 2
user@MX12# set protocols bgp group MX11-EVPN multihop no-nexthop-change
user@MX12# set protocols bgp group MX11-EVPN local-address 192.0.2.22
user@MX12# set protocols bgp group MX11-EVPN export TEST
user@MX12# set protocols bgp group MX11-EVPN peer-as 400
user@MX12# set protocols bgp group MX11-EVPN local-as 500
user@MX12# set protocols bgp group MX11-EVPN neighbor 192.0.2.21 family evpn signaling
user@MX12# set protocols bgp group MX12-EVPN export TEST

```

6. Configure integrated routing and bridging (IRB) interfaces that advertises the MAC and IP routes (MAC+IP type 2 routes) for hosts in the topology. The IRB configuration is the gateway for the VLANs on the hosts.

Step-by-Step Procedure

- a. The following is the IRB gateway configuration for the VLAN-1 on Host (which is the host part of VLAN-1):

```
[edit]

user@MX12# set interfaces irb unit 1 proxy-macip-advertisement
user@MX12# set interfaces irb unit 1 virtual-gateway-esi 00:11:aa:aa:aa:aa:aa:aa:aa
user@MX12# set interfaces irb unit 1 virtual-gateway-esi all-active
user@MX12# set interfaces irb unit 1 family inet address 10.11.1.13/24 virtual-gateway-
address 10.11.1.10
```

- b. The following is the IRB gateway configuration for the VLAN-2 on Host (which is the host part of VLAN-2):

```
[edit]

user@MX12# set interfaces irb unit 2 proxy-macip-advertisement
user@MX12# set interfaces irb unit 2 virtual-gateway-esi 00:11:bb:bb:bb:bb:bb:bb:bb
user@MX12# set interfaces irb unit 2 virtual-gateway-esi all-active
user@MX12# set interfaces irb unit 2 family inet address 10.12.1.13/24 virtual-gateway-
address 10.12.1.10
```

- c. The following is the IRB gateway configuration for the VLAN-3 on Host (which is the host part of VLAN-3):

```
[edit]

user@MX12# set interfaces irb unit 3 proxy-macip-advertisement
user@MX12# set interfaces irb unit 3 virtual-gateway-esi 00:11:cc:cc:cc:cc:cc:cc:cc
user@MX12# set interfaces irb unit 3 virtual-gateway-esi all-active
user@MX12# set interfaces irb unit 3 family inet address 10.13.1.13/24 virtual-gateway-
address 10.13.1.10
```

- d. The following is the IRB gateway configuration for the VLAN-4 on Host (which is the host part of VLAN-4):

```
[edit]

user@MX12# set interfaces irb unit 4 proxy-macip-advertisement
user@MX12# set interfaces irb unit 4 virtual-gateway-esi 00:11:dd:dd:dd:dd:dd:dd:dd
user@MX12# set interfaces irb unit 4 virtual-gateway-esi all-active
user@MX12# set interfaces irb unit 4 family inet address 10.14.1.13/24 virtual-gateway-
address 10.14.1.10
```

- e. The following is the IRB gateway configuration for the VLAN-5 on Host (which is the host part of VLAN-5):

```
[edit]

user@MX12# set interfaces irb unit 5 proxy-macip-advertisement
user@MX12# set interfaces irb unit 5 virtual-gateway-esi 00:11:ee:ee:ee:ee:ee:ee:ee
user@MX12# set interfaces irb unit 5 virtual-gateway-esi all-active
user@MX12# set interfaces irb unit 5 family inet address 10.15.1.13/24 virtual-gateway-
address 10.15.1.10
```

7. Configure trace operations to track all Layer 2 address learning and forwarding properties.

```
[edit]

user@MX12# set protocols l2-learning traceoptions file MX12-L2ALD.log
user@MX12# set protocols l2-learning traceoptions file size 10m
user@MX12# set protocols l2-learning traceoptions level all
user@MX12# set protocols l2-learning traceoptions flag all
```

8. Configure routing policies to accept the direct loopback address route and redirect it into BGP.

```
[edit]

user@MX12# set policy-options policy-statement L0 from protocol direct
user@MX12# set policy-options policy-statement L0 from route-filter 192.0.2.22/32 exact
user@MX12# set policy-options policy-statement L0 then accept
user@MX12# set policy-options policy-statement L0 term 1 from protocol direct
```

```

user@MX12# set policy-options policy-statement L0 term 1 from route-filter 192.0.2.22/32
exact
user@MX12# set policy-options policy-statement L0 term 1 then accept
user@MX12# set policy-options policy-statement TEST from protocol bgp
user@MX12# set policy-options policy-statement TEST from protocol evpn

```

9. Configure community policy options.

```

[edit]

user@MX12# set policy-options community NO-EXPORT members no-advertise
user@MX12# set policy-options community NO-EXPORT members no-export
user@MX12# set policy-options community NO-EXPORT members no-export-subconfed

```

10. Apply load balance.

```

[edit]

user@MX12# set policy-options policy-statement TEST then community add NO-EXPORT
user@MX12# set policy-options policy-statement evpn-pplb from protocol evpn
user@MX12# set policy-options policy-statement evpn-pplb then load-balance per-packet

```

11. Configure an ESI value on the logical tunnel interface. Use the same ESI value on all other gateway/WAN edge routers in the DC1 network.

```

[edit]

user@MX12# set interfaces lt-1/0/0 esi 00:22:22:22:22:22:22:22:22

```

12. Configure active-active multihoming on the logical tunnel interface by including the all-active statement.

```

[edit]

user@MX12# set interfaces lt-1/0/0 esi all-active

```

13. Configure a pair of logical tunnel (lt-) interface on the MX12 gateway router to interconnect the EVPN-VXLAN instance of the data center network with the MPLS-based EVPN instance of the

WAN. One logical tunnel (lt-) interface is configured as the access interface for EVPN-VXLAN and the other logical tunnel (lt-) interface is configured as the access interface for MPLS-based EVPN.

```
[edit]
```

```
user@MX12# set interfaces lt-1/0/0 unit 0 peer-unit 1
user@MX12# set interfaces lt-1/0/0 unit 0 family bridge interface-mode trunk
user@MX12# set interfaces lt-1/0/0 unit 0 family bridge vlan-id-list 1-5
user@MX12# set interfaces lt-1/0/0 unit 1 peer-unit 0
user@MX12# set interfaces lt-1/0/0 unit 1 family bridge interface-mode trunk
user@MX12# set interfaces lt-1/0/0 unit 1 family bridge vlan-id-list 1-5
```

14. Configure the loopback interface address and routing options.

```
[edit]
```

```
user@MX12# set interfaces lo0 unit 85 family inet address 192.0.2.22/32
user@MX12# set interfaces lo0 unit 85 family mpls
user@MX12# set routing-options router-id 192.0.2.22
user@MX12# set routing-options autonomous-system 300
```

15. Configure load balancing on MX12.

```
[edit]
```

```
user@MX12# set routing-options forwarding-table export evpn-pplb
```

16. Enable RSVP, MPLS, BGP, and OSPF protocols on the core interfaces. Create MPLS LSPs and specify the address of the other gateway and WAN edge routers (MX11, MX21, P, MX22).

```
[edit]
```

```
user@MX12# set protocols rsvp interface all
user@MX12# set protocols rsvp interface fxp0.0 disable
user@MX12# set protocols mpls label-switched-path MX12-T0-MX11 to 192.0.2.21
user@MX12# set protocols mpls label-switched-path MX12-T0-P to 203.0.113.1
user@MX12# set protocols mpls label-switched-path MX12-T0-MX21 to 198.51.100.21
user@MX12# set protocols mpls label-switched-path MX12-T0-MX22 to 198.51.100.22
user@MX12# set protocols mpls interface all
```

```

user@MX12# set protocols mpls interface fxp0.0 disable
user@MX12# set protocols bgp local-address 192.0.2.22
user@MX12# set protocols bgp local-as 300
user@MX12# set protocols bgp group INT type internal
user@MX12# set protocols bgp group INT family evpn signaling
user@MX12# set protocols bgp group INT export TEST
user@MX12# set protocols bgp group INT neighbor 203.0.113.1
user@MX12# set protocols ospf traffic-engineering
user@MX12# set protocols ospf area 0.0.0.0 interface ge-1/0/6.0
user@MX12# set protocols ospf area 0.0.0.0 interface lo0.85 passive

```

17. Configure EVPN-based MPLS routing instances on the MX12 router for each virtual network. Define the route distinguisher (used to identify and advertise EVPN-MPLS routes) and vrf-target (exports and tags all routes for that local VRF using the defined route target). Configure a bridge domain for each virtual router that maps VLAN IDs.

[edit]

```

user@MX12# set routing-instances EVPN-MPLS-1 instance-type virtual-switch
user@MX12# set routing-instances EVPN-MPLS-1 interface lt-1/0/0.0
user@MX12# set routing-instances EVPN-MPLS-1 route-distinguisher 192.0.2.22:100
user@MX12# set routing-instances EVPN-MPLS-1 vrf-target target:1:2
user@MX12# set routing-instances EVPN-MPLS-1 protocols evpn traceoptions file MX12-EVPN-
MPLS-1.log
user@MX12# set routing-instances EVPN-MPLS-1 protocols evpn traceoptions file size 10m
user@MX12# set routing-instances EVPN-MPLS-1 protocols evpn traceoptions flag all
user@MX12# set routing-instances EVPN-MPLS-1 protocols evpn extended-vlan-list 1-5
user@MX12# set routing-instances EVPN-MPLS-1 protocols evpn default-gateway no-gateway-
community
user@MX12# set routing-instances EVPN-MPLS-1 bridge-domains BD-1 domain-type bridge
user@MX12# set routing-instances EVPN-MPLS-1 bridge-domains BD-1 vlan-id 1
user@MX12# set routing-instances EVPN-MPLS-1 bridge-domains BD-2 domain-type bridge
user@MX12# set routing-instances EVPN-MPLS-1 bridge-domains BD-2 vlan-id 2
user@MX12# set routing-instances EVPN-MPLS-1 bridge-domains BD-3 domain-type bridge
user@MX12# set routing-instances EVPN-MPLS-1 bridge-domains BD-3 vlan-id 3
user@MX12# set routing-instances EVPN-MPLS-1 bridge-domains BD-4 domain-type bridge
user@MX12# set routing-instances EVPN-MPLS-1 bridge-domains BD-4 vlan-id 4
user@MX12# set routing-instances EVPN-MPLS-1 bridge-domains BD-5 domain-type bridge
user@MX12# set routing-instances EVPN-MPLS-1 bridge-domains BD-5 vlan-id 5

```

18. Configure EVPN-VXLAN routing instances on the MX12 router for each virtual network. Define the VTEP source interface, route distinguisher (used to identify and advertise EVPN routes), and vrf-target (exports and tags all routes for that local VRF using the defined route target). Configure the EVPN protocol, encapsulation method, VNI list, and BUM traffic forwarding method. Finally, configure a bridge domain for each virtual router that maps VNIDs to VLAN IDs, and identify the BUM forwarding method.

[edit]

```

user@MX12# set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.85
user@MX12# set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
user@MX12# set routing-instances EVPN-VXLAN-1 interface lt-1/0/0.1
user@MX12# set routing-instances EVPN-VXLAN-1 route-distinguisher 192.0.2.22:1
user@MX12# set routing-instances EVPN-VXLAN-1 vrf-target target:1:1
user@MX12# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file MX12-EVPN-
VXLAN-1.log
user@MX12# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file size 10m
user@MX12# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions flag all
user@MX12# set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation vxlan
user@MX12# set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list 1-4
user@MX12# set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list 5
user@MX12# set routing-instances EVPN-VXLAN-1 protocols evpn default-gateway no-gateway-
community
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type bridge
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id 1
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 routing-interface irb.1
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni 1
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type bridge
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id 2
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 routing-interface irb.2
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni 2
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type bridge
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id 3
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 routing-interface irb.3
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni 3
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type bridge
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id 4
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 routing-interface irb.4
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni 4
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type bridge
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id 5
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 routing-interface irb.5

```

```

user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni 5
user@MX12# set routing-instances VRF instance-type vrf
user@MX12# set routing-instances VRF interface irb.1
user@MX12# set routing-instances VRF interface irb.2
user@MX12# set routing-instances VRF interface irb.3
user@MX12# set routing-instances VRF interface irb.4
user@MX12# set routing-instances VRF interface irb.5
user@MX12# set routing-instances VRF route-distinguisher 1:1
user@MX12# set routing-instances VRF vrf-target target:10:10

```

Configuring Data Center Gateway and WAN Edge 3 Router (MX21)

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

Configure an MX Series router as the data center gateway and WAN edge router and name it as MX21:

1. Set the system hostname.

```

[edit]
user@MX21# set system host-name MX21

```

2. Configure the interfaces on the MX21 router (DC GW/WAN Edge 3) to enable the underlay connectivity to the MX22, ToR22, ToR21, and P devices, which is the EVPN-VXLAN part of DC2 network.

```

[edit]

user@MX21# set interfaces ge-3/0/0 unit 0 description "CONNECTED TO MX22"
user@MX21# set interfaces ge-3/0/0 unit 0 family inet address 192.168.13.1/24
user@MX21# set interfaces ge-3/1/0 unit 0 description "CONNECTED TO ToR22"
user@MX21# set interfaces ge-3/1/0 unit 0 family inet address 192.168.8.1/24
user@MX21# set interfaces ge-5/0/0 unit 0 description "CONNECTED TO P"
user@MX21# set interfaces ge-5/0/0 unit 0 family inet address 203.0.4.1/24
user@MX21# set interfaces ge-5/0/0 unit 0 family mpls
user@MX21# set interfaces ge-5/0/1 unit 0 description "CONNECTED TO ToR21"
user@MX21# set interfaces ge-5/0/1 unit 0 family inet address 192.168.9.1/24

```


3. Configure a multiprotocol external BGP (MP-EBGP) underlay connectivity between the gateway routers (MX21 and MX22) and ToR (ToR21 and ToR22).

[edit]

```

user@MX21# set protocols bgp group ToR21 type external
user@MX21# set protocols bgp group ToR21 local-address 192.168.9.1
user@MX21# set protocols bgp group ToR21 export TEST
user@MX21# set protocols bgp group ToR21 export L0
user@MX21# set protocols bgp group ToR21 peer-as 600
user@MX21# set protocols bgp group ToR21 local-as 800
user@MX21# set protocols bgp group ToR21 neighbor 192.168.9.2 family inet unicast
user@MX21# set protocols bgp group ToR22 type external
user@MX21# set protocols bgp group ToR22 local-address 192.168.8.1
user@MX21# set protocols bgp group ToR22 export TEST
user@MX21# set protocols bgp group ToR22 export L0
user@MX21# set protocols bgp group ToR22 peer-as 700
user@MX21# set protocols bgp group ToR22 local-as 800
user@MX21# set protocols bgp group ToR22 neighbor 192.168.8.2 family inet unicast
user@MX21# set protocols bgp group MX22 type external
user@MX21# set protocols bgp group MX22 local-address 192.168.13.1
user@MX21# set protocols bgp group MX22 export TEST
user@MX21# set protocols bgp group MX22 export L0
user@MX21# set protocols bgp group MX22 peer-as 900
user@MX21# set protocols bgp group MX22 local-as 800
user@MX21# set protocols bgp group MX22 neighbor 10.115.15.2 family inet unicast

```

4. Configure a multiprotocol external BGP (MP-EBGP) overlay connectivity between the gateway routers (MX21 and MX22) and ToR (ToR21 and ToR22) and set EVPN as the signaling protocol.

[edit]

```

user@MX21# set protocols bgp group ToR21-EVPN type external
user@MX21# set protocols bgp group ToR21-EVPN multihop ttl 2
user@MX21# set protocols bgp group ToR21-EVPN multihop no-nexthop-change
user@MX21# set protocols bgp group ToR21-EVPN local-address 198.51.100.21
user@MX21# set protocols bgp group ToR21-EVPN peer-as 600
user@MX21# set protocols bgp group ToR21-EVPN local-as 800
user@MX21# set protocols bgp group ToR21-EVPN neighbor 198.51.100.11 family evpn signaling
user@MX21# set protocols bgp group ToR22-EVPN type external
user@MX21# set protocols bgp group ToR22-EVPN multihop ttl 2

```

```

user@MX21# set protocols bgp group ToR22-EVPN multihop no-nexthop-change
user@MX21# set protocols bgp group ToR22-EVPN local-address 198.51.100.21
user@MX21# set protocols bgp group ToR22-EVPN peer-as 700
user@MX21# set protocols bgp group ToR22-EVPN local-as 800
user@MX21# set protocols bgp group ToR22-EVPN neighbor 198.51.100.12 family evpn signaling
user@MX21# set protocols bgp group MX22-EVPN type external
user@MX21# set protocols bgp group MX22-EVPN multihop ttl 2
user@MX21# set protocols bgp group MX22-EVPN multihop no-nexthop-change
user@MX21# set protocols bgp group MX22-EVPN local-address 198.51.100.21
user@MX21# set protocols bgp group MX22-EVPN peer-as 900
user@MX21# set protocols bgp group MX22-EVPN local-as 800
user@MX21# set protocols bgp group MX22-EVPN neighbor 198.51.100.22 family evpn signaling

```

5. Configure integrated routing and bridging (IRB) interfaces that advertises the MAC and IP routes (MAC+IP type 2 routes) for hosts in the topology. The IRB configuration is the gateway for the VLANs on the hosts.

Step-by-Step Procedure

- a. The following is the IRB gateway configuration for the VLAN-1 on Host (which is the host part of VLAN-1):

[edit]

```

user@MX21# set interfaces irb unit 1 proxy-macip-advertisement
user@MX21# set interfaces irb unit 1 virtual-gateway-esi 00:22:aa:aa:aa:aa:aa:aa:aa:aa
user@MX21# set interfaces irb unit 1 virtual-gateway-esi all-active
user@MX21# set interfaces irb unit 1 family inet address 10.11.1.14/24 virtual-gateway-
address 10.11.1.11

```

- b. The following is the IRB gateway configuration for the VLAN-2 on Host (which is the host part of VLAN-2):

[edit]

```

user@MX21# set interfaces irb unit 2 proxy-macip-advertisement
user@MX21# set interfaces irb unit 2 virtual-gateway-esi 00:22:bb:bb:bb:bb:bb:bb:bb:bb
user@MX21# set interfaces irb unit 2 virtual-gateway-esi all-active

```

```
user@MX21# set interfaces irb unit 2 family inet address 10.12.1.14/24 virtual-gateway-
address 10.12.1.11
```

- c. The following is the IRB gateway configuration for the VLAN-3 on Host (which is the host part of VLAN-3):

```
[edit]
```

```
user@MX21# set interfaces irb unit 3 proxy-macip-advertisement
user@MX21# set interfaces irb unit 3 virtual-gateway-esi 00:22:cc:cc:cc:cc:cc:cc:cc
user@MX21# set interfaces irb unit 3 virtual-gateway-esi all-active
user@MX21# set interfaces irb unit 3 family inet address 10.13.1.14/24 virtual-gateway-
address 10.13.1.11
```

- d. The following is the IRB gateway configuration for the VLAN-4 on Host (which is the host part of VLAN-4):

```
[edit]
```

```
user@MX21# set interfaces irb unit 4 proxy-macip-advertisement
user@MX21# set interfaces irb unit 4 virtual-gateway-esi 00:22:dd:dd:dd:dd:dd:dd:dd
user@MX21# set interfaces irb unit 4 virtual-gateway-esi all-active
user@MX21# set interfaces irb unit 4 family inet address 10.14.1.14/24 virtual-gateway-
address 10.14.1.11
```

- e. The following is the IRB gateway configuration for the VLAN-5 on Host (which is the host part of VLAN-5):

```
[edit]
```

```
user@MX21# set interfaces irb unit 5 proxy-macip-advertisement
user@MX21# set interfaces irb unit 5 virtual-gateway-esi 00:22:ee:ee:ee:ee:ee:ee:ee
user@MX21# set interfaces irb unit 5 virtual-gateway-esi all-active
user@MX21# set interfaces irb unit 5 family inet address 10.15.1.14/24 virtual-gateway-
address 10.15.1.11
```

6. Configure trace operations to track all Layer 2 address learning and forwarding properties.

[edit]

```
user@MX21# set protocols l2-learning traceoptions file MX21-L2ALD.log
user@MX21# set protocols l2-learning traceoptions file size 10m
user@MX21# set protocols l2-learning traceoptions level all
user@MX21# set protocols l2-learning traceoptions flag all
```

7. Configure routing policies to accept the direct loopback address route and redirect it into BGP.

[edit]

```
user@MX21# set policy-options policy-statement L0 from protocol direct
user@MX21# set policy-options policy-statement L0 from route-filter 198.51.100.21/32 exact
user@MX21# set policy-options policy-statement L0 then accept
user@MX21# set policy-options policy-statement TEST1 term 1 from protocol bgp
user@MX21# set policy-options policy-statement TEST1 term 1 from external
user@MX21# set policy-options policy-statement TEST1 term 1 then reject
```

8. Configure community policy options.

[edit]

```
user@MX21# set policy-options community NO-EXPORT members no-advertise
user@MX21# set policy-options community NO-EXPORT members no-export
user@MX21# set policy-options community NO-EXPORT members no-export-subconfed
```

9. Apply load balance.

[edit]

```
user@MX21# set policy-options policy-statement TEST then community add NO-EXPORT
user@MX21# set policy-options policy-statement evpn-pplb from protocol evpn
user@MX21# set policy-options policy-statement evpn-pplb then load-balance per-packet
```

10. Configure an ESI value on the logical tunnel interface. Use the same ESI value on all other gateway/WAN edge routers in the DC2 network.

```
[edit]
```

```
user@MX21# set interfaces lt-5/0/0 esi 00:33:33:33:33:33:33:33:33
```

11. Configure active-active multihoming on the logical tunnel interface by including the all-active statement.

```
[edit]
```

```
user@MX21# set interfaces lt-5/0/0 esi all-active
```

12. Configure a pair of logical tunnel (lt-) interface on the MX21 gateway router to interconnect the EVPN-VXLAN instance of the data center network with the MPLS-based EVPN instance of the WAN. One logical tunnel (lt-) interface is configured as the access interface for EVPN-VXLAN and the other logical tunnel (lt-) interface is configured as the access interface for MPLS-based EVPN.

```
[edit]
```

```
user@MX21# set interfaces lt-5/0/0 unit 0 peer-unit 1
user@MX21# set interfaces lt-5/0/0 unit 0 family bridge interface-mode trunk
user@MX21# set interfaces lt-5/0/0 unit 0 family bridge vlan-id-list 1-5
user@MX21# set interfaces lt-5/0/0 unit 1 peer-unit 0
user@MX21# set interfaces lt-5/0/0 unit 1 family bridge interface-mode trunk
user@MX21# set interfaces lt-5/0/0 unit 1 family bridge vlan-id-list 1-5
```

13. Configure the loopback interface address and routing options.

```
[edit]
```

```
user@MX21# set interfaces lo0 unit 87 family inet address 198.51.100.21/32
user@MX21# set interfaces lo0 unit 87 family mpls
user@MX21# set routing-options router-id 198.51.100.21
user@MX21# set routing-options autonomous-system 300
```

14. Configure load balancing on MX21.

[edit]

```
user@MX21# set routing-options forwarding-table export evpn-pplb
```

15. Enable RSVP, MPLS, BGP, and OSPF protocols on the core interfaces. Create MPLS LSPs and specify the address of the other gateway and WAN edge routers (MX11, MX12, P, MX22).

[edit]

```
user@MX21# set protocols rsvp interface all
user@MX21# set protocols rsvp interface fxp0.0 disable
user@MX21# set protocols mpls label-switched-path MX21-T0-MX11 to 192.0.2.21
user@MX21# set protocols mpls label-switched-path MX21-T0-MX12 to 192.0.2.22
user@MX21# set protocols mpls label-switched-path MX21-T0-P to 203.0.113.1
user@MX21# set protocols mpls label-switched-path MX21-T0-MX22 to 198.51.100.22
user@MX21# set protocols mpls interface all
user@MX21# set protocols mpls interface fxp0.0 disable
user@MX21# set protocols bgp local-address 198.51.100.21
user@MX21# set protocols bgp export TEST
user@MX21# set protocols bgp local-as 300
user@MX21# set protocols bgp group INT type internal
user@MX21# set protocols bgp group INT local-address 198.51.100.21
user@MX21# set protocols bgp group INT family evpn signaling
user@MX21# set protocols bgp group INT export TEST1
user@MX21# set protocols bgp group INT neighbor 203.0.113.1
user@MX21# set protocols ospf traffic-engineering
user@MX21# set protocols ospf area 0.0.0.0 interface ge-5/0/0.0
user@MX21# set protocols ospf area 0.0.0.0 interface lo0.87 passive
```

16. Configure EVPN-based MPLS routing instances on the MX21 router for each virtual network. Define the route distinguisher (used to identify and advertise EVPN-MPLS routes) and vrf-target (exports and tags all routes for that local VRF using the defined route target). Configure a bridge domain for each virtual router that maps VLAN IDs.

[edit]

```
user@MX21# set routing-instances EVPN-MPLS-1 instance-type virtual-switch
user@MX21# set routing-instances EVPN-MPLS-1 interface lt-5/0/0.0
```

```

user@MX21# set routing-instances EVPN-MPLS-1 route-distinguisher 198.51.100.21:100
user@MX21# set routing-instances EVPN-MPLS-1 vrf-target target:1:2
user@MX21# set routing-instances EVPN-MPLS-1 protocols evpn traceoptions file MX21-EVPN-
MPLS-1.log
user@MX21# set routing-instances EVPN-MPLS-1 protocols evpn traceoptions file size 10m
user@MX21# set routing-instances EVPN-MPLS-1 protocols evpn traceoptions flag all
user@MX21# set routing-instances EVPN-MPLS-1 protocols evpn extended-vlan-list 1-5
user@MX21# set routing-instances EVPN-MPLS-1 protocols evpn default-gateway no-gateway-
community
user@MX21# set routing-instances EVPN-MPLS-1 bridge-domains BD-1 domain-type bridge
user@MX21# set routing-instances EVPN-MPLS-1 bridge-domains BD-1 vlan-id 1
user@MX21# set routing-instances EVPN-MPLS-1 bridge-domains BD-2 domain-type bridge
user@MX21# set routing-instances EVPN-MPLS-1 bridge-domains BD-2 vlan-id 2
user@MX21# set routing-instances EVPN-MPLS-1 bridge-domains BD-3 domain-type bridge
user@MX21# set routing-instances EVPN-MPLS-1 bridge-domains BD-3 vlan-id 3
user@MX21# set routing-instances EVPN-MPLS-1 bridge-domains BD-4 domain-type bridge
user@MX21# set routing-instances EVPN-MPLS-1 bridge-domains BD-4 vlan-id 4
user@MX21# set routing-instances EVPN-MPLS-1 bridge-domains BD-5 domain-type bridge
user@MX21# set routing-instances EVPN-MPLS-1 bridge-domains BD-5 vlan-id 5

```

17. Configure EVPN-VXLAN routing instances on the MX21 router for each virtual network. Define the VTEP source interface, route distinguisher (used to identify and advertise EVPN routes), and vrf-target (exports and tags all routes for that local VRF using the defined route target). Configure the EVPN protocol, encapsulation method, VNI list, and BUM traffic forwarding method. Finally, configure a bridge domain for each virtual router that maps VNIDs to VLAN IDs, and identify the BUM forwarding method.

[edit]

```

user@MX21# set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.87
user@MX21# set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
user@MX21# set routing-instances EVPN-VXLAN-1 interface lt-5/0/0.1
user@MX21# set routing-instances EVPN-VXLAN-1 route-distinguisher 198.51.100.21:1
user@MX21# set routing-instances EVPN-VXLAN-1 vrf-target target:1:3
user@MX21# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file MX21-EVPN-
VXLAN-1.log
user@MX21# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file size 10m
user@MX21# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions flag all
user@MX21# set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation vxlan
user@MX21# set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list 1-5
user@MX21# set routing-instances EVPN-VXLAN-1 protocols evpn default-gateway no-gateway-
community

```

```

user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type bridge
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id 1
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 routing-interface irb.1
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni 1
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type bridge
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id 2
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 routing-interface irb.2
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni 2
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type bridge
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id 3
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 routing-interface irb.3
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni 3
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type bridge
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id 4
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 routing-interface irb.4
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni 4
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type bridge
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id 5
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 routing-interface irb.5
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni 5
user@MX21# set routing-instances VRF instance-type vrf
user@MX21# set routing-instances VRF interface irb.1
user@MX21# set routing-instances VRF interface irb.2
user@MX21# set routing-instances VRF interface irb.3
user@MX21# set routing-instances VRF interface irb.4
user@MX21# set routing-instances VRF interface irb.5
user@MX21# set routing-instances VRF route-distinguisher 1:1
user@MX21# set routing-instances VRF vrf-target target:10:10

```

Configuring Data Center Gateway and WAN Edge 4 Router (MX22)

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

Configure an MX Series router as the data center gateway and WAN edge router and name it as MX22:

1. Set the system hostname.

```
[edit]
user@MX22# set system host-name MX22
```

2. Configure the interfaces on the MX22 router (DC GW/WAN Edge 4) to enable the underlay connectivity to the MX22, ToR21, MX21, and P devices, which is the EVPN-VXLAN part of DC2 network.

```
[edit]

user@MX22# set interfaces xe-0/0/0 unit 0 description "CONNECTED TO ToR22"
user@MX22# set interfaces xe-0/0/0 unit 0 family inet address 192.168.11.1/24
user@MX22# set interfaces xe-0/0/1 unit 0 description "CONNECTED TO ToR21"
user@MX22# set interfaces xe-0/0/1 unit 0 family inet address 192.168.10.1/24
user@MX22# set interfaces ge-1/0/0 unit 0 description "CONNECTED TO MX21"
user@MX22# set interfaces ge-1/0/0 unit 0 family inet address 10.115.15.2/24
user@MX22# set interfaces ge-1/0/2 unit 0 description "CONNECTED TO P"
user@MX22# set interfaces ge-1/0/2 unit 0 family inet address 203.0.3.1/24
user@MX22# set interfaces ge-1/0/2 unit 0 family mpls
```

3. Configure a multiprotocol external BGP (MP-EBGP) underlay connectivity between the gateway routers (MX21 and MX22) and ToR (ToR21 and ToR22).

```
[edit]

user@MX22# set protocols bgp group ToR21 type external
user@MX22# set protocols bgp group ToR21 local-address 192.168.10.1
user@MX22# set protocols bgp group ToR21 export TEST
user@MX22# set protocols bgp group ToR21 export L0
user@MX22# set protocols bgp group ToR21 peer-as 600
user@MX22# set protocols bgp group ToR21 local-as 900
user@MX22# set protocols bgp group ToR21 neighbor 10.102.2.1 family inet unicast
user@MX22# set protocols bgp group ToR22 type external
user@MX22# set protocols bgp group ToR22 local-address 192.168.11.1
user@MX22# set protocols bgp group ToR22 export TEST
user@MX22# set protocols bgp group ToR22 export L0
user@MX22# set protocols bgp group ToR22 peer-as 700
user@MX22# set protocols bgp group ToR22 local-as 900
user@MX22# set protocols bgp group ToR22 neighbor 192.168.11.2 family inet unicast
```

```

user@MX22# set protocols bgp group MX21 type external
user@MX22# set protocols bgp group MX21 local-address 10.115.15.2
user@MX22# set protocols bgp group MX21 export TEST
user@MX22# set protocols bgp group MX21 export L0
user@MX22# set protocols bgp group MX21 peer-as 800
user@MX22# set protocols bgp group MX21 local-as 900
user@MX22# set protocols bgp group MX21 neighbor 192.168.13.1 family inet unicast

```

4. Configure a multiprotocol external BGP (MP-EBGP) overlay connectivity between the gateway routers (MX21 and MX22) and ToR (ToR21 and ToR22) and set EVPN as the signaling protocol.

[edit]

```

user@MX22# set protocols bgp group ToR21-EVPN type external
user@MX22# set protocols bgp group ToR21-EVPN multihop ttl 2
user@MX22# set protocols bgp group ToR21-EVPN multihop no-nexthop-change
user@MX22# set protocols bgp group ToR21-EVPN local-address 198.51.100.22
user@MX22# set protocols bgp group ToR21-EVPN peer-as 600
user@MX22# set protocols bgp group ToR21-EVPN local-as 900
user@MX22# set protocols bgp group ToR21-EVPN neighbor 198.51.100.11 family evpn signaling
user@MX22# set protocols bgp group ToR22-EVPN type external
user@MX22# set protocols bgp group ToR22-EVPN multihop ttl 2
user@MX22# set protocols bgp group ToR22-EVPN multihop no-nexthop-change
user@MX22# set protocols bgp group ToR22-EVPN local-address 198.51.100.22
user@MX22# set protocols bgp group ToR22-EVPN peer-as 700
user@MX22# set protocols bgp group ToR22-EVPN local-as 900
user@MX22# set protocols bgp group ToR22-EVPN neighbor 198.51.100.12 family evpn signaling
user@MX22# set protocols bgp group MX21-EVPN type external
user@MX22# set protocols bgp group MX21-EVPN multihop ttl 2
user@MX22# set protocols bgp group MX21-EVPN multihop no-nexthop-change
user@MX22# set protocols bgp group MX21-EVPN local-address 198.51.100.22
user@MX22# set protocols bgp group MX21-EVPN peer-as 800
user@MX22# set protocols bgp group MX21-EVPN local-as 900
user@MX22# set protocols bgp group MX21-EVPN neighbor 198.51.100.21 family evpn signaling

```

5. Configure integrated routing and bridging (IRB) interfaces that advertises the MAC and IP routes (MAC+IP type 2 routes) for hosts in the topology. The IRB configuration is the gateway for the VLANs on the hosts.

Step-by-Step Procedure

- a. The following is the IRB gateway configuration for the VLAN-1 on Host (which is the host part of VLAN-1):

```
[edit]

user@MX22# set interfaces irb unit 1 proxy-macip-advertisement
user@MX22# set interfaces irb unit 1 virtual-gateway-esi 00:22:aa:aa:aa:aa:aa:aa:aa:aa
user@MX22# set interfaces irb unit 1 virtual-gateway-esi all-active
user@MX22# set interfaces irb unit 1 family inet address 10.11.1.15/24 virtual-gateway-
address 10.11.1.11
```

- b. The following is the IRB gateway configuration for the VLAN-2 on Host (which is the host part of VLAN-2):

```
[edit]

user@MX22# set interfaces irb unit 2 proxy-macip-advertisement
user@MX22# set interfaces irb unit 2 virtual-gateway-esi 00:22:bb:bb:bb:bb:bb:bb:bb:bb
user@MX22# set interfaces irb unit 2 virtual-gateway-esi all-active
user@MX22# set interfaces irb unit 2 family inet address 10.12.1.15/24 virtual-gateway-
address 10.12.1.11
```

- c. The following is the IRB gateway configuration for the VLAN-3 on Host (which is the host part of VLAN-3):

```
[edit]

user@MX22# set interfaces irb unit 3 proxy-macip-advertisement
user@MX22# set interfaces irb unit 3 virtual-gateway-esi 00:22:cc:cc:cc:cc:cc:cc:cc:cc
user@MX22# set interfaces irb unit 3 virtual-gateway-esi all-active
user@MX22# set interfaces irb unit 3 family inet address 10.13.1.15/24 virtual-gateway-
address 10.13.1.11
```

- d. The following is the IRB gateway configuration for the VLAN-4 on Host (which is the host part of VLAN-4):

```
[edit]

user@MX22# set interfaces irb unit 4 proxy-macip-advertisement
```

```

user@MX22# set interfaces irb unit 4 virtual-gateway-esi 00:22:dd:dd:dd:dd:dd:dd:dd
user@MX22# set interfaces irb unit 4 virtual-gateway-esi all-active
user@MX22# set interfaces irb unit 4 family inet address 10.14.1.15/24 virtual-gateway-
address 10.14.1.11

```

- e. The following is the IRB gateway configuration for the VLAN-5 on Host (which is the host part of VLAN-5):

```

[edit]

user@MX22# set interfaces irb unit 5 proxy-macip-advertisement
user@MX22# set interfaces irb unit 5 virtual-gateway-esi 00:22:ee:ee:ee:ee:ee:ee:ee
user@MX22# set interfaces irb unit 5 virtual-gateway-esi all-active
user@MX22# set interfaces irb unit 5 family inet address 10.15.1.15/24 virtual-gateway-
address 10.15.1.11

```

6. Configure trace operations to track all Layer 2 address learning and forwarding properties.

```

[edit]

user@MX22# set protocols l2-learning traceoptions file MX22-L2ALD.log
user@MX22# set protocols l2-learning traceoptions file size 10m
user@MX22# set protocols l2-learning traceoptions level all
user@MX22# set protocols l2-learning traceoptions flag all

```

7. Configure routing policies to accept the direct loopback address route and redirect it into BGP.

```

[edit]

user@MX22# set policy-options policy-statement L0 from protocol direct
user@MX22# set policy-options policy-statement L0 from route-filter 198.51.100.22/32 exact
user@MX22# set policy-options policy-statement L0 then accept
user@MX22# set policy-options policy-statement TEST1 term 1 from protocol bgp
user@MX22# set policy-options policy-statement TEST1 term 1 from external
user@MX22# set policy-options policy-statement TEST1 term 1 then reject

```

8. Configure community policy options.

```
[edit]

user@MX22# set policy-options community NO-EXPORT members no-advertise
user@MX22# set policy-options community NO-EXPORT members no-export
user@MX22# set policy-options community NO-EXPORT members no-export-subconfed
```

9. Apply load balance.

```
[edit]

user@MX22# set policy-options policy-statement TEST then community add NO-EXPORT
user@MX22# set policy-options policy-statement evpn-pplb from protocol evpn
user@MX22# set policy-options policy-statement evpn-pplb then load-balance per-packet
```

10. Configure an ESI value on the logical tunnel interface. Use the same ESI value on all other gateway/WAN edge routers in the DC2 network.

```
[edit]

user@MX22# set interfaces lt-1/0/0 esi 00:33:33:33:33:33:33:33:33
```

11. Configure active-active multihoming on the logical tunnel interface by including the all-active statement.

```
[edit]

user@MX22# set interfaces lt-1/0/0 esi all-active
```

12. Configure a pair of logical tunnel (lt-) interface on the MX22 gateway router to interconnect the EVPN-VXLAN instance of the data center network with the MPLS-based EVPN instance of the WAN. One logical tunnel (lt-) interface is configured as the access interface for EVPN-VXLAN and the other logical tunnel (lt-) interface is configured as the access interface for MPLS-based EVPN.

```
[edit]

user@MX22# set interfaces lt-1/0/0 unit 0 peer-unit 1
```

```

user@MX22# set interfaces lt-1/0/0 unit 0 family bridge interface-mode trunk
user@MX22# set interfaces lt-1/0/0 unit 0 family bridge vlan-id-list 1-5
user@MX22# set interfaces lt-1/0/0 unit 1 peer-unit 0
user@MX22# set interfaces lt-1/0/0 unit 1 family bridge interface-mode trunk
user@MX22# set interfaces lt-1/0/0 unit 1 family bridge vlan-id-list 1-5

```

13. Configure the loopback interface address and routing options.

```

[edit]

user@MX22# set interfaces lo0 unit 88 family inet address 198.51.100.22/32
user@MX22# set routing-options router-id 198.51.100.22
user@MX22# set routing-options autonomous-system 300

```

14. Configure load balancing on MX22.

```

[edit]

user@MX22# set routing-options forwarding-table export evpn-pplb

```

15. Enable RSVP, MPLS, BGP, and OSPF protocols on the core interfaces. Create MPLS LSPs and specify the address of the other gateway and WAN edge routers (MX11, MX12, P, MX21).

```

[edit]

user@MX22# set protocols rsvp interface all
user@MX22# set protocols rsvp interface fxp0.0 disable
user@MX22# set protocols mpls label-switched-path MX22-T0-MX11 to 192.0.2.21
user@MX22# set protocols mpls label-switched-path MX22-T0-MX12 to 192.0.2.22
user@MX22# set protocols mpls label-switched-path MX22-T0-P to 203.0.113.1
user@MX22# set protocols mpls label-switched-path MX22-T0-MX21 to 198.51.100.21
user@MX22# set protocols mpls interface all
user@MX22# set protocols mpls interface fxp0.0 disable
user@MX22# set protocols bgp local-address 198.51.100.22
user@MX22# set protocols bgp export TEST
user@MX22# set protocols bgp local-as 300
user@MX22# set protocols bgp group INT type internal
user@MX22# set protocols bgp group INT family evpn signaling
user@MX22# set protocols bgp group INT export TEST1
user@MX22# set protocols bgp group INT neighbor 203.0.113.1

```

```

user@MX22# set protocols ospf traffic-engineering
user@MX22# set protocols ospf area 0.0.0.0 interface ge-1/0/2.0
user@MX22# set protocols ospf area 0.0.0.0 interface lo0.88 passive

```

16. Configure EVPN-based MPLS routing instances on the MX22 router for each virtual network. Define the route distinguisher (used to identify and advertise EVPN-MPLS routes) and vrf-target (exports and tags all routes for that local VRF using the defined route target). Configure a bridge domain for each virtual router that maps VLAN IDs.

```

[edit]

user@MX22# set routing-instances EVPN-MPLS-1 instance-type virtual-switch
user@MX22# set routing-instances EVPN-MPLS-1 interface lt-1/0/0.0
user@MX22# set routing-instances EVPN-MPLS-1 route-distinguisher 198.51.100.22:100
user@MX22# set routing-instances EVPN-MPLS-1 vrf-target target:1:2
user@MX22# set routing-instances EVPN-MPLS-1 protocols evpn traceoptions file MX22-EVPN-
MPLS-1.log
user@MX22# set routing-instances EVPN-MPLS-1 protocols evpn traceoptions file size 10m
user@MX22# set routing-instances EVPN-MPLS-1 protocols evpn traceoptions flag all
user@MX22# set routing-instances EVPN-MPLS-1 protocols evpn extended-vlan-list 1-5
user@MX22# set routing-instances EVPN-MPLS-1 protocols evpn default-gateway no-gateway-
community
user@MX22# set routing-instances EVPN-MPLS-1 bridge-domains BD-1 domain-type bridge
user@MX22# set routing-instances EVPN-MPLS-1 bridge-domains BD-1 vlan-id 1
user@MX22# set routing-instances EVPN-MPLS-1 bridge-domains BD-2 domain-type bridge
user@MX22# set routing-instances EVPN-MPLS-1 bridge-domains BD-2 vlan-id 2
user@MX22# set routing-instances EVPN-MPLS-1 bridge-domains BD-3 domain-type bridge
user@MX22# set routing-instances EVPN-MPLS-1 bridge-domains BD-3 vlan-id 3
user@MX22# set routing-instances EVPN-MPLS-1 bridge-domains BD-4 domain-type bridge
user@MX22# set routing-instances EVPN-MPLS-1 bridge-domains BD-4 vlan-id 4
user@MX22# set routing-instances EVPN-MPLS-1 bridge-domains BD-5 domain-type bridge
user@MX22# set routing-instances EVPN-MPLS-1 bridge-domains BD-5 vlan-id 5

```

17. Configure EVPN-VXLAN routing instances on the MX22 router for each virtual network. Define the VTEP source interface, route distinguisher (used to identify and advertise EVPN routes), and vrf-target (exports and tags all routes for that local VRF using the defined route target). Configure the EVPN protocol, encapsulation method, VNI list, and BUM traffic forwarding method. Finally,

configure a bridge domain for each virtual router that maps VNIDs to VLAN IDs, and identify the BUM forwarding method.

[edit]

```

user@MX22# set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.88
user@MX22# set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
user@MX22# set routing-instances EVPN-VXLAN-1 interface lt-1/0/0.1
user@MX22# set routing-instances EVPN-VXLAN-1 route-distinguisher 198.51.100.22:1
user@MX22# set routing-instances EVPN-VXLAN-1 vrf-target target:1:3
user@MX22# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file MX22-EVPN-
VXLAN-1.log
user@MX22# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file size 10m
user@MX22# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions flag all
user@MX22# set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation vxlan
user@MX22# set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list 1-5
user@MX22# set routing-instances EVPN-VXLAN-1 protocols evpn default-gateway no-gateway-
community
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type bridge
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id 1
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 routing-interface irb.1
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni 1
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type bridge
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id 2
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 routing-interface irb.2
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni 2
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type bridge
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id 3
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 routing-interface irb.3
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni 3
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type bridge
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id 4
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 routing-interface irb.4
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni 4
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type bridge
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id 5
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 routing-interface irb.5
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni 5
user@MX22# set routing-instances VRF instance-type vrf
user@MX22# set routing-instances VRF interface irb.1
user@MX22# set routing-instances VRF interface irb.2
user@MX22# set routing-instances VRF interface irb.3

```



```

user@MX22# set routing-instances VRF interface irb.4
user@MX22# set routing-instances VRF interface irb.5
user@MX22# set routing-instances VRF route-distinguisher 1:1
user@MX22# set routing-instances VRF vrf-target target:10:10

```

Configuring ToR21

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

Configure the MX router as ToR21:

1. Set the system hostname.

```

[edit]
user@ToR21# set system host-name ToR21

```

2. Configure the interfaces and bridge domains on the CE4 device to enable Layer 2 connectivity.

```

[edit]

user@ce4# set logical-systems CE-4 interfaces ge-1/0/9 unit 0 description "CONNECTED TO Host 4"
user@ce4# set logical-systems CE-4 interfaces ge-1/0/9 unit 0 family bridge interface-mode trunk
user@ce4# set logical-systems CE-4 interfaces ge-1/0/9 unit 0 family bridge vlan-id-list 1-5
user@ce4# set logical-systems CE-4 interfaces ge-1/1/6 unit 0 description "CONNECTED TO ToR21"
user@ce4# set logical-systems CE-4 interfaces ge-1/1/6 unit 0 family bridge interface-mode trunk
user@ce4# set logical-systems CE-4 interfaces ge-1/1/6 unit 0 family bridge vlan-id-list 1-5
user@ce4# set logical-systems CE-4 bridge-domains BD-1 domain-type bridge
user@ce4# set logical-systems CE-4 bridge-domains BD-1 vlan-id 1
user@ce4# set logical-systems CE-4 bridge-domains BD-2 domain-type bridge
user@ce4# set logical-systems CE-4 bridge-domains BD-2 vlan-id 2
user@ce4# set logical-systems CE-4 bridge-domains BD-3 domain-type bridge
user@ce4# set logical-systems CE-4 bridge-domains BD-3 vlan-id 3
user@ce4# set logical-systems CE-4 bridge-domains BD-4 domain-type bridge

```

```

user@ce4# set logical-systems CE-4 bridge-domains BD-4 vlan-id 4
user@ce4# set logical-systems CE-4 bridge-domains BD-5 domain-type bridge
user@ce4# set logical-systems CE-4 bridge-domains BD-5 vlan-id 5

```

3. Configure trace options for the interfaces to enable trace logs.

```

[edit]

user@ce4# set interfaces traceoptions file R6-DCD.log
user@ce4# set interfaces traceoptions file size 10m
user@ce4# set interfaces traceoptions flag all

```

4. Set the number of aggregated Ethernet interfaces.

```

[edit]

user@ToR21# set chassis aggregated-devices ethernet device-count 1

```

5. Configure the interfaces on the ToR21 device to connect to the MX22, CE-5, CE-4, ToR22, and MX21 devices to enable underlay connectivity.

```

[edit]

user@ToR21# set interfaces xe-0/0/0 unit 0 description "CONNECTED TO MX22"
user@ToR21# set interfaces xe-0/0/0 unit 0 family inet address 192.168.10.2/24
user@ToR21# set interfaces ge-1/0/0 description "CONNECTED TO CE-5"
user@ToR21# set interfaces ge-1/0/0 gigether-options 802.3ad ae0
user@ToR21# set interfaces ge-1/0/1 unit 0 description "CONNECTED TO MX21"
user@ToR21# set interfaces ge-1/0/1 unit 0 family inet address 192.168.101.1/24
user@ToR21# set interfaces ge-1/0/6 unit 0 description "CONNECTED TO CE-4"
user@ToR21# set interfaces ge-1/0/6 unit 0 family bridge interface-mode trunk
user@ToR21# set interfaces ge-1/0/6 unit 0 family bridge vlan-id-list 1-5
user@ToR21# set interfaces ge-1/1/3 unit 0 description "CONNECTED TO ToR22"
user@ToR21# set interfaces ge-1/1/3 unit 0 family inet address 192.168.12.1/24

```

6. Configure a Link Aggregation Control Protocol (LACP)-enabled link aggregation group (LAG) interface towards the CE-5 end host device. The ESI value is globally unique across the entire EVPN

domain. The all-active configuration enables ToR21 and ToR22 to forward traffic to, and from the CE devices, such that all CE links are actively used.

```
[edit]

user@ToR21# set interfaces ae0 esi 00:44:44:44:44:44:44:44
user@ToR21# set interfaces ae0 esi all-active
user@ToR21# set interfaces ae0 aggregated-ether-options lacp active
user@ToR21# set interfaces ae0 aggregated-ether-options lacp periodic fast
user@ToR21# set interfaces ae0 aggregated-ether-options lacp system-id 22:22:22:22:22:22
user@ToR21# set interfaces ae0 unit 0 family bridge interface-mode trunk
user@ToR21# set interfaces ae0 unit 0 family bridge vlan-id-list 1-5
```

7. Configure the loopback interface address and routing options.

```
[edit]

user@ToR21# set interfaces lo0 unit 90 family inet address 198.51.100.11/32
user@ToR21# set routing-options router-id 198.51.100.11
user@ToR21# set routing-options autonomous-system 600
```

8. Configure load balancing on ToR21.

```
[edit]

user@ToR21# set routing-options forwarding-table export evpn-pplb
```

9. Configure a multiprotocol external BGP (MP-EBGP) underlay connectivity between the ToR (ToR21 and ToR22) and gateway routers (MX21 and MX22).

```
[edit]

user@ToR21# set protocols bgp export TEST
user@ToR21# set protocols bgp local-as 600
user@ToR21# set protocols bgp group MX21 type external
user@ToR21# set protocols bgp group MX21 local-address 192.168.9.2
user@ToR21# set protocols bgp group MX21 export L0
user@ToR21# set protocols bgp group MX21 export TEST
user@ToR21# set protocols bgp group MX21 peer-as 800
```

```

user@ToR21# set protocols bgp group MX21 local-as 600
user@ToR21# set protocols bgp group MX21 neighbor 192.168.9.1 family inet unicast
user@ToR21# set protocols bgp group MX22 type external
user@ToR21# set protocols bgp group MX22 local-address 10.102.2.1
user@ToR21# set protocols bgp group MX22 export L0
user@ToR21# set protocols bgp group MX22 export TEST
user@ToR21# set protocols bgp group MX22 peer-as 900
user@ToR21# set protocols bgp group MX22 local-as 600
user@ToR21# set protocols bgp group MX22 neighbor 192.168.10.1 family inet unicast
user@ToR21# set protocols bgp group ToR22 type external
user@ToR21# set protocols bgp group ToR22 local-address 10.105.5.1
user@ToR21# set protocols bgp group ToR22 export L0
user@ToR21# set protocols bgp group ToR22 export TEST
user@ToR21# set protocols bgp group ToR22 peer-as 700
user@ToR21# set protocols bgp group ToR22 local-as 600
user@ToR21# set protocols bgp group ToR22 neighbor 192.168.12.2 family inet unicast

```

10. Configure a multiprotocol external BGP (MP-EBGP) overlay between the ToR (ToR21 and ToR22) and gateway routers (MX21 and MX22) and set EVPN as the signaling protocol.

Step-by-Step Procedure

- a. Configure a MP-EBGP overlay to connect between ToR21 and MX21 using EVPN signaling.

[edit]

```

user@ToR21# set protocols bgp group MX21-EVPN type external
user@ToR21# set protocols bgp group MX21-EVPN multihop ttl 2
user@ToR21# set protocols bgp group MX21-EVPN multihop no-nexthop-change
user@ToR21# set protocols bgp group MX21-EVPN local-address 198.51.100.11
user@ToR21# set protocols bgp group MX21-EVPN peer-as 800
user@ToR21# set protocols bgp group MX21-EVPN local-as 600
user@ToR21# set protocols bgp group MX21-EVPN neighbor 198.51.100.21 family evpn
signaling

```

- b. Configure a MP-EBGP overlay to connect between ToR21 and MX22 using EVPN signaling.

[edit]

```

user@ToR21# set protocols bgp group MX22-EVPN type external

```

```

user@ToR21# set protocols bgp group MX22-EVPN multihop ttl 2
user@ToR21# set protocols bgp group MX22-EVPN multihop no-nexthop-change
user@ToR21# set protocols bgp group MX22-EVPN local-address 198.51.100.11
user@ToR21# set protocols bgp group MX22-EVPN peer-as 900
user@ToR21# set protocols bgp group MX22-EVPN local-as 600
user@ToR21# set protocols bgp group MX22-EVPN neighbor 198.51.100.22 family evpn
signaling

```

- c. Configure a MP-EBGP overlay to connect between ToR21 and ToR22 using EVPN signaling.

```

[edit]

user@ToR21# set protocols bgp group ToR22-EVPN type external
user@ToR21# set protocols bgp group ToR22-EVPN multihop ttl 2
user@ToR21# set protocols bgp group ToR22-EVPN multihop no-nexthop-change
user@ToR21# set protocols bgp group ToR22-EVPN local-address 198.51.100.11
user@ToR21# set protocols bgp group ToR22-EVPN peer-as 700
user@ToR21# set protocols bgp group ToR22-EVPN local-as 600
user@ToR21# set protocols bgp group ToR22-EVPN neighbor 198.51.100.12 family evpn
signaling

```

11. Configure trace operations to track all Layer 2 address learning and forwarding properties.

```

[edit]

user@ToR21# set protocols l2-learning traceoptions file TOR21-L2ALD.log
user@ToR21# set protocols l2-learning traceoptions file size 10m
user@ToR21# set protocols l2-learning traceoptions level all
user@ToR21# set protocols l2-learning traceoptions flag all

```

12. Configure routing policies to accept the direct loopback address route and redirect it into BGP.

```

[edit]

user@ToR21# set policy-options policy-statement L0 term 1 from protocol direct
user@ToR21# set policy-options policy-statement L0 term 1 from route-filter
198.51.100.11/32 exact
user@ToR21# set policy-options policy-statement L0 term 1 then accept

```

13. Configure community policy options.

```
[edit]

user@ToR21# set policy-options community NO-EXPORT members no-advertise
user@ToR21# set policy-options community NO-EXPORT members no-export
user@ToR21# set policy-options community NO-EXPORT members no-export-subconfed
```

14. Apply load balance.

```
[edit]

user@ToR21# set policy-options policy-statement TEST then community add NO-EXPORT
user@ToR21# set policy-options policy-statement evpn-pplb from protocol evpn
user@ToR21# set policy-options policy-statement evpn-pplb then load-balance per-packet
```

15. Configure EVPN routing instances for each virtual network. Define the VTEP source interface, route distinguisher (used to identify and advertise EVPN routes), and vrf-target (exports and tags all routes for that local VRF using the defined route target). Configure the EVPN protocol, encapsulation method, VNI list, and BUM traffic forwarding method. Finally, configure a bridge domain for each virtual router that maps VNIDs to VLAN IDs, and identify the BUM forwarding method.

```
[edit]

user@ToR21# set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.90
user@ToR21# set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
user@ToR21# set routing-instances EVPN-VXLAN-1 interface ge-1/0/6.0
user@ToR21# set routing-instances EVPN-VXLAN-1 interface ae0.0
user@ToR21# set routing-instances EVPN-VXLAN-1 route-distinguisher 198.51.100.11:1
user@ToR21# set routing-instances EVPN-VXLAN-1 vrf-target target:1:3
user@ToR21# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file TOR21-EVPN-
VXLAN-1.log
user@ToR21# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file size 10m
user@ToR21# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions flag all
user@ToR21# set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation vxlan
user@ToR21# set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list 1-5
user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type bridge
user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id 1
user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni 1
```

```

user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type bridge
user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id 2
user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni 2
user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type bridge
user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id 3
user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni 3
user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type bridge
user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id 4
user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni 4
user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type bridge
user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id 5
user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni 5

```

Configuring ToR22

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

Configure the MX router ToR22:

1. Set the system hostname.

```

[edit]
user@ToR22# set system host-name ToR22

```

2. Configure the interfaces and bridge domains on the CE-5 device to enable Layer 2 connectivity.

```

[edit]

user@ce5# set logical-systems CE-5 interfaces ge-1/0/9 unit 0 description "CONNECTED TO Host 5"
user@ce5# set logical-systems CE-5 interfaces ge-1/0/9 unit 0 family bridge interface-mode trunk
user@ce5# set logical-systems CE-5 interfaces ge-1/0/9 unit 0 family bridge vlan-id-list 1-5
user@ce5# set logical-systems CE-5 interfaces ae1 unit 0 description "CONNECTED TO ToR21"
user@ce5# set logical-systems CE-5 interfaces ae1 unit 0 family bridge interface-mode trunk
user@ce5# set logical-systems CE-5 interfaces ae1 unit 0 family bridge vlan-id-list 1-5
user@ce5# set logical-systems CE-5 bridge-domains BD-1 domain-type bridge

```

```

user@ce5# set logical-systems CE-5 bridge-domains BD-1 vlan-id 1
user@ce5# set logical-systems CE-5 bridge-domains BD-2 domain-type bridge
user@ce5# set logical-systems CE-5 bridge-domains BD-2 vlan-id 2
user@ce5# set logical-systems CE-5 bridge-domains BD-3 domain-type bridge
user@ce5# set logical-systems CE-5 bridge-domains BD-3 vlan-id 3
user@ce5# set logical-systems CE-5 bridge-domains BD-4 domain-type bridge
user@ce5# set logical-systems CE-5 bridge-domains BD-4 vlan-id 4
user@ce5# set logical-systems CE-5 bridge-domains BD-5 domain-type bridge
user@ce5# set logical-systems CE-5 bridge-domains BD-5 vlan-id 5

```

3. Configure the interfaces and bridge domains on the CE-3 device to enable Layer 2 connectivity.

```

[edit]

user@ce3# set logical-systems CE-3 interfaces ge-1/1/7 unit 0 description "CONNECTED TO
ToR12"
user@ce3# set logical-systems CE-3 interfaces ge-1/1/7 unit 0 family bridge interface-mode
trunk
user@ce3# set logical-systems CE-3 interfaces ge-1/1/7 unit 0 family bridge vlan-id-list 1-5
user@ce3# set logical-systems CE-3 interfaces ge-1/1/9 unit 0 description "CONNECTED TO
Host 3"
user@ce3# set logical-systems CE-3 interfaces ge-1/1/9 unit 0 family bridge interface-mode
trunk
user@ce3# set logical-systems CE-3 interfaces ge-1/1/9 unit 0 family bridge vlan-id-list 1-5
user@ce3# set logical-systems CE-3 bridge-domains BD-1 domain-type bridge
user@ce3# set logical-systems CE-3 bridge-domains BD-1 vlan-id 1
user@ce3# set logical-systems CE-3 bridge-domains BD-2 domain-type bridge
user@ce3# set logical-systems CE-3 bridge-domains BD-2 vlan-id 2
user@ce3# set logical-systems CE-3 bridge-domains BD-3 domain-type bridge
user@ce3# set logical-systems CE-3 bridge-domains BD-3 vlan-id 3
user@ce3# set logical-systems CE-3 bridge-domains BD-4 domain-type bridge
user@ce3# set logical-systems CE-3 bridge-domains BD-4 vlan-id 4
user@ce3# set logical-systems CE-3 bridge-domains BD-5 domain-type bridge
user@ce3# set logical-systems CE-3 bridge-domains BD-5 vlan-id 5

```

4. Configure trace options for the interfaces to enable trace logs.

```

[edit]

user@ce3# set interfaces traceoptions file R1-DCD.log

```



```

user@ce3# set interfaces traceoptions file size 10m
user@ce3# set interfaces traceoptions flag all

```

5. Set the number of aggregated Ethernet interfaces.

```

[edit]

user@ToR12# set chassis aggregated-devices ethernet device-count 2

```

6. Configure the interfaces on the ToR12 device to connect to the MX12, CE-2, CE-3, ToR11, and MX11 devices to enable underlay connectivity.

```

[edit]

user@ToR12# set interfaces ge-1/0/0 unit 0 description "CONNECTED TO MX11"
user@ToR12# set interfaces ge-1/0/0 unit 0 family inet address 192.168.6.1/24
user@ToR12# set interfaces ge-1/0/4 unit 0 description "CONNECTED TO MX12"
user@ToR12# set interfaces ge-1/0/4 unit 0 family inet address 192.168.5.1/24
user@ToR12# set interfaces ge-1/0/6 description "CONNECTED TO CE-1"
user@ToR12# set interfaces ge-1/0/6 gigether-options 802.3ad ae0
user@ToR12# set interfaces ge-1/0/7 unit 0 description "CONNECTED TO CE-3"
user@ToR12# set interfaces ge-1/0/7 unit 0 family bridge interface-mode trunk
user@ToR12# set interfaces ge-1/0/7 unit 0 family bridge vlan-id-list 1-5
user@ToR12# set interfaces ge-1/1/0 description "CONNECTED TO ToR11"
user@ToR12# set interfaces ge-1/1/0 gigether-options 802.3ad ae1
user@ToR12# set interfaces ge-1/1/3 unit 0 description "CONNECTED TO ToR11"
user@ToR12# set interfaces ge-1/1/3 unit 0 family inet address 192.168.2.2/24
user@ToR12# set interfaces ge-1/1/6 description "CONNECTED TO ToR12"
user@ToR12# set interfaces ge-1/1/6 gigether-options 802.3ad ae1

```

7. Configure a Link Aggregation Control Protocol (LACP)-enabled link aggregation group (LAG) interface towards the CE-1 end host device. The ESI value is globally unique across the entire EVPN domain. The all-active configuration enables ToR11 and ToR12 to forward traffic to, and from the CE devices, such that all CE links are actively used.

```

[edit]

user@ToR12# set interfaces ae0 esi 00:11:11:11:11:11:11:11:11
user@ToR12# set interfaces ae0 esi all-active
user@ToR12# set interfaces ae0 aggregated-ether-options lacp system-id 11:11:11:11:11:11

```

```

user@ToR12# set interfaces ae0 unit 0 family bridge interface-mode trunk
user@ToR12# set interfaces ae0 unit 0 family bridge vlan-id-list 1-5
user@ToR12# set interfaces ae1 aggregated-ether-options lacp active
user@ToR12# set interfaces ae1 aggregated-ether-options lacp periodic fast

```

8. Configure the loopback interface address and routing options.

```

[edit]

user@ToR12# set interfaces lo0 unit 82 family inet address 192.0.2.12/32
user@ToR12# set routing-options router-id 192.0.2.12
user@ToR12# set routing-options autonomous-system 200

```

9. Configure load balancing on ToR12.

```

[edit]

user@ToR12# set routing-options forwarding-table export evpn-pplb

```

10. Configure a multiprotocol external BGP (MP-EBGP) underlay connectivity between the ToR (ToR12 and ToR11) and gateway routers (MX11 and MX12).

```

[edit]

user@ToR12# set protocols bgp local-as 200
user@ToR12# set protocols bgp group MX11 type external
user@ToR12# set protocols bgp group MX11 local-address 192.168.6.1
user@ToR12# set protocols bgp group MX11 export L0
user@ToR12# set protocols bgp group MX11 export TEST
user@ToR12# set protocols bgp group MX11 peer-as 400
user@ToR12# set protocols bgp group MX11 local-as 200
user@ToR12# set protocols bgp group MX11 neighbor 192.168.6.2 family inet unicast
user@ToR12# set protocols bgp group MX12 type external
user@ToR12# set protocols bgp group MX12 local-address 192.168.5.1
user@ToR12# set protocols bgp group MX12 export L0
user@ToR12# set protocols bgp group MX12 export TEST
user@ToR12# set protocols bgp group MX12 peer-as 500
user@ToR12# set protocols bgp group MX12 local-as 200
user@ToR12# set protocols bgp group MX12 neighbor 192.168.5.2 family inet unicast
user@ToR12# set protocols bgp group ToR11 type external

```

```

user@ToR12# set protocols bgp group ToR11 local-address 192.168.2.2
user@ToR12# set protocols bgp group ToR11 export L0
user@ToR12# set protocols bgp group ToR11 export TEST
user@ToR12# set protocols bgp group ToR11 peer-as 100
user@ToR12# set protocols bgp group ToR11 local-as 200
user@ToR12# set protocols bgp group ToR11 neighbor 192.168.2.1 family inet unicast

```

11. Configure a multiprotocol external BGP (MP-EBGP) overlay between the ToR (ToR12 and ToR11) and gateway routers (MX11 and MX12) and set EVPN as the signaling protocol.

Step-by-Step Procedure

- a. Configure a MP-EBGP overlay to connect between ToR12 and MX11 using EVPN signaling.

```

[edit]

user@ToR12# set protocols bgp group MX11-EVPN type external
user@ToR12# set protocols bgp group MX11-EVPN multihop ttl 2
user@ToR12# set protocols bgp group MX11-EVPN multihop no-nexthop-change
user@ToR12# set protocols bgp group MX11-EVPN local-address 192.0.2.12
user@ToR12# set protocols bgp group MX11-EVPN export TEST
user@ToR12# set protocols bgp group MX11-EVPN peer-as 400
user@ToR12# set protocols bgp group MX11-EVPN local-as 200
user@ToR12# set protocols bgp group MX11-EVPN neighbor 192.0.2.21 family evpn signaling

```

- b. Configure a MP-EBGP overlay to connect between ToR12 and MX12 using EVPN signaling.

```

[edit]

user@ToR12# set protocols bgp group MX12-EVPN type external
user@ToR12# set protocols bgp group MX12-EVPN multihop ttl 2
user@ToR12# set protocols bgp group MX12-EVPN multihop no-nexthop-change
user@ToR12# set protocols bgp group MX12-EVPN local-address 192.0.2.12
user@ToR12# set protocols bgp group MX12-EVPN export TEST
user@ToR12# set protocols bgp group MX12-EVPN peer-as 500
user@ToR12# set protocols bgp group MX12-EVPN local-as 200
user@ToR12# set protocols bgp group MX12-EVPN neighbor 192.0.2.22 family evpn signaling

```

- c. Configure a MP-EBGP overlay to connect between ToR12 and ToR11 using EVPN signaling.

```
[edit]

user@ToR12# set protocols bgp group ToR11-EVPN type external
user@ToR12# set protocols bgp group ToR11-EVPN multihop ttl 2
user@ToR12# set protocols bgp group ToR11-EVPN multihop no-nexthop-change
user@ToR12# set protocols bgp group ToR11-EVPN local-address 192.0.2.12
user@ToR12# set protocols bgp group ToR11-EVPN export TEST
user@ToR12# set protocols bgp group ToR11-EVPN peer-as 100
user@ToR12# set protocols bgp group ToR11-EVPN local-as 200
user@ToR12# set protocols bgp group ToR11-EVPN neighbor 192.0.2.11 family evpn signaling
user@ToR12# set protocols bgp group ToR12-EVPN export TEST
```

12. Configure trace operations to track all Layer 2 address learning and forwarding properties.

```
[edit]

user@ToR12# set protocols l2-learning traceoptions file TOR12-L2ALD.log
user@ToR12# set protocols l2-learning traceoptions file size 10m
user@ToR12# set protocols l2-learning traceoptions level all
user@ToR12# set protocols l2-learning traceoptions flag all
```

13. Configure routing policies to accept the direct loopback address route and redirect it into BGP.

```
[edit]

user@ToR12# set policy-options policy-statement L0 term 1 from protocol direct
user@ToR12# set policy-options policy-statement L0 term 1 from route-filter 192.0.2.12/32
exact
user@ToR12# set policy-options policy-statement L0 term 1 then accept
```

14. Configure community policy options.

```
[edit]

user@ToR12# set policy-options community NO-EXPORT members no-advertise
```

```

user@ToR12# set policy-options community NO-EXPORT members no-export
user@ToR12# set policy-options community NO-EXPORT members no-export-subconfed

```

15. Apply load balance.

```

[edit]

user@ToR12# set policy-options policy-statement TEST then community add NO-EXPORT
user@ToR12# set policy-options policy-statement evpn-pplb from protocol evpn
user@ToR12# set policy-options policy-statement evpn-pplb then load-balance per-packet

```

16. Configure EVPN routing instances for each virtual network. Define the VTEP source interface, route distinguisher (used to identify and advertise EVPN routes), and vrf-target (exports and tags all routes for that local VRF using the defined route target). Configure the EVPN protocol, encapsulation method, VNI list, and BUM traffic forwarding method. Finally, configure a bridge domain for each virtual router that maps VNIDs to VLAN IDs, and identify the BUM forwarding method.

```

[edit]

user@ToR12# set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.82
user@ToR12# set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
user@ToR12# set routing-instances EVPN-VXLAN-1 interface ge-1/0/7.0
user@ToR12# set routing-instances EVPN-VXLAN-1 interface ae0.0
user@ToR12# set routing-instances EVPN-VXLAN-1 route-distinguisher 192.0.2.12:1
user@ToR12# set routing-instances EVPN-VXLAN-1 vrf-target target:1:1
user@ToR12# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file TOR12-EVPN-
VXLAN-1.log
user@ToR12# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file size 10m
user@ToR12# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions flag all
user@ToR12# set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation vxlan
user@ToR12# set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list 1-5
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type bridge
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id 1
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni 1
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type bridge
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id 2
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni 2
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type bridge
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id 3
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni 3

```

```

user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type bridge
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id 4
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni 4
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type bridge
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id 5
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni 5

```

Verification

IN THIS SECTION

- [Verifying ToR11 Configuration | 1275](#)
- [Verifying ToR12 Configuration | 1283](#)
- [Verifying Data Center Gateway and WAN Edge 1 Router \(MX11\) Configuration | 1293](#)
- [Verifying Data Center Gateway and WAN Edge 2 Router \(MX12\) Configuration | 1304](#)
- [Verifying Data Center Gateway and WAN Edge 3 Router \(MX21\) Configuration | 1316](#)
- [Verifying Data Center Gateway and WAN Edge 4 Router \(MX22\) Configuration | 1327](#)
- [Verifying ToR21 Configuration | 1338](#)
- [Verifying ToR22 Configuration | 1346](#)

After you configure both the underlay and EVPN overlay we recommend that you verify that the configurations work as you intended.

Verifying ToR11 Configuration

Purpose

Verify that ToR11 is properly configured.

Action

Verify that the logical system interfaces and bridge domains on the CE2 device are properly configured to enable Layer 2 connectivity.

```

user@ce2> show configuration logical-systems

```

```
CE-2 {
  interfaces {
    ge-1/0/9 {
      unit 0 {
        description "CONNECTED TO Host 2";
        family bridge {
          interface-mode trunk;
          vlan-id-list 1-5;
        }
      }
    }
    ge-1/1/6 {
      unit 0 {
        description "CONNECTED TO ToR11";
        family bridge {
          interface-mode trunk;
          vlan-id-list 1-5;
        }
      }
    }
  }
  bridge-domains {
    BD-1 {
      domain-type bridge;
      vlan-id 1;
    }
    BD-2 {
      domain-type bridge;
      vlan-id 2;
    }
    BD-3 {
      domain-type bridge;
      vlan-id 3;
    }
    BD-4 {
      domain-type bridge;
      vlan-id 4;
    }
  }
}
```

```

        BD-5 {
            domain-type bridge;
            vlan-id 5;
        }
    }
}

```

Verify that the interfaces and trace options on ToR11 are configured properly to enable underlay connectivity to other ToR and gateway and WAN edge devices.

```

user@ToR11>show configuration interfaces

```

```

traceoptions {
    file R0-DCD.log size 10m;
    flag all;
}
ge-1/0/5 {
    unit 0 {
        description "CONNECTED TO MX12";
        family inet {
            address 192.168.4.1/24;
        }
    }
}
ge-1/0/6 {
    unit 0 {
        description "CONNECTED TO CE-2";
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-5;
        }
    }
}
ge-1/1/0 {
    description "CONNECTED TO CE-1";
    gigether-options {
        802.3ad ae0;
    }
}
ge-1/1/1 {
    unit 0 {

```



```

        description "CONNECTED TO MX11";
        family inet {
            address 192.168.3.1/24;
        }
    }
}
ge-1/1/3 {
    unit 0 {
        description "CONNECTED TO ToR12";
        family inet {
            address 192.168.2.1/24;
        }
    }
}
ae0 {
    esi {
        00:11:11:11:11:11:11:11:11:11;
        all-active;
    }
    aggregated-ether-options {
        lacp {
            active;
            periodic fast;
            system-id 11:11:11:11:11:11;
        }
    }
    unit 0 {
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-5;
        }
    }
}
lo0 {
    unit 81 {
        family inet {
            address 192.0.2.11/32;
        }
    }
}
}

```

Verify that the routing and load balancing options are properly configured.

```
user@ToR11> show configuration routing-options

router-id 192.0.2.11;
autonomous-system 100;
forwarding-table {
    export evpn-pplb;
}
```

Verify that the multiprotocol external BGP (MP-EBGP) underlay and overlay protocols and Layer 2 address learning and forwarding properties are properly configured.

```
user@ToR11> show configuration protocols

bgp {
    local-as 100;
    group MX11 {
        type external;
        local-address 192.168.3.1;
        export [ LO TEST ];
        peer-as 400;
        neighbor 192.168.3.2 {
            family inet {
                unicast;
            }
        }
    }
    group MX12 {
        type external;
        local-address 192.168.4.1;
        export [ LO TEST ];
        peer-as 500;
        neighbor 192.168.4.2 {
            family inet {
                unicast;
            }
        }
    }
    group ToR12 {
        type external;
```

```

    local-address 192.168.2.1;
    export [ LO TEST ];
    peer-as 200;
    local-as 100;
    neighbor 192.168.2.2 {
        family inet {
            unicast;
        }
    }
}

group MX11-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 192.0.2.11;
    export TEST;
    peer-as 400;
    local-as 100;
    neighbor 192.0.2.21 {
        family evpn {
            signaling;
        }
    }
}

group MX12-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 192.0.2.11;
    export TEST;
    peer-as 500;
    local-as 100;
    neighbor 192.0.2.22 {
        family evpn {
            signaling;
        }
    }
}

group ToR12-EVPN {

```

```

        type external;
        multihop {
            ttl 2;
            no-nexthop-change;
        }
        local-address 192.0.2.11;
        export TEST;
        peer-as 200;
        local-as 100;
        neighbor 192.0.2.12 {
            family evpn {
                signaling;
            }
        }
    }
}
l2-learning {
    traceoptions {
        file TOR11-L2ALD.log size 10m;
        level all;
        flag all;
    }
}

```

Verify that the routing policies and community policy options and load balancing are properly configured to accept the direct loopback address route and redirect it into BGP.

```

user@ToR11> show configuration policy-options

policy-statement LO {
    term 1 {
        from {
            protocol direct;
            route-filter 192.0.2.11/32 exact;
        }
        then accept;
    }
}
policy-statement TEST {
    then {
        community add NO-EXPORT;
    }
}

```

```

}
policy-statement evpn-pplb {
    from protocol evpn;
    then {
        load-balance per-packet;
    }
}
community NO-EXPORT members [ no-advertise no-export no-export-subconfed ];

```

Verify that the EVPN-VXLAN routing instances for each virtual network are properly configured.

```

user@ToR11> show configuration routing-instances

EVPN-VXLAN-1 {
    vtep-source-interface lo0.81;
    instance-type virtual-switch;
    interface ge-1/0/6.0;
    interface ae0.0;
    route-distinguisher 192.0.2.11:1;
    vrf-target target:1:1;
    protocols {
        evpn {
            traceoptions {
                file TOR11-EVPN-VXLAN-1.log size 10m;
                flag all;
            }
            encapsulation vxlan;
            extended-vni-list 1-5;
        }
    }
}
bridge-domains {
    BD-1 {
        domain-type bridge;
        vlan-id 1;
        vxlan {
            vni 1;
        }
    }
    BD-2 {
        domain-type bridge;
        vlan-id 2;
        vxlan {

```

```

        vni 2;
    }
}
BD-3 {
    domain-type bridge;
    vlan-id 3;
    vxlan {
        vni 3;
    }
}
BD-4 {
    domain-type bridge;
    vlan-id 4;
    vxlan {
        vni 4;
    }
}
BD-5 {
    domain-type bridge;
    vlan-id 5;
    vxlan {
        vni 5;
    }
}
}
}
}

```

Verifying ToR12 Configuration

Purpose

Verify that ToR12 is properly configured.

Action

Verify that the logical system interfaces and bridge domains on the CE1 and CE3 devices are properly configured to enable Layer 2 connectivity.

```
user@ce1> show configuration logical-systems
```

```
CE-1 {
```

```

interfaces {
    ge-1/0/9 {
        unit 0 {
            description "CONNECTED TO Host 1";
            family bridge {
                interface-mode trunk;
                vlan-id-list 1-5;
            }
        }
    }
    ae1 {
        unit 0 {
            description "CONNECTED TO ToR12";
            family bridge {
                interface-mode trunk;
                vlan-id-list 1-5;
            }
        }
    }
}

bridge-domains {
    BD-1 {
        domain-type bridge;
        vlan-id 1;
    }

    BD-2 {
        domain-type bridge;
        vlan-id 2;
    }

    BD-3 {
        domain-type bridge;
        vlan-id 3;
    }

    BD-4 {
        domain-type bridge;
        vlan-id 4;
    }

    BD-5 {

```

```

        domain-type bridge;
        vlan-id 5;
    }
}

user@ce3> show configuration logical-systems

CE-3 {
  interfaces {
    ge-1/1/7 {
      unit 0 {
        description "CONNECTED TO ToR12";
        family bridge {
          interface-mode trunk;
          vlan-id-list 1-5;
        }
      }
    }
    ge-1/1/9 {
      unit 0 {
        description "CONNECTED TO Host 3";
        family bridge {
          interface-mode trunk;
          vlan-id-list 1-5;
        }
      }
    }
  }
  bridge-domains {
    BD-1 {
      domain-type bridge;
      vlan-id 1;
    }
    BD-2 {
      domain-type bridge;
      vlan-id 2;
    }
    BD-3 {

```



```

        domain-type bridge;
        vlan-id 3;

    }
    BD-4 {
        domain-type bridge;
        vlan-id 4;

    }
    BD-5 {
        domain-type bridge;
        vlan-id 5;

    }
}
}
}

```

Verify that the interfaces and trace options on ToR12 are configured properly to enable underlay connectivity to other ToR and gateway and WAN edge devices.

```

user@ToR12>show configuration interfaces

traceoptions {
    file R1-DCD.log size 10m;
    flag all;
}
ge-1/0/0 {
    unit 0 {
        description "CONNECTED TO MX11";
        family inet {
            address 192.168.6.1/24;
        }
    }
}
ge-1/0/4 {
    unit 0 {
        description "CONNECTED TO MX12";
        family inet {
            address 192.168.5.1/24;
        }
    }
}
}

```

```

ge-1/0/6 {
    description "CONNECTED TO CE-1";
    gigether-options {
        802.3ad ae0;
    }
}
ge-1/0/7 {
    unit 0 {
        description "CONNECTED TO CE-3";
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-5;
        }
    }
}
ge-1/1/0 {
    description "CONNECTED TO ToR11";
    gigether-options {
        802.3ad ae1;
    }
}
ge-1/1/3 {
    unit 0 {
        description "CONNECTED TO ToR11";
        family inet {
            address 192.168.2.2/24;
        }
    }
}
ge-1/1/6 {
    description "CONNECTED TO ToR12";
    gigether-options {
        802.3ad ae1;
    }
}
ae0 {
    esi {
        00:11:11:11:11:11:11:11:11:11;
        all-active;
    }
    aggregated-ether-options {
        lacp {
            system-id 11:11:11:11:11:11;
        }
    }
}

```

```

    }
  }
  unit 0 {
    family bridge {
      interface-mode trunk;
      vlan-id-list 1-5;
    }
  }
}
ae1 {
  aggregated-ether-options {
    lacp {
      active;
      periodic fast;
    }
  }
}
lo0 {
  unit 82 {
    family inet {
      address 192.0.2.12/32;
    }
  }
}
}

```

Verify that the routing and load balancing options are properly configured.

```

user@ToR12> show configuration routing-options

router-id 192.0.2.12;
autonomous-system 200;
forwarding-table {
  export evpn-pplb;
}

```

Verify that the multiprotocol external BGP (MP-EBGP) underlay and overlay protocols and Layer 2 address learning and forwarding properties are properly configured.

```

user@ToR12> show configuration protocols

bgp {

```

```

local-as 200;
group MX11 {
    type external;
    local-address 192.168.6.1;
    export [ LO TEST ];
    peer-as 400;
    local-as 200;
    neighbor 192.168.6.2 {
        family inet {
            unicast;
        }
    }
}
group MX12 {
    type external;
    local-address 192.168.5.1;
    export [ LO TEST ];
    peer-as 500;
    local-as 200;
    neighbor 192.168.5.2 {
        family inet {
            unicast;
        }
    }
}
group ToR11 {
    type external;
    local-address 192.168.2.2;
    export [ LO TEST ];
    peer-as 100;
    local-as 200;
    neighbor 192.168.2.1 {
        family inet {
            unicast;
        }
    }
}
group MX11-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
}

```

```

    local-address 192.0.2.12;
    export TEST;
    peer-as 400;
    local-as 200;
    neighbor 192.0.2.21 {
        family evpn {
            signaling;
        }
    }
}

group MX12-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 192.0.2.12;
    export TEST;
    peer-as 500;
    local-as 200;
    neighbor 192.0.2.22 {
        family evpn {
            signaling;
        }
    }
}

group ToR11-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 192.0.2.12;
    export TEST;
    peer-as 100;
    local-as 200;
    neighbor 192.0.2.11 {
        family evpn {
            signaling;
        }
    }
}

group ToR12-EVPN {

```

```

        export TEST;
    }
}
l2-learning {
    traceoptions {
        file TOR12-L2ALD.log size 10m;
        level all;
        flag all;
    }
}

```

Verify that the routing policies and community policy options and load balancing are properly configured to accept the direct loopback address route and redirect it into BGP.

```

user@ToR12> show configuration policy-options

policy-statement LO {
    term 1 {
        from {
            protocol direct;
            route-filter 192.0.2.12/32 exact;
        }
        then accept;
    }
}
policy-statement TEST {
    then {
        community add NO-EXPORT;
    }
}
policy-statement evpn-pplb {
    from protocol evpn;
    then {
        load-balance per-packet;
    }
}
community NO-EXPORT members [ no-advertise no-export no-export-subconfed ];

```

Verify that the EVPN-VXLAN routing instances for each virtual network are properly configured.

```

user@ToR12> show configuration routing-instances

EVPN-VXLAN-1 {
    vtep-source-interface lo0.82;
    instance-type virtual-switch;
    interface ge-1/0/7.0;
    interface ae0.0;
    route-distinguisher 192.0.2.12:1;
    vrf-target target:1:1;
    protocols {
        evpn {
            traceoptions {
                file TOR12-EVPN-VXLAN-1.log size 10m;
                flag all;
            }
            encapsulation vxlan;
            extended-vni-list 1-5;
        }
    }
}
bridge-domains {
    BD-1 {
        domain-type bridge;
        vlan-id 1;
        vxlan {
            vni 1;
        }
    }
    BD-2 {
        domain-type bridge;
        vlan-id 2;
        vxlan {
            vni 2;
        }
    }
    BD-3 {
        domain-type bridge;
        vlan-id 3;
        vxlan {
            vni 3;
        }
    }
}

```

```

    }
    BD-4 {
        domain-type bridge;
        vlan-id 4;
        vxlan {
            vni 4;
        }
    }
    BD-5 {
        domain-type bridge;
        vlan-id 5;
        vxlan {
            vni 5;
        }
    }
}
}
}

```

Verifying Data Center Gateway and WAN Edge 1 Router (MX11) Configuration

Purpose

Verify that MX11 is properly configured.

Action

Verify that the interfaces on the MX11 router (DC GW/WAN Edge1) are configured for the following:

Underlay connectivity to the MX12, ToR11, ToR12, and P devices, which is the EVPN-VXLAN part of DC1 network.

```

user@MX11> show configuration interfaces

traceoptions {
    file R2-DCD.log size 10m;
    flag all;
}
ge-5/1/0 {
    unit 0 {
        description "CONNECTED TO MX-21";
        family inet {

```



```

        address 192.168.7.1/24;
    }
}
ge-5/1/1 {
    unit 0 {
        description "CONNECTED TO TOR-11";
        family inet {
            address 192.168.3.2/24;
        }
    }
}
ge-5/1/8 {
    unit 0 {
        description "CONNECTED TO TOR-12";
        family inet {
            address 192.168.6.2/24;
        }
    }
}
ge-5/1/9 {
    unit 0 {
        description "CONNECTED TO P";
        family inet {
            address 203.0.1.1/24;
        }
        family mpls;
    }
}

```

Integrated routing and bridging (IRB) interfaces that advertises the MAC and IP routes (MAC+IP type 2 routes) for hosts in the topology. The IRB configuration is the gateway for the VLANs on the hosts.

```

user@MX11> show configuration interfaces

irb {
    unit 1 {
        proxy-macip-advertisement;
        virtual-gateway-esi {
            00:11:aa:aa:aa:aa:aa:aa:aa;
            all-active;
        }
    }
}

```

```

    family inet {
        address 10.11.1.12/24 {
            virtual-gateway-address 10.11.1.10;
        }
    }
}
unit 2 {
    proxy-macip-advertisement;
    virtual-gateway-esi {
        00:11:bb:bb:bb:bb:bb:bb:bb:bb;
        all-active;
    }
    family inet {
        address 10.12.1.12/24 {
            virtual-gateway-address 10.12.1.10;
        }
    }
}
unit 3 {
    proxy-macip-advertisement;
    virtual-gateway-esi {
        00:11:cc:cc:cc:cc:cc:cc:cc:cc;
        all-active;
    }
    family inet {
        address 10.13.1.12/24 {
            virtual-gateway-address 10.13.1.10;
        }
    }
}
unit 4 {
    proxy-macip-advertisement;
    virtual-gateway-esi {
        00:11:dd:dd:dd:dd:dd:dd:dd:dd;
        all-active;
    }
    family inet {
        address 10.14.1.12/24 {
            virtual-gateway-address 10.14.1.10;
        }
    }
}
unit 5 {

```

```

    proxy-macip-advertisement;
    virtual-gateway-esi {
        00:11:ee:ee:ee:ee:ee:ee:ee;
        all-active;
    }
    family inet {
        address 10.15.1.12/24 {
            virtual-gateway-address 10.15.1.10;
        }
    }
}
}

```

An ESI value and active-active multihoming on the logical tunnel interface. Use the same ESI value on all other gateway/WAN edge routers in the DC1 network.

```

user@MX11> show configuration interfaces

lt-5/1/0 {
    esi {
        00:22:22:22:22:22:22:22:22;
        all-active;
    }
}

```

A pair of logical tunnel (lt-) interface on the MX11 gateway router to interconnect the EVPN-VXLAN instance of the data center network with the MPLS-based EVPN instance of the WAN. One logical tunnel (lt-) interface is configured as the access interface for EVPN-VXLAN and the other logical tunnel (lt-) interface is configured as the access interface for MPLS-based EVPN.

```

user@MX11> show configuration interfaces

lt-5/1/0 {
    unit 0 {
        peer-unit 1;
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-5;
        }
    }
    unit 1 {

```

```

        peer-unit 0;
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-5;
        }
    }
}

```

Loopback interface address.

```

user@MX11> show configuration interfaces

lo0 {
    unit 84 {
        family inet {
            address 192.0.2.21/32;
        }
        family mpls;
    }
}

```

Verify that the routing options and load balancing are properly configured.

```

user@MX11> show configuration routing-options

router-id 192.0.2.21;
autonomous-system 300;
forwarding-table {
    export evpn-pplb;
}

```

Verify that the multiprotocol external BGP (MP-EBGP) underlay and overlay protocols and RSVP, MPLS, BGP, and OSPF protocols are properly configured.

```

user@MX11> show configuration protocols

rsvp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}

```

```

}
mpls {
    label-switched-path MX11-T0-MX12 {
        to 192.0.2.22;
    }
    label-switched-path MX11-T0-P {
        to 203.0.113.1;
    }
    label-switched-path MX11-T0-MX21 {
        to 198.51.100.21;
    }
    label-switched-path MX11-T0-MX22 {
        to 198.51.100.22;
    }
    interface all;
    interface fxp0.0 {
        disable;
    }
}
bgp {
    local-address 192.0.2.21;
    local-as 300;
    group INT {
        type internal;
        local-address 192.0.2.21;
        family evpn {
            signaling;
        }
        export TEST;
        neighbor 203.0.113.1;
    }
    group TOR-11 {
        type external;
        local-address 192.168.3.2;
        import TEST;
        export [ TEST LO ];
        peer-as 100;
        local-as 400;
        neighbor 192.168.3.1 {
            family inet {
                unicast;
            }
        }
    }
}

```

```

}
group TOR-12 {
    type external;
    local-address 192.168.6.2;
    export [ TEST LO ];
    peer-as 200;
    local-as 400;
    neighbor 192.168.6.1 {
        family inet {
            unicast;
        }
    }
}
group MX-12 {
    type external;
    local-address 192.168.7.1;
    export [ TEST LO ];
    peer-as 500;
    local-as 400;
    neighbor 192.168.7.2 {
        family inet {
            unicast;
        }
    }
}
group TOR-11-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 192.0.2.21;
    export TEST;
    peer-as 100;
    local-as 400;
    neighbor 192.0.2.11 {
        family evpn {
            signaling;
        }
    }
}
group TOR-12-EVPN {
    type external;

```

```

        multihop {
            ttl 2;
            no-nexthop-change;
        }
        local-address 192.0.2.21;
        export TEST;
        peer-as 200;
        local-as 400;
        neighbor 192.0.2.12 {
            family evpn {
                signaling;
            }
        }
    }
}
group MX-12-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 192.0.2.21;
    export TEST;
    peer-as 500;
    local-as 400;
    neighbor 192.0.2.22 {
        family evpn {
            signaling;
        }
    }
}
group MX-11-EVPN {
    export TEST;
}
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface ge-5/1/9.0;
        interface lo0.84 {
            passive;
        }
    }
}
}

```

```

l2-learning {
    traceoptions {
        file MX11-L2ALD.log size 10m;
        level all;
        flag all;
    }
}

```

Verify that the routing policies and community policy options and load balancing are properly configured to accept the direct loopback address route and redirect it into BGP.

```

user@MX11> show configuration policy-options

policy-statement LO {
    term 1 {
        from {
            protocol direct;
            route-filter 192.0.2.21/32 exact;
        }
        then accept;
    }
    from {
        protocol direct;
        route-filter 192.0.2.21/32 exact;
    }
    then accept;
}
policy-statement TEST {
    then {
        community add NO-EXPORT;
    }
}
policy-statement evpn-pplb {
    from protocol evpn;
    then {
        load-balance per-packet;
    }
}
community NO-EXPORT members [ no-advertise no-export no-export-subconfed ];

```


Verify that the EVPN-based MPLS routing instances and EVPN-VXLAN routing instances are properly configured.

```

user@MX11> show configuration routing-instances

EVPN-MPLS-1 {
    instance-type virtual-switch;
    interface lt-5/1/0.0;
    route-distinguisher 192.0.2.21:100;
    vrf-target target:1:2;
    protocols {
        evpn {
            traceoptions {
                file MX11-EVPN-MPLS-1.log size 10m;
                flag all;
            }
            extended-vlan-list 1-5;
            default-gateway no-gateway-community;
        }
    }
    bridge-domains {
        BD-1 {
            domain-type bridge;
            vlan-id 1;
        }
        BD-2 {
            domain-type bridge;
            vlan-id 2;
        }
        BD-3 {
            domain-type bridge;
            vlan-id 3;
        }
        BD-4 {
            domain-type bridge;
            vlan-id 4;
        }
        BD-5 {
            domain-type bridge;
            vlan-id 5;
        }
    }
}

```

```

}
EVPN-VXLAN-1 {
    vtep-source-interface lo0.84;
    instance-type virtual-switch;
    interface lt-5/1/0.1;
    route-distinguisher 192.0.2.21:1;
    vrf-target target:1:1;
    protocols {
        evpn {
            traceoptions {
                file MX11-EVPN-VXLAN-1.log size 10m;
                flag all;
            }
            encapsulation vxlan;
            extended-vni-list 1-5;
            default-gateway no-gateway-community;
        }
    }
}
bridge-domains {
    BD-1 {
        domain-type bridge;
        vlan-id 1;
        routing-interface irb.1;
        vxlan {
            vni 1;
        }
    }
    BD-2 {
        domain-type bridge;
        vlan-id 2;
        routing-interface irb.2;
        vxlan {
            vni 2;
        }
    }
    BD-3 {
        domain-type bridge;
        vlan-id 3;
        routing-interface irb.3;
        vxlan {
            vni 3;
        }
    }
}

```

```

    BD-4 {
        domain-type bridge;
        vlan-id 4;
        routing-interface irb.4;
        vxlan {
            vni 4;
        }
    }
    BD-5 {
        domain-type bridge;
        vlan-id 5;
        routing-interface irb.5;
        vxlan {
            vni 5;
        }
    }
}
VRF {
    instance-type vrf;
    interface irb.1;
    interface irb.2;
    interface irb.3;
    interface irb.4;
    interface irb.5;
    route-distinguisher 1:1;
    vrf-target target:10:10;
}

```

Verifying Data Center Gateway and WAN Edge 2 Router (MX12) Configuration

Purpose

Verify that MX12 is properly configured.

Action

Verify that the interfaces on the MX11 router (DC GW/WAN Edge1) are configured for the following:

Underlay connectivity to the MX12, ToR11, ToR12, and P devices, which is the EVPN-VXLAN part of DC1 network.

```
user@MX11> show configuration interfaces

traceoptions {
    file R2-DCD.log size 10m;
    flag all;
}
ge-5/1/0 {
    unit 0 {
        description "CONNECTED TO MX-21";
        family inet {
            address 192.168.7.1/24;
        }
    }
}
ge-5/1/1 {
    unit 0 {
        description "CONNECTED TO TOR-11";
        family inet {
            address 192.168.3.2/24;
        }
    }
}
ge-5/1/8 {
    unit 0 {
        description "CONNECTED TO TOR-12";
        family inet {
            address 192.168.6.2/24;
        }
    }
}
ge-5/1/9 {
    unit 0 {
        description "CONNECTED TO P";
        family inet {
            address 203.0.1.1/24;
        }
        family mpls;
    }
}
```

```

    }
}

```

Integrated routing and bridging (IRB) interfaces that advertises the MAC and IP routes (MAC+IP type 2 routes) for hosts in the topology. The IRB configuration is the gateway for the VLANs on the hosts.

```
user@MX11> show configuration interfaces
```

```

irb {
  unit 1 {
    proxy-macip-advertisement;
    virtual-gateway-esi {
      00:11:aa:aa:aa:aa:aa:aa:aa:aa;
      all-active;
    }
    family inet {
      address 10.11.1.12/24 {
        virtual-gateway-address 10.11.1.10;
      }
    }
  }
  unit 2 {
    proxy-macip-advertisement;
    virtual-gateway-esi {
      00:11:bb:bb:bb:bb:bb:bb:bb:bb;
      all-active;
    }
    family inet {
      address 10.12.1.12/24 {
        virtual-gateway-address 10.12.1.10;
      }
    }
  }
  unit 3 {
    proxy-macip-advertisement;
    virtual-gateway-esi {
      00:11:cc:cc:cc:cc:cc:cc:cc:cc;
      all-active;
    }
    family inet {
      address 10.13.1.12/24 {
        virtual-gateway-address 10.13.1.10;
      }
    }
  }
}

```

```

    }
  }
}
unit 4 {
  proxy-macip-advertisement;
  virtual-gateway-esi {
    00:11:dd:dd:dd:dd:dd:dd:dd;
    all-active;
  }
  family inet {
    address 10.14.1.12/24 {
      virtual-gateway-address 10.14.1.10;
    }
  }
}
unit 5 {
  proxy-macip-advertisement;
  virtual-gateway-esi {
    00:11:ee:ee:ee:ee:ee:ee:ee;
    all-active;
  }
  family inet {
    address 10.15.1.12/24 {
      virtual-gateway-address 10.15.1.10;
    }
  }
}
}
}

```

An ESI value and active-active multihoming on the logical tunnel interface. Use the same ESI value on all other gateway/WAN edge routers in the DC1 network.

```

user@MX11> show configuration interfaces

lt-5/1/0 {
  esi {
    00:22:22:22:22:22:22:22:22;
    all-active;
  }
}

```

A pair of logical tunnel (lt-) interface on the MX11 gateway router to interconnect the EVPN-VXLAN instance of the data center network with the MPLS-based EVPN instance of the WAN. One logical tunnel (lt-) interface is configured as the access interface for EVPN-VXLAN and the other logical tunnel (lt-) interface is configured as the access interface for MPLS-based EVPN.

```
user@MX11> show configuration interfaces
```

```
lt-5/1/0 {
  unit 0 {
    peer-unit 1;
    family bridge {
      interface-mode trunk;
      vlan-id-list 1-5;
    }
  }
  unit 1 {
    peer-unit 0;
    family bridge {
      interface-mode trunk;
      vlan-id-list 1-5;
    }
  }
}
```

Loopback interface address.

```
user@MX11> show configuration interfaces
```

```
lo0 {
  unit 84 {
    family inet {
      address 192.0.2.21/32;
    }
    family mpls;
  }
}
```

Verify that the routing options and load balancing are properly configured.

```
user@MX11> show configuration routing-options
```

```

router-id 192.0.2.21;
autonomous-system 300;
forwarding-table {
    export evpn-pplb;
}

```

Verify that the multiprotocol external BGP (MP-EBGP) underlay and overlay protocols and RSVP, MPLS, BGP, and OSPF protocols are properly configured.

```

user@MX11> show configuration protocols

```

```

rsvp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
mpls {
    label-switched-path MX11-T0-MX12 {
        to 192.0.2.22;
    }
    label-switched-path MX11-T0-P {
        to 203.0.113.1;
    }
    label-switched-path MX11-T0-MX21 {
        to 198.51.100.21;
    }
    label-switched-path MX11-T0-MX22 {
        to 198.51.100.22;
    }
    interface all;
    interface fxp0.0 {
        disable;
    }
}
bgp {
    local-address 192.0.2.21;
    local-as 300;
    group INT {
        type internal;
        local-address 192.0.2.21;
        family evpn {

```



```

        signaling;
    }
    export TEST;
    neighbor 203.0.113.1;
}
group TOR-11 {
    type external;
    local-address 192.168.3.2;
    import TEST;
    export [ TEST LO ];
    peer-as 100;
    local-as 400;
    neighbor 192.168.3.1 {
        family inet {
            unicast;
        }
    }
}
group TOR-12 {
    type external;
    local-address 192.168.6.2;
    export [ TEST LO ];
    peer-as 200;
    local-as 400;
    neighbor 192.168.6.1 {
        family inet {
            unicast;
        }
    }
}
group MX-12 {
    type external;
    local-address 192.168.7.1;
    export [ TEST LO ];
    peer-as 500;
    local-as 400;
    neighbor 192.168.7.2 {
        family inet {
            unicast;
        }
    }
}
group TOR-11-EVPN {

```

```

    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 192.0.2.21;
    export TEST;
    peer-as 100;
    local-as 400;
    neighbor 192.0.2.11 {
        family evpn {
            signaling;
        }
    }
}

group TOR-12-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 192.0.2.21;
    export TEST;
    peer-as 200;
    local-as 400;
    neighbor 192.0.2.12 {
        family evpn {
            signaling;
        }
    }
}

group MX-12-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 192.0.2.21;
    export TEST;
    peer-as 500;
    local-as 400;
    neighbor 192.0.2.22 {
        family evpn {

```

```

        signaling;
    }
}
}
group MX-11-EVPN {
    export TEST;
}
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface ge-5/1/9.0;
        interface lo0.84 {
            passive;
        }
    }
}
l2-learning {
    traceoptions {
        file MX11-L2ALD.log size 10m;
        level all;
        flag all;
    }
}
}

```

Verify that the routing policies and community policy options and load balancing are properly configured to accept the direct loopback address route and redirect it into BGP.

```

user@MX11> show configuration policy-options

policy-statement LO {
    term 1 {
        from {
            protocol direct;
            route-filter 192.0.2.21/32 exact;
        }
        then accept;
    }
    from {
        protocol direct;
        route-filter 192.0.2.21/32 exact;
    }
}

```

```

        then accept;
    }
    policy-statement TEST {
        then {
            community add NO-EXPORT;
        }
    }
    policy-statement evpn-pplb {
        from protocol evpn;
        then {
            load-balance per-packet;
        }
    }
    community NO-EXPORT members [ no-advertise no-export no-export-subconfed ];

```

Verify that the EVPN-based MPLS routing instances and EVPN-VXLAN routing instances are properly configured.

```

user@MX11> show configuration routing-instances

EVPN-MPLS-1 {
    instance-type virtual-switch;
    interface lt-5/1/0.0;
    route-distinguisher 192.0.2.21:100;
    vrf-target target:1:2;
    protocols {
        evpn {
            traceoptions {
                file MX11-EVPN-MPLS-1.log size 10m;
                flag all;
            }
            extended-vlan-list 1-5;
            default-gateway no-gateway-community;
        }
    }
    bridge-domains {
        BD-1 {
            domain-type bridge;
            vlan-id 1;
        }
        BD-2 {
            domain-type bridge;

```

```

        vlan-id 2;
    }
    BD-3 {
        domain-type bridge;
        vlan-id 3;
    }
    BD-4 {
        domain-type bridge;
        vlan-id 4;
    }
    BD-5 {
        domain-type bridge;
        vlan-id 5;
    }
}
}
EVPN-VXLAN-1 {
    vtep-source-interface lo0.84;
    instance-type virtual-switch;
    interface lt-5/1/0.1;
    route-distinguisher 192.0.2.21:1;
    vrf-target target:1:1;
    protocols {
        evpn {
            traceoptions {
                file MX11-EVPN-VXLAN-1.log size 10m;
                flag all;
            }
            encapsulation vxlan;
            extended-vni-list 1-5;
            default-gateway no-gateway-community;
        }
    }
    bridge-domains {
        BD-1 {
            domain-type bridge;
            vlan-id 1;
            routing-interface irb.1;
            vxlan {
                vni 1;
            }
        }
        BD-2 {

```

```

        domain-type bridge;
        vlan-id 2;
        routing-interface irb.2;
        vxlan {
            vni 2;
        }
    }
    BD-3 {
        domain-type bridge;
        vlan-id 3;
        routing-interface irb.3;
        vxlan {
            vni 3;
        }
    }
    BD-4 {
        domain-type bridge;
        vlan-id 4;
        routing-interface irb.4;
        vxlan {
            vni 4;
        }
    }
    BD-5 {
        domain-type bridge;
        vlan-id 5;
        routing-interface irb.5;
        vxlan {
            vni 5;
        }
    }
}
VRF {
    instance-type vrf;
    interface irb.1;
    interface irb.2;
    interface irb.3;
    interface irb.4;
    interface irb.5;
    route-distinguisher 1:1;

```

```
vrf-target target:10:10;
}
```

Verifying Data Center Gateway and WAN Edge 3 Router (MX21) Configuration

Purpose

Verify that MX21 is properly configured.

Action

Verify that the interfaces on the MX11 router (DC GW/WAN Edge1) are configured for the following:

Underlay connectivity to the MX12, ToR11, ToR12, and P devices, which is the EVPN-VXLAN part of DC1 network.

```
user@MX11> show configuration interfaces

traceoptions {
  file R2-DCD.log size 10m;
  flag all;
}
ge-5/1/0 {
  unit 0 {
    description "CONNECTED TO MX-21";
    family inet {
      address 192.168.7.1/24;
    }
  }
}
ge-5/1/1 {
  unit 0 {
    description "CONNECTED TO TOR-11";
    family inet {
      address 192.168.3.2/24;
    }
  }
}
ge-5/1/8 {
  unit 0 {
    description "CONNECTED TO TOR-12";
```

```

        family inet {
            address 192.168.6.2/24;
        }
    }
}
ge-5/1/9 {
    unit 0 {
        description "CONNECTED TO P";
        family inet {
            address 203.0.1.1/24;
        }
        family mpls;
    }
}

```

Integrated routing and bridging (IRB) interfaces that advertises the MAC and IP routes (MAC+IP type 2 routes) for hosts in the topology. The IRB configuration is the gateway for the VLANs on the hosts.

```

user@MX11> show configuration interfaces

irb {
    unit 1 {
        proxy-macip-advertisement;
        virtual-gateway-esi {
            00:11:aa:aa:aa:aa:aa:aa:aa:aa;
            all-active;
        }
        family inet {
            address 10.11.1.12/24 {
                virtual-gateway-address 10.11.1.10;
            }
        }
    }
    unit 2 {
        proxy-macip-advertisement;
        virtual-gateway-esi {
            00:11:bb:bb:bb:bb:bb:bb:bb:bb;
            all-active;
        }
        family inet {
            address 10.12.1.12/24 {
                virtual-gateway-address 10.12.1.10;
            }
        }
    }
}

```



```

    }
  }
}
unit 3 {
  proxy-macip-advertisement;
  virtual-gateway-esi {
    00:11:cc:cc:cc:cc:cc:cc:cc:cc;
    all-active;
  }
  family inet {
    address 10.13.1.12/24 {
      virtual-gateway-address 10.13.1.10;
    }
  }
}
unit 4 {
  proxy-macip-advertisement;
  virtual-gateway-esi {
    00:11:dd:dd:dd:dd:dd:dd:dd:dd;
    all-active;
  }
  family inet {
    address 10.14.1.12/24 {
      virtual-gateway-address 10.14.1.10;
    }
  }
}
unit 5 {
  proxy-macip-advertisement;
  virtual-gateway-esi {
    00:11:ee:ee:ee:ee:ee:ee:ee:ee;
    all-active;
  }
  family inet {
    address 10.15.1.12/24 {
      virtual-gateway-address 10.15.1.10;
    }
  }
}
}

```

An ESI value and active-active multihoming on the logical tunnel interface. Use the same ESI value on all other gateway/WAN edge routers in the DC1 network.

```
user@MX11> show configuration interfaces

lt-5/1/0 {
    esi {
        00:22:22:22:22:22:22:22:22:22;
        all-active;
    }
}
```

A pair of logical tunnel (lt-) interface on the MX11 gateway router to interconnect the EVPN-VXLAN instance of the data center network with the MPLS-based EVPN instance of the WAN. One logical tunnel (lt-) interface is configured as the access interface for EVPN-VXLAN and the other logical tunnel (lt-) interface is configured as the access interface for MPLS-based EVPN.

```
user@MX11> show configuration interfaces

lt-5/1/0 {
    unit 0 {
        peer-unit 1;
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-5;
        }
    }
    unit 1 {
        peer-unit 0;
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-5;
        }
    }
}
```

Loopback interface address.

```
user@MX11> show configuration interfaces

lo0 {
```

```

    unit 84 {
        family inet {
            address 192.0.2.21/32;
        }
        family mpls;
    }
}

```

Verify that the routing options and load balancing are properly configured.

```

user@MX11> show configuration routing-options

router-id 192.0.2.21;
autonomous-system 300;
forwarding-table {
    export evpn-pplb;
}

```

Verify that the multiprotocol external BGP (MP-EBGP) underlay and overlay protocols and RSVP, MPLS, BGP, and OSPF protocols are properly configured.

```

user@MX11> show configuration protocols

rsvp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}

mpls {
    label-switched-path MX11-T0-MX12 {
        to 192.0.2.22;
    }
    label-switched-path MX11-T0-P {
        to 203.0.113.1;
    }
    label-switched-path MX11-T0-MX21 {
        to 198.51.100.21;
    }
    label-switched-path MX11-T0-MX22 {
        to 198.51.100.22;
    }
}

```

```

    }
    interface all;
    interface fxp0.0 {
        disable;
    }
}
bgp {
    local-address 192.0.2.21;
    local-as 300;
    group INT {
        type internal;
        local-address 192.0.2.21;
        family evpn {
            signaling;
        }
        export TEST;
        neighbor 203.0.113.1;
    }
    group TOR-11 {
        type external;
        local-address 192.168.3.2;
        import TEST;
        export [ TEST LO ];
        peer-as 100;
        local-as 400;
        neighbor 192.168.3.1 {
            family inet {
                unicast;
            }
        }
    }
}
group TOR-12 {
    type external;
    local-address 192.168.6.2;
    export [ TEST LO ];
    peer-as 200;
    local-as 400;
    neighbor 192.168.6.1 {
        family inet {
            unicast;
        }
    }
}
}

```

```

group MX-12 {
    type external;
    local-address 192.168.7.1;
    export [ TEST L0 ];
    peer-as 500;
    local-as 400;
    neighbor 192.168.7.2 {
        family inet {
            unicast;
        }
    }
}

group TOR-11-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 192.0.2.21;
    export TEST;
    peer-as 100;
    local-as 400;
    neighbor 192.0.2.11 {
        family evpn {
            signaling;
        }
    }
}

group TOR-12-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 192.0.2.21;
    export TEST;
    peer-as 200;
    local-as 400;
    neighbor 192.0.2.12 {
        family evpn {
            signaling;
        }
    }
}

```

```

}
group MX-12-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 192.0.2.21;
    export TEST;
    peer-as 500;
    local-as 400;
    neighbor 192.0.2.22 {
        family evpn {
            signaling;
        }
    }
}
group MX-11-EVPN {
    export TEST;
}
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface ge-5/1/9.0;
        interface lo0.84 {
            passive;
        }
    }
}
}
l2-learning {
    traceoptions {
        file MX11-L2ALD.log size 10m;
        level all;
        flag all;
    }
}
}

```

Verify that the routing policies and community policy options and load balancing are properly configured to accept the direct loopback address route and redirect it into BGP.

```
user@MX11> show configuration policy-options

policy-statement LO {
  term 1 {
    from {
      protocol direct;
      route-filter 192.0.2.21/32 exact;
    }
    then accept;
  }
  from {
    protocol direct;
    route-filter 192.0.2.21/32 exact;
  }
  then accept;
}
policy-statement TEST {
  then {
    community add NO-EXPORT;
  }
}
policy-statement evpn-pplb {
  from protocol evpn;
  then {
    load-balance per-packet;
  }
}
community NO-EXPORT members [ no-advertise no-export no-export-subconfed ];
```

Verify that the EVPN-based MPLS routing instances and EVPN-VXLAN routing instances are properly configured.

```
user@MX11> show configuration routing-instances

EVPN-MPLS-1 {
  instance-type virtual-switch;
  interface lt-5/1/0.0;
  route-distinguisher 192.0.2.21:100;
```

```

vrf-target target:1:2;
protocols {
    evpn {
        traceoptions {
            file MX11-EVPN-MPLS-1.log size 10m;
            flag all;
        }
        extended-vlan-list 1-5;
        default-gateway no-gateway-community;
    }
}
bridge-domains {
    BD-1 {
        domain-type bridge;
        vlan-id 1;
    }
    BD-2 {
        domain-type bridge;
        vlan-id 2;
    }
    BD-3 {
        domain-type bridge;
        vlan-id 3;
    }
    BD-4 {
        domain-type bridge;
        vlan-id 4;
    }
    BD-5 {
        domain-type bridge;
        vlan-id 5;
    }
}
}
EVPN-VXLAN-1 {
    vtep-source-interface lo0.84;
    instance-type virtual-switch;
    interface lt-5/1/0.1;
    route-distinguisher 192.0.2.21:1;
    vrf-target target:1:1;
    protocols {
        evpn {
            traceoptions {

```



```

        file MX11-EVPN-VXLAN-1.log size 10m;
        flag all;
    }
    encapsulation vxlan;
    extended-vni-list 1-5;
    default-gateway no-gateway-community;
}
}
bridge-domains {
    BD-1 {
        domain-type bridge;
        vlan-id 1;
        routing-interface irb.1;
        vxlan {
            vni 1;
        }
    }
    BD-2 {
        domain-type bridge;
        vlan-id 2;
        routing-interface irb.2;
        vxlan {
            vni 2;
        }
    }
    BD-3 {
        domain-type bridge;
        vlan-id 3;
        routing-interface irb.3;
        vxlan {
            vni 3;
        }
    }
    BD-4 {
        domain-type bridge;
        vlan-id 4;
        routing-interface irb.4;
        vxlan {
            vni 4;
        }
    }
    BD-5 {
        domain-type bridge;

```

```

        vlan-id 5;
        routing-interface irb.5;
        vxlan {
            vni 5;
        }
    }
}
VRF {
    instance-type vrf;
    interface irb.1;
    interface irb.2;
    interface irb.3;
    interface irb.4;
    interface irb.5;
    route-distinguisher 1:1;
    vrf-target target:10:10;
}

```

Verifying Data Center Gateway and WAN Edge 4 Router (MX22) Configuration

Purpose

Verify that MX22 is properly configured.

Action

Verify that the interfaces on the MX11 router (DC GW/WAN Edge1) are configured for the following:

Underlay connectivity to the MX12, ToR11, ToR12, and P devices, which is the EVPN-VXLAN part of DC1 network.

```

user@MX11> show configuration interfaces

traceoptions {
    file R2-DCD.log size 10m;
    flag all;
}
ge-5/1/0 {
    unit 0 {
        description "CONNECTED TO MX-21";
    }
}

```

```

        family inet {
            address 192.168.7.1/24;
        }
    }
}
ge-5/1/1 {
    unit 0 {
        description "CONNECTED TO TOR-11";
        family inet {
            address 192.168.3.2/24;
        }
    }
}
ge-5/1/8 {
    unit 0 {
        description "CONNECTED TO TOR-12";
        family inet {
            address 192.168.6.2/24;
        }
    }
}
ge-5/1/9 {
    unit 0 {
        description "CONNECTED TO P";
        family inet {
            address 203.0.1.1/24;
        }
        family mpls;
    }
}
}

```

Integrated routing and bridging (IRB) interfaces that advertises the MAC and IP routes (MAC+IP type 2 routes) for hosts in the topology. The IRB configuration is the gateway for the VLANs on the hosts.

```

user@MX11> show configuration interfaces

irb {
    unit 1 {
        proxy-macip-advertisement;
        virtual-gateway-esi {
            00:11:aa:aa:aa:aa:aa:aa:aa:aa;
            all-active;
        }
    }
}

```

```

    }
    family inet {
        address 10.11.1.12/24 {
            virtual-gateway-address 10.11.1.10;
        }
    }
}
unit 2 {
    proxy-macip-advertisement;
    virtual-gateway-esi {
        00:11:bb:bb:bb:bb:bb:bb:bb;
        all-active;
    }
    family inet {
        address 10.12.1.12/24 {
            virtual-gateway-address 10.12.1.10;
        }
    }
}
unit 3 {
    proxy-macip-advertisement;
    virtual-gateway-esi {
        00:11:cc:cc:cc:cc:cc:cc:cc;
        all-active;
    }
    family inet {
        address 10.13.1.12/24 {
            virtual-gateway-address 10.13.1.10;
        }
    }
}
unit 4 {
    proxy-macip-advertisement;
    virtual-gateway-esi {
        00:11:dd:dd:dd:dd:dd:dd:dd;
        all-active;
    }
    family inet {
        address 10.14.1.12/24 {
            virtual-gateway-address 10.14.1.10;
        }
    }
}
}

```

```

unit 5 {
    proxy-macip-advertisement;
    virtual-gateway-esi {
        00:11:ee:ee:ee:ee:ee:ee:ee;
        all-active;
    }
    family inet {
        address 10.15.1.12/24 {
            virtual-gateway-address 10.15.1.10;
        }
    }
}
}

```

An ESI value and active-active multihoming on the logical tunnel interface. Use the same ESI value on all other gateway/WAN edge routers in the DC1 network.

```

user@MX11> show configuration interfaces

```

```

lt-5/1/0 {
    esi {
        00:22:22:22:22:22:22:22:22;
        all-active;
    }
}

```

A pair of logical tunnel (lt-) interface on the MX11 gateway router to interconnect the EVPN-VXLAN instance of the data center network with the MPLS-based EVPN instance of the WAN. One logical tunnel (lt-) interface is configured as the access interface for EVPN-VXLAN and the other logical tunnel (lt-) interface is configured as the access interface for MPLS-based EVPN.

```

user@MX11> show configuration interfaces

```

```

lt-5/1/0 {
    unit 0 {
        peer-unit 1;
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-5;
        }
    }
}

```

```

    unit 1 {
        peer-unit 0;
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-5;
        }
    }
}

```

Loopback interface address.

```

user@MX11> show configuration interfaces

lo0 {
    unit 84 {
        family inet {
            address 192.0.2.21/32;
        }
        family mpls;
    }
}

```

Verify that the routing options and load balancing are properly configured.

```

user@MX11> show configuration routing-options

router-id 192.0.2.21;
autonomous-system 300;
forwarding-table {
    export evpn-pplb;
}

```

Verify that the multiprotocol external BGP (MP-EBGP) underlay and overlay protocols and RSVP, MPLS, BGP, and OSPF protocols are properly configured.

```

user@MX11> show configuration protocols

rsvp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}

```

```

    }
}
mpls {
    label-switched-path MX11-T0-MX12 {
        to 192.0.2.22;
    }
    label-switched-path MX11-T0-P {
        to 203.0.113.1;
    }
    label-switched-path MX11-T0-MX21 {
        to 198.51.100.21;
    }
    label-switched-path MX11-T0-MX22 {
        to 198.51.100.22;
    }
    interface all;
    interface fxp0.0 {
        disable;
    }
}
bgp {
    local-address 192.0.2.21;
    local-as 300;
    group INT {
        type internal;
        local-address 192.0.2.21;
        family evpn {
            signaling;
        }
        export TEST;
        neighbor 203.0.113.1;
    }
    group TOR-11 {
        type external;
        local-address 192.168.3.2;
        import TEST;
        export [ TEST LO ];
        peer-as 100;
        local-as 400;
        neighbor 192.168.3.1 {
            family inet {
                unicast;
            }
        }
    }
}

```

```

    }
}
group TOR-12 {
    type external;
    local-address 192.168.6.2;
    export [ TEST LO ];
    peer-as 200;
    local-as 400;
    neighbor 192.168.6.1 {
        family inet {
            unicast;
        }
    }
}
group MX-12 {
    type external;
    local-address 192.168.7.1;
    export [ TEST LO ];
    peer-as 500;
    local-as 400;
    neighbor 192.168.7.2 {
        family inet {
            unicast;
        }
    }
}
group TOR-11-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nextthop-change;
    }
    local-address 192.0.2.21;
    export TEST;
    peer-as 100;
    local-as 400;
    neighbor 192.0.2.11 {
        family evpn {
            signaling;
        }
    }
}
group TOR-12-EVPN {

```



```

    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 192.0.2.21;
    export TEST;
    peer-as 200;
    local-as 400;
    neighbor 192.0.2.12 {
        family evpn {
            signaling;
        }
    }
}
group MX-12-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 192.0.2.21;
    export TEST;
    peer-as 500;
    local-as 400;
    neighbor 192.0.2.22 {
        family evpn {
            signaling;
        }
    }
}
group MX-11-EVPN {
    export TEST;
}
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface ge-5/1/9.0;
        interface lo0.84 {
            passive;
        }
    }
}

```

```

}
l2-learning {
    traceoptions {
        file MX11-L2ALD.log size 10m;
        level all;
        flag all;
    }
}

```

Verify that the routing policies and community policy options and load balancing are properly configured to accept the direct loopback address route and redirect it into BGP.

```

user@MX11> show configuration policy-options

policy-statement LO {
    term 1 {
        from {
            protocol direct;
            route-filter 192.0.2.21/32 exact;
        }
        then accept;
    }
    from {
        protocol direct;
        route-filter 192.0.2.21/32 exact;
    }
    then accept;
}
policy-statement TEST {
    then {
        community add NO-EXPORT;
    }
}
policy-statement evpn-pplb {
    from protocol evpn;
    then {
        load-balance per-packet;
    }
}
community NO-EXPORT members [ no-advertise no-export no-export-subconfed ];

```

Verify that the EVPN-based MPLS routing instances and EVPN-VXLAN routing instances are properly configured.

```

user@MX11> show configuration routing-instances

EVPN-MPLS-1 {
    instance-type virtual-switch;
    interface lt-5/1/0.0;
    route-distinguisher 192.0.2.21:100;
    vrf-target target:1:2;
    protocols {
        evpn {
            traceoptions {
                file MX11-EVPN-MPLS-1.log size 10m;
                flag all;
            }
            extended-vlan-list 1-5;
            default-gateway no-gateway-community;
        }
    }
    bridge-domains {
        BD-1 {
            domain-type bridge;
            vlan-id 1;
        }
        BD-2 {
            domain-type bridge;
            vlan-id 2;
        }
        BD-3 {
            domain-type bridge;
            vlan-id 3;
        }
        BD-4 {
            domain-type bridge;
            vlan-id 4;
        }
        BD-5 {
            domain-type bridge;
            vlan-id 5;
        }
    }
}

```

```

}
EVPN-VXLAN-1 {
    vtep-source-interface lo0.84;
    instance-type virtual-switch;
    interface lt-5/1/0.1;
    route-distinguisher 192.0.2.21:1;
    vrf-target target:1:1;
    protocols {
        evpn {
            traceoptions {
                file MX11-EVPN-VXLAN-1.log size 10m;
                flag all;
            }
            encapsulation vxlan;
            extended-vni-list 1-5;
            default-gateway no-gateway-community;
        }
    }
}
bridge-domains {
    BD-1 {
        domain-type bridge;
        vlan-id 1;
        routing-interface irb.1;
        vxlan {
            vni 1;
        }
    }
    BD-2 {
        domain-type bridge;
        vlan-id 2;
        routing-interface irb.2;
        vxlan {
            vni 2;
        }
    }
    BD-3 {
        domain-type bridge;
        vlan-id 3;
        routing-interface irb.3;
        vxlan {
            vni 3;
        }
    }
}

```

```
    BD-4 {  
        domain-type bridge;  
        vlan-id 4;  
        routing-interface irb.4;  
        vxlan {  
            vni 4;  
        }  
    }  
    BD-5 {  
        domain-type bridge;  
        vlan-id 5;  
        routing-interface irb.5;  
        vxlan {  
            vni 5;  
        }  
    }  
}  
VRF {  
    instance-type vrf;  
    interface irb.1;  
    interface irb.2;  
    interface irb.3;  
    interface irb.4;  
    interface irb.5;  
    route-distinguisher 1:1;  
    vrf-target target:10:10;  
}
```

Verifying ToR21 Configuration

Purpose

Verify that ToR21 is properly configured.

Action

Verify that the logical system interfaces and bridge domains on the CE4 device are properly configured to enable Layer 2 connectivity and to handle inter-VXLAN traffic.

```
user@ce4> show configuration logical-systems

CE-4 {
  interfaces {
    ge-1/0/9 {
      unit 0 {
        description "CONNECTED TO Host 4";
        family bridge {
          interface-mode trunk;
          vlan-id-list 1-5;
        }
      }
    }
    ge-1/1/6 {
      unit 0 {
        description "CONNECTED TO ToR21";
        family bridge {
          interface-mode trunk;
          vlan-id-list 1-5;
        }
      }
    }
  }
  bridge-domains {
    BD-1 {
      domain-type bridge;
      vlan-id 1;
    }
    BD-2 {
      domain-type bridge;
      vlan-id 2;
    }
    BD-3 {
      domain-type bridge;
```

```

        vlan-id 3;

    }
    BD-4 {
        domain-type bridge;
        vlan-id 4;

    }
    BD-5 {
        domain-type bridge;
        vlan-id 5;

    }
}
}
}

```

Verify that the interfaces and trace options on ToR21 are configured properly to enable underlay connectivity to other ToR and gateway and WAN edge devices.

```

user@ToR21>show configuration interfaces

traceoptions {
    file R6-DCD.log size 10m;
    flag all;
}
xe-0/0/0 {
    unit 0 {
        description "CONNECTED TO MX22";
        family inet {
            address 192.168.10.2/24;
        }
    }
}
ge-1/0/0 {
    description "CONNECTED TO CE-5";
    gigether-options {
        802.3ad ae0;
    }
}
ge-1/0/1 {
    unit 0 {
        description "CONNECTED TO MX21";
    }
}

```

```

        family inet {
            address 192.168.101.1/24;
        }
    }
}
ge-1/0/6 {
    unit 0 {
        description "CONNECTED TO CE-4";
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-5;
        }
    }
}
ge-1/1/3 {
    unit 0 {
        description "CONNECTED TO ToR22";
        family inet {
            address 192.168.12.1/24;
        }
    }
}
ae0 {
    esi {
        00:44:44:44:44:44:44:44:44:44;
        all-active;
    }
    aggregated-ether-options {
        lacp {
            active;
            periodic fast;
            system-id 22:22:22:22:22:22;
        }
    }
    unit 0 {
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-5;
        }
    }
}
lo0 {
    unit 90 {

```



```

        family inet {
            address 198.51.100.11/32;
        }
    }
}

```

Verify that the routing and load balancing options are properly configured.

```

user@ToR21> show configuration routing-options

router-id 198.51.100.11;
autonomous-system 600;
forwarding-table {
    export evpn-pplb;
}

```

Verify that the multiprotocol external BGP (MP-EBGP) underlay and overlay protocols and Layer 2 address learning and forwarding properties are properly configured.

```

user@ToR21> show configuration protocols

bgp {
    export TEST;
    local-as 600;
    group MX21 {
        type external;
        local-address 192.168.9.2;
        export [ LO TEST ];
        peer-as 800;
        local-as 600;
        neighbor 192.168.9.1 {
            family inet {
                unicast;
            }
        }
    }
}
group MX22 {
    type external;
    local-address 10.102.2.1;
    export [ LO TEST ];
    peer-as 900;
}

```

```

    local-as 600;
    neighbor 192.168.10.1 {
        family inet {
            unicast;
        }
    }
}

group ToR22 {
    type external;
    local-address 10.105.5.1;
    export [ LO TEST ];
    peer-as 700;
    local-as 600;
    neighbor 192.168.12.2 {
        family inet {
            unicast;
        }
    }
}

group MX21-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 198.51.100.11;
    peer-as 800;
    local-as 600;
    neighbor 198.51.100.21 {
        family evpn {
            signaling;
        }
    }
}

group MX22-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 198.51.100.11;
    peer-as 900;
    local-as 600;

```

```

        neighbor 198.51.100.22 {
            family evpn {
                signaling;
            }
        }
    }
    group ToR22-EVPN {
        type external;
        multihop {
            ttl 2;
            no-nexthop-change;
        }
        local-address 198.51.100.11;
        peer-as 700;
        local-as 600;
        neighbor 198.51.100.12 {
            family evpn {
                signaling;
            }
        }
    }
}
l2-learning {
    traceoptions {
        file TOR21-L2ALD.log size 10m;
        level all;
        flag all;
    }
}

```

Verify that the routing policies and community policy options and load balancing are properly configured to accept the direct loopback address route and redirect it into BGP.

```

user@ToR21> show configuration policy-options

policy-statement LO {
    term 1 {
        from {
            protocol direct;
            route-filter 198.51.100.11/32 exact;
        }
        then accept;
    }
}

```

```

    }
}
policy-statement TEST {
    then {
        community add NO-EXPORT;
    }
}
policy-statement evpn-pplb {
    from protocol evpn;
    then {
        load-balance per-packet;
    }
}
community NO-EXPORT members [ no-advertise no-export no-export-subconfed ];

```

Verify that the EVPN-VXLAN routing instances for each virtual network are properly configured.

```

user@ToR21> show configuration routing-instances

EVPN-VXLAN-1 {
    vtep-source-interface lo0.90;
    instance-type virtual-switch;
    interface ge-1/0/6.0;
    interface ae0.0;
    route-distinguisher 198.51.100.11:1;
    vrf-target target:1:3;
    protocols {
        evpn {
            traceoptions {
                file TOR21-EVPN-VXLAN-1.log size 10m;
                flag all;
            }
            encapsulation vxlan;
            extended-vni-list 1-5;
        }
    }
    bridge-domains {
        BD-1 {
            domain-type bridge;
            vlan-id 1;
            vxlan {
                vni 1;
            }
        }
    }
}

```

```

    }
}
BD-2 {
    domain-type bridge;
    vlan-id 2;
    vxlan {
        vni 2;
    }
}
BD-3 {
    domain-type bridge;
    vlan-id 3;
    vxlan {
        vni 3;
    }
}
BD-4 {
    domain-type bridge;
    vlan-id 4;
    vxlan {
        vni 4;
    }
}
BD-5 {
    domain-type bridge;
    vlan-id 5;
    vxlan {
        vni 5;
    }
}
}
}
}

```

Verifying ToR22 Configuration

Purpose

Verify that ToR22 is properly configured.

Action

Verify that the logical system interfaces and bridge domains on the CE5 and CE6 devices are properly configured to enable Layer 2 connectivity and to handle inter-VXLAN traffic.

```
user@ce5> show configuration logical-systems

CE-5 {
  interfaces {
    ge-1/0/9 {
      unit 0 {
        description "CONNECTED TO Host 5";
        family bridge {
          interface-mode trunk;
          vlan-id-list 1-5;
        }
      }
    }
  }
  ae1 {
    unit 0 {
      description "CONNECTED TO ToR21";
      family bridge {
        interface-mode trunk;
        vlan-id-list 1-5;
      }
    }
  }
}

bridge-domains {
  BD-1 {
    domain-type bridge;
    vlan-id 1;
  }

  BD-2 {
    domain-type bridge;
    vlan-id 2;
  }

  BD-3 {
    domain-type bridge;
```

```

        vlan-id 3;

    }
    BD-4 {
        domain-type bridge;
        vlan-id 4;

    }
    BD-5 {
        domain-type bridge;
        vlan-id 5;

    }
}
}
user@ce6> show configuration logical-systems

CE-6 {
    interfaces {
        ge-1/1/6 {
            unit 0 {
                description "CONNECTED TO ToR22";
                family bridge {
                    interface-mode trunk;
                    vlan-id-list 1-5;
                }
            }
        }
        ge-1/1/9 {
            unit 0 {
                description "CONNECTED TO Host 6";
                family bridge {
                    interface-mode trunk;
                    vlan-id-list 1-5;
                }
            }
        }
    }
    bridge-domains {
        BD-1 {
            domain-type bridge;
            vlan-id 1;

        }
    }
}

```

```

    BD-2 {
        domain-type bridge;
        vlan-id 2;

    }
    BD-3 {
        domain-type bridge;
        vlan-id 3;

    }
    BD-4 {
        domain-type bridge;
        vlan-id 4;

    }
    BD-5 {
        domain-type bridge;
        vlan-id 5;

    }
}
}

```

Verify that the interfaces and trace options on ToR22 are configured properly to enable underlay connectivity to other ToR and gateway and WAN edge devices.

```

user@ToR22>show configuration interfaces

traceoptions {
    file R7-DCD.log size 10m;
    flag all;
}
xe-0/0/0 {
    unit 0 {
        description "CONNECTED TO MX22";
        family inet {
            address 192.168.11.2/24;
        }
    }
}
ge-1/0/0 {
    description "CONNECTED TO ToR21";
}

```



```

    gigether-options {
        802.3ad ae1;
    }
}
ge-1/0/6 {
    unit 0 {
        description "CONNECTED TO CE-6";
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-5;
        }
    }
}
ge-1/0/7 {
    description "CONNECTED TO ToR22";
    gigether-options {
        802.3ad ae1;
    }
}
ge-1/1/0 {
    unit 0 {
        description "CONNECTED TO MX21";
        family inet {
            address 192.168.8.2/24;
        }
    }
}
ge-1/1/3 {
    unit 0 {
        description "CONNECTED TO ToR21";
        family inet {
            address 192.168.12.2/24;
        }
    }
}
ge-1/1/7 {
    description "CONNECTED TO CE-5";
    gigether-options {
        802.3ad ae0;
    }
}
ae0 {
    esi {

```

```

    00:44:44:44:44:44:44:44:44;
    all-active;
}
aggregated-ether-options {
    lacp {
        active;
        periodic fast;
        system-id 22:22:22:22:22:22;
    }
}
unit 0 {
    family bridge {
        interface-mode trunk;
        vlan-id-list 1-5;
    }
}
}
ae1 {
    aggregated-ether-options {
        lacp {
            active;
            periodic fast;
            system-id 22:22:22:22:22:22;
        }
    }
}
lo0 {
    unit 92 {
        family inet {
            address 198.51.100.12/32;
        }
    }
}
}

```

Verify that the routing and load balancing options are properly configured.

```

user@ToR22> show configuration routing-options

```

```

router-id 198.51.100.12;
autonomous-system 700;
forwarding-table {

```

```

export evpn-pplb;
}

```

Verify that the multiprotocol external BGP (MP-EBGP) underlay and overlay protocols and Layer 2 address learning and forwarding properties are properly configured.

```

user@ToR22> show configuration protocols

```

```

bgp {
  export TEST;
  local-as 700;
  group MX21 {
    type external;
    local-address 192.168.8.2;
    export [ LO TEST ];
    peer-as 800;
    local-as 700;
    neighbor 192.168.8.1 {
      family inet {
        unicast;
      }
    }
  }
  group MX22 {
    type external;
    local-address 192.168.11.2;
    export [ LO TEST ];
    peer-as 900;
    local-as 700;
    neighbor 192.168.11.1 {
      family inet {
        unicast;
      }
    }
  }
  group ToR21 {
    type external;
    local-address 192.168.12.2;
    export [ LO TEST ];
    peer-as 600;
    local-as 700;
    neighbor 10.105.5.1 {

```

```

        family inet {
            unicast;
        }
    }
}

group MX21-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 198.51.100.12;
    peer-as 800;
    local-as 700;
    neighbor 198.51.100.21 {
        family evpn {
            signaling;
        }
    }
}

group MX22-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 198.51.100.12;
    peer-as 900;
    local-as 700;
    neighbor 198.51.100.22 {
        family evpn {
            signaling;
        }
    }
}

group ToR21-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 198.51.100.12;
    peer-as 600;
}

```

```

        local-as 700;
        neighbor 198.51.100.11 {
            family evpn {
                signaling;
            }
        }
    }
}
l2-learning {
    traceoptions {
        file TOR22-L2ALD.log size 10m;
        level all;
        flag all;
    }
}

```

Verify that the routing policies and community policy options and load balancing are properly configured to accept the direct loopback address route and redirect it into BGP.

```

user@Tor22> show configuration policy-options

policy-statement LO {
    term 1 {
        from {
            protocol direct;
            route-filter 198.51.100.12/32 exact;
        }
        then accept;
    }
}
policy-statement TEST {
    then {
        community add NO-EXPORT;
    }
}
policy-statement evpn-pplb {
    from protocol evpn;
    then {
        load-balance per-packet;
    }
}

```

```

}
community NO-EXPORT members [ no-advertise no-export no-export-subconfed ];

```

Verify that the EVPN-VXLAN routing instances for each virtual network are properly configured.

```

user@ToR22> show configuration routing-instances

EVPN-VXLAN-1 {
  vtep-source-interface lo0.92;
  instance-type virtual-switch;
  interface ge-1/0/6.0;
  interface ae0.0;
  route-distinguisher 198.51.100.12:1;
  vrf-target target:1:3;
  protocols {
    evpn {
      traceoptions {
        file TOR22-EVPN-VXLAN-1.log size 10m;
        flag all;
      }
      encapsulation vxlan;
      extended-vni-list 1-5;
    }
  }
}
bridge-domains {
  BD-1 {
    domain-type bridge;
    vlan-id 1;
    vxlan {
      vni 1;
    }
  }
  BD-2 {
    domain-type bridge;
    vlan-id 2;
    vxlan {
      vni 2;
    }
  }
  BD-3 {
    domain-type bridge;
    vlan-id 3;

```

```

        vxlan {
            vni 3;
        }
    }
    BD-4 {
        domain-type bridge;
        vlan-id 4;
        vxlan {
            vni 4;
        }
    }
    BD-5 {
        domain-type bridge;
        vlan-id 5;
        vxlan {
            vni 5;
        }
    }
}
}
}

```

RELATED DOCUMENTATION

| [EVPN-VXLAN Data Center Interconnect Through EVPN-MPLS WAN Overview](#) | 1163

Overview of EVPN-VXLAN Interconnects through EVPN MPLS-MPLS WAN Using Gateways

IN THIS SECTION

- [Configuration](#) | 1358

Configure seamless stitching between an EVPN-VXLAN data center, through an EVPN-MPLS fabric, to another EVPN-VXLAN data center, for interconnecting unicast and BUM traffic using WAN gateways with gateway-redundancy multihoming support. This feature is vlan-based, and includes vlan-bundle

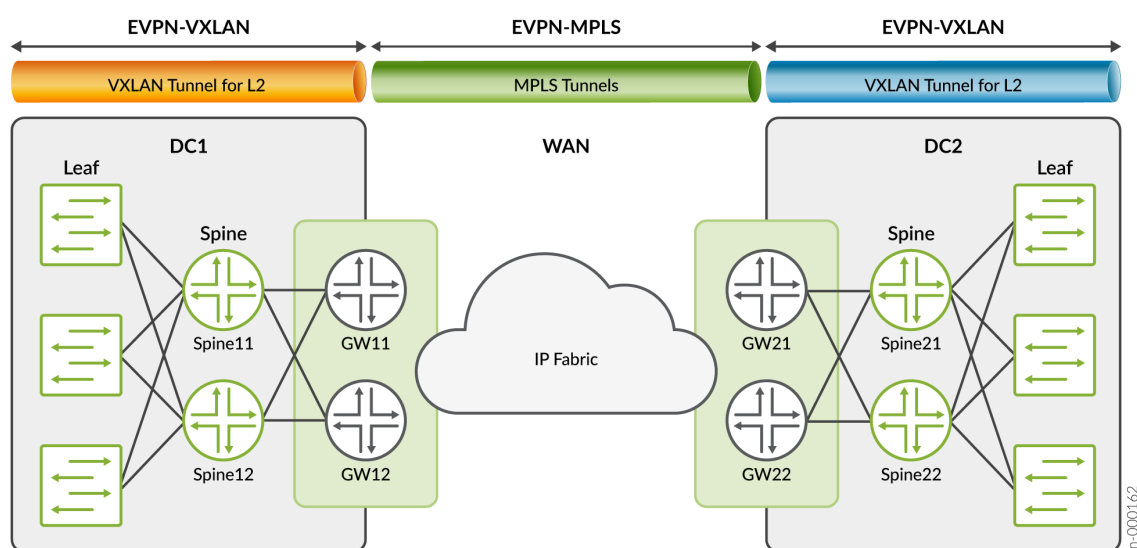
and vlan-aware support using a VLAN list on interconnected MX platform bridge domains for both AFT-based and Ukern-based line cards.

This feature updates use of the interconnect stanza:

```
interconnected-vlan-list [vlan-list];
encapsulation mpls
```

See also the Configuration example below.

Figure 136: EVPN-VXLAN DC to EVPN-MPLS to EVPN-VXLAN DC



Seamless stitching of EVPN-VXLAN to EVPN-MPLS is achieved by configuring both encapsulations, one for EVPN-VXLAN encapsulation going to the data center (DC) side, and EVPN-MPLS encapsulation on the WAN (DCI) side (under interconnect stanza).

BUM traffic coming from the data center is flooded to DCI peers by pushing the IM label received from remote peers. BUM traffic coming from the WAN side is flooded to all DC VTEPs after popping the IM label.

One of the multihoming GW nodes is elected as the designated forward (DF) based on I-ESI (interconnect and DF GW will forward BUM traffic in either direction). To void the traffic duplication, Non-DF drops the BUM traffic coming from DCI or the DC side. When there are multihoming CEs attached to gateway nodes, the current local bias filters handle split horizon functionality.

The unicast traffic from the DC side will be load balanced to remote DCI peers, and unicast traffic from WAN is load balanced to DC VTEPs within the data center.

Configuration

This sample configuration defines which VLANs to interconnect from the EVPN-VXLAN domain to the EVPN-MPLS domain.

```
interconnect {
    vrf-target target:2:2;
    route-distinguisher 100:120;
    esi 00:0a:0b:0c:0d:0a:0b:0c:0d:0a; {
        all-active;
    }
    interconnected-vlan-list [ 51 52 ];
    encapsulation mpls;
}
traceoptions {
    file evpn-vxlan-gw12.log size 100m;
    flag all;
}
}
```

RELATED DOCUMENTATION

| [Implementing EVPN-VXLAN for Data Centers](#)

Extending a Junos Fusion Enterprise Using EVPN-MPLS

IN THIS CHAPTER

- [Understanding EVPN-MPLS Interworking with Junos Fusion Enterprise and MC-LAG | 1359](#)
- [Example: EVPN-MPLS Interworking With Junos Fusion Enterprise | 1366](#)
- [Example: EVPN-MPLS Interworking With an MC-LAG Topology | 1387](#)

Understanding EVPN-MPLS Interworking with Junos Fusion Enterprise and MC-LAG

IN THIS SECTION

- [Benefits of Using EVPN-MPLS with Junos Fusion Enterprise and MC-LAG | 1362](#)
- [BUM Traffic Handling | 1362](#)
- [Split Horizon | 1363](#)
- [MAC Learning | 1364](#)
- [Handling Down Link Between Cascade and Uplink Ports in Junos Fusion Enterprise | 1365](#)
- [Layer 3 Gateway Support | 1366](#)

Starting with Junos OS Release 17.4R1, you can use Ethernet VPN (EVPN) to extend a Junos Fusion Enterprise or multichassis link aggregation group (MC-LAG) network over an MPLS network to a data center or campus network. With the introduction of this feature, you can now interconnect dispersed campus and data center sites to form a single Layer 2 virtual bridge.

[Figure 137 on page 1360](#) shows a Junos Fusion Enterprise topology with two EX9200 switches that serve as aggregation devices (PE2 and PE3) to which the satellite devices are multihomed. The two

aggregation devices use an interchassis link (ICL) and the Inter-Chassis Control Protocol (ICCP) protocol from MC-LAG to connect and maintain the Junos Fusion Enterprise topology. PE1 in the EVPN-MPLS environment interworks with PE2 and PE3 in the Junos Fusion Enterprise with MC-LAG.

Figure 137: EVPN-MPLS Interworking with Junos Fusion Enterprise

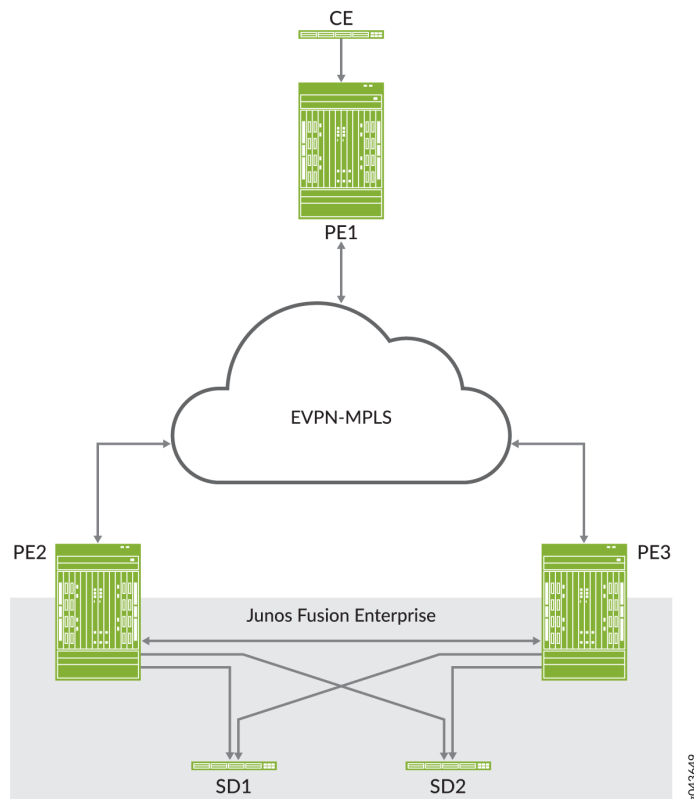
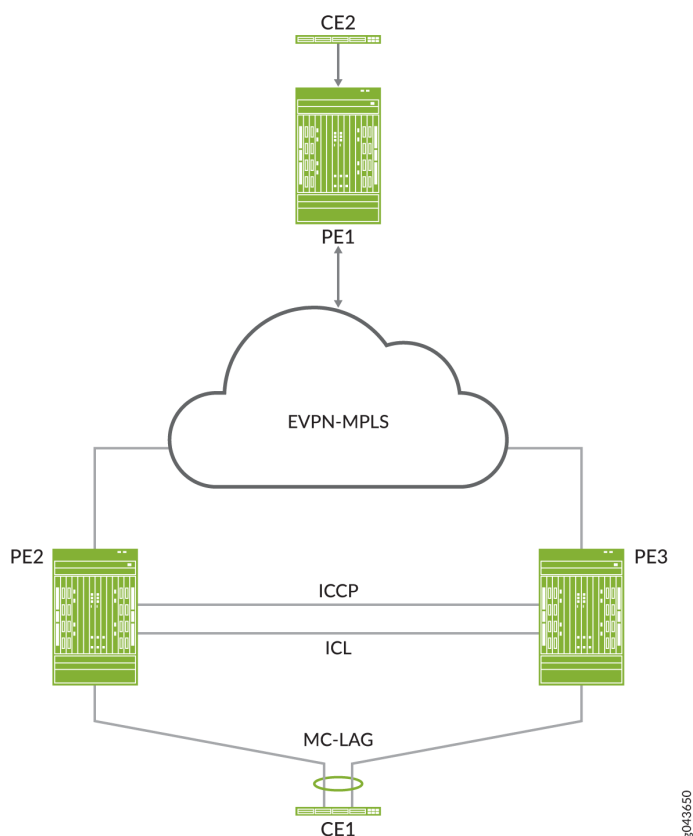


Figure 138 on page 1361 shows an MC-LAG topology in which customer edge (CE) device CE1 is multihomed to PE2 and PE3. PE2 and PE3 use an ICL and the ICCP protocol from MC-LAG to connect

and maintain the topology. PE1 in the EVPN-MPLS environment interworks with PE2 and PE3 in the MC-LAG environment.

Figure 138: EVPN-MPLS Interworking with MC-LAG



Throughout this topic, [Figure 137 on page 1360](#) and [Figure 138 on page 1361](#) serve as references to illustrate various scenarios and points.

The use cases depicted in [Figure 137 on page 1360](#) and [Figure 138 on page 1361](#) require the configuration of both EVPN multihoming in active-active mode and MC-LAG on PE2 and PE3. EVPN with multihoming active-active and MC-LAG have their own forwarding logic for handling traffic, in particular, broadcast, unknown unicast, and multicast (BUM) traffic. At times, the forwarding logic for EVPN with multihoming active-active and MC-LAG contradict each other and causes issues. This topic describes the issues and how the EVPN-MPLS interworking feature resolves these issues.

NOTE:

Other than the EVPN-MPLS interworking-specific implementations described in this topic, EVPN-MPLS, Junos Fusion Enterprise, and MC-LAG offer the same functionality and function the same as the standalone features.

Benefits of Using EVPN-MPLS with Junos Fusion Enterprise and MC-LAG

Use EVPN-MPLS with Junos Fusion Enterprise and MC-LAG to interconnect dispersed campus and data center sites to form a single Layer 2 virtual bridge.

BUM Traffic Handling

In the use cases shown in [Figure 137 on page 1360](#) and [Figure 138 on page 1361](#), PE1, PE2, and PE3 are EVPN peers, and PE2 and PE3 are MC-LAG peers. Both sets of peers exchange control information and forward traffic to each other, which causes issues. [Table 42 on page 1362](#) outlines the issues that arise, and how Juniper Networks resolves the issues in its implementation of the EVPN-MPLS interworking feature.

Table 42: BUM Traffic: Issues and Resolutions

BUM Traffic Direction	EVPN Interworking with Junos Fusion Enterprise and MC-LAG Logic	Issue	Juniper Networks Implementation Approach
North bound (PE2 receives BUM packet from a locally attached single- or dual-homed interfaces).	PE2 floods BUM packet to the following: <ul style="list-style-type: none"> All locally attached interfaces, including the ICL, for a particular broadcast domain. All remote EVPN peers for which PE2 has received inclusive multicast routes. 	Between PE2 and PE3, there are two BUM forwarding paths—the MC-LAG ICL and an EVPN-MPLS path. The multiple forwarding paths result in packet duplication and loops.	<ul style="list-style-type: none"> BUM traffic is forwarded on the ICL only. Incoming traffic from the EVPN core is not forwarded on the ICL. Incoming traffic from the ICL is not forwarded to the EVPN core.

Table 42: BUM Traffic: Issues and Resolutions *(Continued)*

BUM Traffic Direction	EVPN Interworking with Junos Fusion Enterprise and MC-LAG Logic	Issue	Juniper Networks Implementation Approach
South bound (PE1 forwards BUM packet to PE2 and PE3).	PE2 and PE3 both receive a copy of the BUM packet and flood the packet out of all of their local interfaces, including the ICL.	PE2 and PE3 both forward the BUM packet out of the ICL, which results in packet duplication and loops.	

Split Horizon

In the use cases shown in [Figure 137 on page 1360](#) and [Figure 138 on page 1361](#), split horizon prevents multiple copies of a BUM packet from being forwarded to a CE device (satellite device). However, the EVPN-MPLS and MC-LAG split horizon implementations contradict each other, which causes an issue. [Table 43 on page 1364](#) explains the issue and how Juniper Networks resolves it in its implementation of the EVPN-MPLS interworking feature.

Table 43: BUM Traffic: Split Horizon-Related Issue and Resolution

BUM Traffic Direction	EVPN Interworking with Junos Fusion Enterprise and MC-LAG Logic	Issue	Juniper Networks Implementation Approach
North bound (PE2 receives BUM packet from a locally attached dual-homed interface).	<ul style="list-style-type: none"> • Per EVPN-MPLS forwarding logic: <ul style="list-style-type: none"> • Only the designated forwarder (DF) for the Ethernet segment (ES) can forward BUM traffic. • The local bias rule, in which the local peer forwards the BUM packet and the remote peer drops it, is not supported. • Per MC-LAG forwarding logic, local bias is supported. 	The EVPN-MPLS and MC-LAG forwarding logic contradicts each other and can prevent BUM traffic from being forwarded to the ES.	Support local bias, thereby ignoring the DF and non-DF status of the port for locally switched traffic.
South bound (PE1 forwards BUM packet to PE2 and PE3).	Traffic received from PE1 follows the EVPN DF and non-DF forwarding rules for a multihomed ES.	None.	Not applicable.

MAC Learning

EVPN and MC-LAG use the same method for learning MAC addresses—namely, a PE device learns MAC addresses from its local interfaces and synchronizes the addresses to its peers. However, given that both EVPN and MC-LAG are synchronizing the addresses, an issue arises.

[Table 44 on page 1365](#) describes the issue and how the EVPN-MPLS interworking implementation prevents the issue. The use cases shown in [Figure 137 on page 1360](#) and [Figure 138 on page 1361](#)

illustrate the issue. In both use cases, PE1, PE2, and PE3 are EVPN peers, and PE2 and PE3 are MC-LAG peers.

Table 44: MAC Learning: EVPN and MC-LAG Synchronization Issue and Implementation Details

MAC Synchronization Use Case	EVPN Interworking with Junos Fusion Enterprise and MC-LAG Logic	Issue	Juniper Networks Implementation Approach
MAC addresses learned locally on single- or dual-homed interfaces on PE2 and PE3.	<ul style="list-style-type: none"> Between the EVPN peers, MAC addresses are synchronized using the EVPN BGP control plane. Between the MC-LAG peers, MAC addresses are synchronized using the MC-LAG ICCP control plane. 	PE2 and PE3 function as both EVPN peers and MC-LAG peers, which result in these devices having multiple MAC synchronization paths.	<ul style="list-style-type: none"> For PE1: use MAC addresses synchronized by EVPN BGP control plane. For PE2 and PE3: use MAC addresses synchronized by MC-LAG ICCP control plane.
MAC addresses learned locally on single- or dual-homed interfaces on PE1.	Between the EVPN peers, MAC addresses are synchronized using the EVPN BGP control plane.	None.	Not applicable.

Handling Down Link Between Cascade and Uplink Ports in Junos Fusion Enterprise

NOTE: This section applies only to EVPN-MPLS interworking with a Junos Fusion Enterprise.

In the Junos Fusion Enterprise shown in [Figure 137 on page 1360](#), assume that aggregation device PE2 receives a BUM packet from PE1 and that the link between the cascade port on PE2 and the corresponding uplink port on satellite device SD1 is down. Regardless of whether the BUM packet is handled by MC-LAG or EVPN multihoming active-active, the result is the same—the packet is forwarded via the ICL interface to PE3, which forwards it to dual-homed SD1.

To further illustrate how EVPN with multihoming active-active handles this situation with dual-homed SD1, assume that the DF interface resides on PE2 and is associated with the down link and that the non-DF interface resides on PE3. Typically, per EVPN with multihoming active-active forwarding logic, the non-DF interface drops the packet. However, because of the down link associated with the DF

interface, PE2 forwards the BUM packet via the ICL to PE3, and the non-DF interface on PE3 forwards the packet to SD1.

Layer 3 Gateway Support

The EVPN-MPLS interworking feature supports the following Layer 3 gateway functionality for extended bridge domains and VLANs:

- Integrated routing and bridging (IRB) interfaces to forward traffic between the extended bridge domains or VLANs.
- Default Layer 3 gateways to forward traffic from a physical (bare-metal) server in an extended bridge domain or VLAN to a physical server or virtual machine in another extended bridge domain or VLAN.

Release History Table

Release	Description
17.4R1	Starting with Junos OS Release 17.4R1, you can use Ethernet VPN (EVPN) to extend a Junos Fusion Enterprise or multichassis link aggregation group (MC-LAG) network over an MPLS network to a data center or campus network.

Example: EVPN-MPLS Interworking With Junos Fusion Enterprise

IN THIS SECTION

- [Requirements | 1367](#)
- [Overview and Topology | 1368](#)
- [Aggregation Device \(PE1 and PE2\) Configuration | 1370](#)
- [PE3 Configuration | 1383](#)

This example shows how to use Ethernet VPN (EVPN) to extend a Junos Fusion Enterprise over an MPLS network to a geographically distributed campus or enterprise network.

EVPN-MPLS interworking is supported with a Junos Fusion Enterprise, which is based on a multichassis link aggregation group (MC-LAG) infrastructure to provide redundancy for the EX9200 switches that function as aggregation devices.

The aggregation devices in the Junos Fusion Enterprise are connected to a provider edge (PE) device in an MPLS network. The PE device can be either an MX Series router or an EX9200 switch.

This example shows how to configure the aggregation devices in the Junos Fusion Enterprise and the PE device in the MPLS network to interwork with each other.

Requirements

This example uses the following hardware and software components:

- Three EX9200 switches:
 - PE1 and PE2, which both function as aggregation devices in the Junos Fusion Enterprise and EVPN BGP peers in the EVPN-MPLS overlay network.
 - PE3, which functions as an EVPN BGP peer in the EVPN-MPLS overlay network.
- The EX9200 switches are running Junos OS Release 17.4R1 or later software.

NOTE: Although the Junos Fusion Enterprise includes three satellite devices, this example focuses on the configuration of the PE1, PE2, and PE3. For more information about configuring satellite devices, see [Configuring or Expanding a Junos Fusion Enterprise](#).

The topology in [Figure 139 on page 1368](#) also includes PE3, which is positioned at the edge of an MPLS network. PE3 functions as the gateway between the Junos Fusion Enterprise network and a geographically distributed campus or enterprise network. PE1, PE2, and PE3 run EVPN, which enables

hosts in the Junos Fusion Enterprise network to communicate with hosts in the campus or enterprise network by way of the intervening MPLS network.

From the perspective of the EVPN-MPLS interworking feature, PE3 functions solely as an EVPN BGP peer, and PE1 and PE2 in the Junos Fusion Enterprise have dual roles:

- Aggregation devices in the Junos Fusion Enterprise.
- EVPN BGP peers in the EVPN-MPLS network.

Because of the dual roles, PE1 and PE2 are configured with Junos Fusion Enterprise, EVPN, BGP, and MPLS attributes.

[Table 45 on page 1369](#) outlines key Junos Fusion Enterprise and EVPN (BGP and MPLS) attributes configured on PE1, PE2, and PE3.

Table 45: Key Junos Fusion Enterprise and EVPN (BGP and MPLS) Attributes Configured on PE1, PE2, and PE3

Key Attributes	PE1	PE2	PE3
Junos Fusion Enterprise Attributes			
Interfaces	ICL: ge-1/0/3 ICCP: ge-1/0/2	ICL: ge-3/1/9 ICCP: ge-3/1/7	Not applicable
EVPN-MPLS			
Interfaces	Connection to PE3: ge-1/1/3 Connection to PE2: ge-1/1/7	Connection to PE3: ge-3/1/5 Connection to PE1: ge-3/1/8	Connection to PE1: ge-0/3/5 Connection to PE2: ge-0/3/7
IP addresses	BGP peer address: 10.25.0.1	BGP peer address: 10.25.0.2	BGP peer address: 10.25.0.3
Autonomous system	100	100	100
Virtual switch routing instances	evpn1	evpn1	evpn1

Note the following about the EVPN-MPLS interworking feature and its configuration:

- You must configure Ethernet segment identifiers (ESIs) on the dual-homed extended ports in the Junos Fusion Enterprise. The ESIs enable EVPN to identify the dual-homed extended ports.
- The only type of routing instance that is supported is the virtual switch instance (set routing-instances *name* instance-type virtual-switch).
- Only one virtual switch instance is supported with Junos Fusion Enterprise.
- On the aggregation devices in the Junos Fusion Enterprise, you must include the [bgp-peer](#) configuration statement in the [edit routing-instances *name* protocols evpn mclag] hierarchy level. This configuration statement enables the interworking of EVPN-MPLS with Junos Fusion Enterprise on the aggregation devices.
- Address Resolution Protocol (ARP) suppression is not supported.

Aggregation Device (PE1 and PE2) Configuration

IN THIS SECTION

- [CLI Quick Configuration | 1371](#)
- [PE1: Configuring Junos Fusion Enterprise | 1374](#)
- [PE1: Configuring EVPN-MPLS | 1376](#)
- [PE2: Configuring Junos Fusion Enterprise | 1379](#)
- [PE2: Configuring EVPN-MPLS | 1381](#)

To configure aggregation devices PE1 and PE2, perform these tasks.

NOTE: This section focuses on enabling EVPN-MPLS on PE1 and PE2. As a result, the Junos Fusion Enterprise configuration on PE1 and PE2 is performed without the use of the configuration synchronization feature. For information about configuration synchronization, see [Understanding Configuration Synchronization](#).

CLI Quick Configuration

PE1: Junos Fusion Enterprise Configuration

```

set interfaces ge-1/1/9 cascade-port
set interfaces ge-1/1/5 cascade-port
set chassis satellite-management fpc 120 cascade-ports ge-1/1/9
set chassis satellite-management cluster Cluster_100_108 cluster-id 2
set chassis satellite-management cluster Cluster_100_108 cascade-ports ge-1/1/5
set chassis satellite-management cluster Cluster_100_108 fpc 100 alias SD100
set chassis satellite-management cluster Cluster_100_108 fpc 100 system-id 88:e0:f3:1f:3d:50
set chassis satellite-management cluster Cluster_100_108 fpc 108 alias SD108
set chassis satellite-management cluster Cluster_100_108 fpc 108 system-id 88:e0:f3:1f:c8:d1
set chassis satellite-management cluster Cluster_100_108 fpc 100 member-id 1
set chassis satellite-management cluster Cluster_100_108 fpc 108 member-id 8
set chassis satellite-management upgrade-groups upgrade_120 satellite 120
set chassis satellite-management upgrade-groups upgrade_100 satellite 100
set chassis satellite-management redundancy-groups rg1 redundancy-group-id 2
set chassis satellite-management redundancy-groups chassis-id 1
set chassis satellite-management redundancy-groups rg1 peer-chassis-id 2 inter-chassis-link
ge-1/0/3
set chassis satellite-management redundancy-groups rg1 cluster Cluster_100_108
set interfaces ge-1/0/2 description iccp-link
set interfaces ge-1/0/2 unit 0 family inet address 10.20.20.1/24
set interfaces ge-1/0/3 description icl-link
set interfaces ge-1/0/3 unit 0 family ethernet-switching interface-mode trunk
set interfaces ge-1/0/3 unit 0 family ethernet-switching vlan members 100
set switch-options service-id 1

```

PE1: EVPN-MPLS Configuration

```

set interfaces lo0 unit 0 family inet address 10.25.0.1/32
set interfaces ge-1/1/3 unit 0 family inet address 10.0.1.1/30
set interfaces ge-1/1/3 unit 0 family mpls
set interfaces ge-1/1/7 unit 0 family inet address 10.0.3.1/30
set interfaces ge-1/1/7 unit 0 family mpls
set interfaces ge-108/0/25 unit 0 esi 00:01:02:03:04:00:01:02:04:26
set interfaces ge-108/0/25 unit 0 esi all-active
set interfaces ge-108/0/25 unit 0 family ethernet-switching vlan members v100
set interfaces ge-108/0/27 unit 0 esi 00:01:02:03:04:00:01:02:04:28
set interfaces ge-108/0/27 unit 0 esi all-active

```

```

set interfaces ge-108/0/27 unit 0 family ethernet-switching vlan members v100
set routing-options router-id 10.25.0.1
set routing-options autonomous-system 100
set protocols mpls interface lo0.0
set protocols mpls interface ge-1/1/3.0
set protocols mpls interface ge-1/1/7.0
set protocols bgp local-address 10.25.0.1
set protocols bgp peer-as 100
set protocols bgp local-as 100
set protocols bgp group evpn-mes type internal
set protocols bgp group evpn-mes family evpn signaling
set protocols bgp group evpn-mes peer-as 100
set protocols bgp group evpn-mes neighbor 10.25.0.2
set protocols bgp group evpn-mes neighbor 10.25.0.3
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-1/1/3.0
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface ge-1/1/7.0
set protocols ldp interface lo0.0
set protocols ldp interface ge-1/1/3.0
set protocols ldp interface ge-1/1/7.0
set routing-instances evpn1 instance-type virtual-switch
set routing-instances evpn1 interface ge-108/0/25.0
set routing-instances evpn1 interface ge-108/0/27.0
set routing-instances evpn1 interface ge-1/0/3.0
set routing-instances evpn1 route-distinguisher 10.25.0.1:1
set routing-instances evpn1 vrf-target target:100:1
set routing-instances evpn1 protocols evpn label-allocation per-instance
set routing-instances evpn1 protocols evpn extended-vlan-list 100
set routing-instances evpn1 protocols evpn mlag bgp-peer 10.25.0.2
set routing-instances evpn1 switch-options service-id 2
set routing-instances evpn1 vlans v100 vlan-id 100

```

PE2: Junos Fusion Enterprise Configuration

```

set interfaces ge-3/1/4 cascade-port
set chassis satellite-management cluster Cluster_100_108 cluster-id 2
set chassis satellite-management cluster Cluster_100_108 cascade-ports ge-3/1/4
set chassis satellite-management cluster Cluster_100_108 fpc 100 alias SD100
set chassis satellite-management cluster Cluster_100_108 fpc 100 system-id 88:e0:f3:1f:3d:50
set chassis satellite-management cluster Cluster_100_108 fpc 108 alias SD108

```

```

set chassis satellite-management cluster Cluster_100_108 fpc 108 system-id 88:e0:f3:1f:c8:d1
set chassis satellite-management cluster Cluster_100_108 fpc 100 member-id 1
set chassis satellite-management cluster Cluster_100_108 fpc 108 member-id 8
set chassis satellite-management upgrade-groups upgrade_100 satellite 100
set chassis satellite-management redundancy-groups rg1 redundancy-group-id 2
set chassis satellite-management redundancy-groups chassis-id 2
set chassis satellite-management redundancy-groups rg1 peer-chassis-id 1 inter-chassis-link
ge-3/1/9
set chassis satellite-management redundancy-groups rg1 cluster Cluster_100_108
set interfaces ge-3/1/7 description iccp-link
set interfaces ge-3/1/7 unit 0 family inet address 10.20.20.2/24
set interfaces ge-3/1/9 description icl-link
set interfaces ge-3/1/9 unit 0 family ethernet-switching interface-mode trunk
set interfaces ge-3/1/9 unit 0 family ethernet-switching vlan members 100
set switch-options service-id 1

```

PE2: EVPN-MPLS Configuration

```

set interfaces lo0 unit 0 family inet address 10.25.0.2/32
set interfaces ge-3/1/5 unit 0 family inet address 10.0.4.2/30
set interfaces ge-3/1/5 unit 0 family mpls
set interfaces ge-3/1/8 unit 0 family inet address 10.0.3.2/30
set interfaces ge-3/1/8 unit 0 family mpls
set interfaces irb unit 0 family inet address 10.5.5.1/24 virtual-gateway-address 10.5.5.5
set interfaces ge-108/0/25 unit 0 esi 00:01:02:03:04:00:01:02:04:26
set interfaces ge-108/0/25 unit 0 esi all-active
set interfaces ge-108/0/25 unit 0 family ethernet-switching vlan members v100
set interfaces ge-108/0/27 unit 0 esi 00:01:02:03:04:00:01:02:04:28
set interfaces ge-108/0/27 unit 0 esi all-active
set interfaces ge-108/0/27 unit 0 family ethernet-switching vlan members v100
set routing-options router-id 10.25.0.2
set routing-options autonomous-system 100
set protocols mpls interface lo0.0
set protocols mpls interface ge-3/1/5.0
set protocols mpls interface ge-3/1/8.0
set protocols bgp local-address 10.25.0.2
set protocols bgp peer-as 100
set protocols bgp local-as 100
set protocols bgp group evpn-mes type internal
set protocols bgp group evpn-mes family evpn signaling
set protocols bgp group evpn-mes peer-as 100
set protocols bgp group evpn-mes neighbor 10.25.0.1

```



```

set protocols bgp group evpn-mes neighbor 10.25.0.3
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-3/1/5.0
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface ge-3/1/8.0
set protocols ldp interface lo0.0
set protocols ldp interface ge-3/1/5.0
set protocols ldp interface ge-3/1/8.0
set routing-instances evpn1 instance-type virtual-switch
set routing-instances evpn1 interface ge-108/0/25.0
set routing-instances evpn1 interface ge-108/0/27.0
set routing-instances evpn1 interface ge-3/1/9.0
set routing-instances evpn1 route-distinguisher 10.25.0.2:1
set routing-instances evpn1 vrf-target target:100:1
set routing-instances evpn1 protocols evpn label-allocation per-instance
set routing-instances evpn1 protocols evpn extended-vlan-list 100
set routing-instances evpn1 protocols evpn mclag bgp-peer 10.25.0.1
set routing-instances evpn1 switch-options service-id 2
set routing-instances evpn1 vlans v100 vlan-id 100
set routing-instances evpn1 vlans v100 l3-interface irb.0
set routing-instances evpn1 vlans v100 no-arp-suppression

```

PE1: Configuring Junos Fusion Enterprise

Step-by-Step Procedure

1. Configure the cascade ports.

```

[edit]
user@switch# set interfaces ge-1/1/9 cascade-port
user@switch# set interfaces ge-1/1/5 cascade-port

```

2. Configure the FPC slot ID for standalone satellite device SD120 and map it to a cascade port.

```

[edit]
user@switch# set chassis satellite-management fpc 120 cascade-ports ge-1/1/9

```

3. Create a satellite device cluster, and assign a name and a cluster ID to it.

```
[edit]
user@switch# set chassis satellite-management cluster Cluster_100_108 cluster-id 2
```

4. Define the cascade ports associated with the satellite device cluster.

```
[edit]
user@switch# set chassis satellite-management cluster Cluster_100_108 cascade-ports ge-1/1/5
user@switch# set chassis satellite-management cluster Cluster_100_108 cascade-ports ge-1/1/9
```

5. Configure the FPC slot ID number, and map it to the MAC address of satellite devices SD100 and SD108, respectively.

```
[edit]
user@switch# set chassis satellite-management cluster Cluster_100_108 fpc 100 alias SD100
user@switch# set chassis satellite-management cluster Cluster_100_108 fpc 100 system-id
88:e0:f3:1f:3d:50
user@switch# set chassis satellite-management cluster Cluster_100_108 fpc 108 alias SD108
user@switch# set chassis satellite-management cluster Cluster_100_108 fpc 108 system-id
88:e0:f3:1f:c8:d1
```

6. Assign a member ID to each satellite device in the satellite device cluster.

```
[edit]
user@switch# set chassis satellite-management cluster Cluster_100_108 fpc 100 member-id 1
user@switch# set chassis satellite-management cluster Cluster_100_108 fpc 108 member-id 8
```

7. Create two satellite software upgrade groups—one that includes satellite device SD120 and another that includes satellite device SD100.

```
[edit]
user@switch# set chassis satellite-management upgrade-groups upgrade_120 satellite 120
user@switch# set chassis satellite-management upgrade-groups upgrade_100 satellite 100
```

8. Create and configure a redundancy group, which includes the aggregation devices and satellite devices in Cluster_100_108.

```
[edit]
user@switch# set chassis satellite-management redundancy-groups rg1 redundancy-group-id 2
user@switch# set chassis satellite-management redundancy-groups chassis-id 1
user@switch# set chassis satellite-management redundancy-groups rg1 peer-chassis-id 2 inter-
chassis-link ge-1/0/3
user@switch# set chassis satellite-management redundancy-groups rg1 cluster Cluster_100_108
```

9. Configure the ICL and ICCP links.

```
[edit]
user@switch# set interfaces ge-1/0/2 description iccp-link
user@switch# set interfaces ge-1/0/2 unit 0 family inet address 10.20.20.1/24
user@switch# set interfaces ge-1/0/3 description icl-link
user@switch# set interfaces ge-1/0/3 unit 0 family ethernet-switching interface-mode trunk
user@switch# set interfaces ge-1/0/3 unit 0 family ethernet-switching vlan members 100
user@switch# set switch-options service-id 1
```

NOTE: While this step shows the configuration of interface ge-1/0/2, which is designated as the ICCP interface, it does not show how to configure the ICCP attributes on interface ge-1/0/2. By default, ICCP is automatically provisioned in a Junos Fusion Enterprise using dual aggregation devices. For more information about the automatic provisioning of ICCP, see [Configuring or Expanding a Junos Fusion Enterprise](#).

PE1: Configuring EVPN-MPLS

Step-by-Step Procedure

1. Configure the loopback interface and the interfaces connected to the other PE devices.

```
[edit]
user@switch# set interfaces lo0 unit 0 family inet address 10.25.0.1/32
user@switch# set interfaces ge-1/1/3 unit 0 family inet address 10.0.1.1/30
user@switch# set interfaces ge-1/1/3 unit 0 family mpls
```

```

user@switch# set interfaces ge-1/1/7 unit 0 family inet address 10.0.3.1/30
user@switch# set interfaces ge-1/1/7 unit 0 family mpls

```

2. Configure the extended ports with EVPN multihoming in active-active mode, an ESI, and map the ports to VLAN v100..

```

[edit]
user@switch# set interfaces ge-108/0/25 unit 0 esi 00:01:02:03:04:00:01:02:04:26
user@switch# set interfaces ge-108/0/25 unit 0 esi all-active
user@switch# set interfaces ge-108/0/25 unit 0 family ethernet-switching vlan members v100
user@switch# set interfaces ge-108/0/27 unit 0 esi 00:01:02:03:04:00:01:02:04:28
user@switch# set interfaces ge-108/0/27 unit 0 esi all-active
user@switch# set interfaces ge-108/0/27 unit 0 family ethernet-switching vlan members v100

```

3. Assign a router ID and the autonomous system in which PE1, PE2, and PE3 reside.

```

[edit]
user@switch# set routing-options router-id 10.25.0.1
user@switch# set routing-options autonomous-system 100

```

4. Enable MPLS on the loopback interface and interfaces ge-1/1/3.0 and ge-1/1/7.0.

```

[edit]
user@switch# set protocols mpls interface lo0.0
user@switch# set protocols mpls interface ge-1/1/3.0
user@switch# set protocols mpls interface ge-1/1/7.0

```

5. Configure an IBGP overlay that includes PE1, PE2, and PE3.

```

[edit]
user@switch# set protocols bgp local-address 10.25.0.1
user@switch# set protocols bgp peer-as 100
user@switch# set protocols bgp local-as 100
user@switch# set protocols bgp group evpn-mes type internal
user@switch# set protocols bgp group evpn-mes family evpn signaling
user@switch# set protocols bgp group evpn-mes peer-as 100
user@switch# set protocols bgp group evpn-mes neighbor 10.25.0.2
user@switch# set protocols bgp group evpn-mes neighbor 10.25.0.3

```

6. Configure OSPF as the internal routing protocol for EVPN by specifying an area ID and interfaces on which EVPN-MPLS is enabled.

```
[edit]
user@switch# set protocols ospf traffic-engineering
user@switch# set protocols ospf area 0.0.0.0 interface ge-1/1/3.0
user@switch# set protocols ospf area 0.0.0.0 interface lo0.0
user@switch# set protocols ospf area 0.0.0.0 interface fxp0.0 disable
user@switch# set protocols ospf area 0.0.0.0 interface ge-1/1/7.0
```

7. Configure the Label Distribution Protocol (LDP) on the loopback interface and the interfaces on which EVPN-MPLS is enabled.

```
[edit]
user@switch# set protocols ldp interface lo0.0
user@switch# set protocols ldp interface ge-1/1/3.0
user@switch# set protocols ldp interface ge-1/1/7.0
```

8. Configure a virtual switch routing instance for VLAN v100, and include the interfaces and other entities associated with the VLAN.

```
[edit]
user@switch# set routing-instances evpn1 instance-type virtual-switch
user@switch# set routing-instances evpn1 interface ge-108/0/25.0
user@switch# set routing-instances evpn1 interface ge-108/0/27.0
user@switch# set routing-instances evpn1 interface ge-1/0/3.0
user@switch# set routing-instances evpn1 route-distinguisher 10.25.0.1:1
user@switch# set routing-instances evpn1 vrf-target target:100:1
user@switch# set routing-instances evpn1 protocols evpn label-allocation per-instance
user@switch# set routing-instances evpn1 protocols evpn extended-vlan-list 100
user@switch# set routing-instances evpn1 protocols evpn mclag bgp-peer 10.25.0.2
user@switch# set routing-instances evpn1 switch-options service-id 2
user@switch# set routing-instances evpn1 vlans v100 vlan-id 100
```

PE2: Configuring Junos Fusion Enterprise

Step-by-Step Procedure

1. Configure the cascade port.

```
[edit]
user@switch# set interfaces ge-3/1/4 cascade-port
```

2. Create a satellite device cluster, and assign a name and a cluster ID to it.

```
[edit]
user@switch# set chassis satellite-management cluster Cluster_100_108 cluster-id 2
```

3. Define the cascade port associated with the satellite device cluster.

```
[edit]
user@switch# set chassis satellite-management cluster Cluster_100_108 cascade-ports ge-3/1/4
```

4. Configure the FPC slot ID number, and map it to the MAC address of satellite devices SD100 and SD108, respectively.

```
[edit]
user@switch# set chassis satellite-management cluster Cluster_100_108 fpc 100 alias SD100
user@switch# set chassis satellite-management cluster Cluster_100_108 fpc 100 system-id
88:e0:f3:1f:3d:50
user@switch# set chassis satellite-management cluster Cluster_100_108 fpc 108 alias SD108
user@switch# set chassis satellite-management cluster Cluster_100_108 fpc 108 system-id
88:e0:f3:1f:c8:d1
```

5. Assign a member ID to each satellite device in the satellite device cluster.

```
[edit]
user@switch# set chassis satellite-management cluster Cluster_100_108 fpc 100 member-id 1
user@switch# set chassis satellite-management cluster Cluster_100_108 fpc 108 member-id 8
```

6. Create a satellite software upgrade group that includes satellite device SD100.

```
[edit]
user@switch# set chassis satellite-management upgrade-groups upgrade_100 satellite 100
```

7. Create and configure a redundancy group, which includes the aggregation devices and satellite devices in Cluster_100_108.

```
[edit]
user@switch# set chassis satellite-management redundancy-groups rg1 redundancy-group-id 2
user@switch# set chassis satellite-management redundancy-groups chassis-id 2
user@switch# set chassis satellite-management redundancy-groups rg1 peer-chassis-id 1 inter-
chassis-link ge-3/1/9
user@switch# set chassis satellite-management redundancy-groups rg1 cluster Cluster_100_108
```

8. Configure the ICL and ICCP links.

```
[edit]
user@switch# set interfaces ge-3/1/7 description iccp-link
user@switch# set interfaces ge-3/1/7 unit 0 family inet address 10.20.20.2/24
user@switch# set interfaces ge-3/1/9 description icl-link
user@switch# set interfaces ge-3/1/9 unit 0 family ethernet-switching interface-mode trunk
user@switch# set interfaces ge-3/1/9 unit 0 family ethernet-switching vlan members 100
user@switch# set switch-options service-id 1
```

NOTE: While this step shows the configuration of interface ge-3/1/7, which is designated as the ICCP interface, it does not show how to configure the ICCP attributes on interface ge-3/1/7. By default, ICCP is automatically provisioned in a Junos Fusion Enterprise using dual aggregation devices. For more information about the automatic provisioning of ICCP, see [Configuring or Expanding a Junos Fusion Enterprise](#).

PE2: Configuring EVPN-MPLS

Step-by-Step Procedure

1. Configure the loopback interface, the interfaces connected to the other PE devices, and an IRB interface that is also configured as a default Layer 3 gateway.

```
[edit]
user@switch# set interfaces lo0 unit 0 family inet address 10.25.0.2/32
user@switch# set interfaces ge-3/1/5 unit 0 family inet address 10.0.4.2/30
user@switch# set interfaces ge-3/1/5 unit 0 family mpls
user@switch# set interfaces ge-3/1/8 unit 0 family inet address 10.0.3.2/30
user@switch# set interfaces ge-3/1/8 unit 0 family mpls
user@switch# set interfaces irb unit 0 family inet address 10.5.5.1/24 virtual-gateway-
address 10.5.5.5
```

2. Configure the extended ports with EVPN multihoming in active-active mode, an ESI, and map the ports to VLAN v100..

```
[edit]
user@switch# set interfaces ge-108/0/25 unit 0 esi 00:01:02:03:04:00:01:02:04:26
user@switch# set interfaces ge-108/0/25 unit 0 esi all-active
user@switch# set interfaces ge-108/0/25 unit 0 family ethernet-switching vlan members v100
user@switch# set interfaces ge-108/0/27 unit 0 esi 00:01:02:03:04:00:01:02:04:28
user@switch# set interfaces ge-108/0/27 unit 0 esi all-active
user@switch# set interfaces ge-108/0/27 unit 0 family ethernet-switching vlan members v100
```

3. Assign a router ID and the autonomous system in which PE1, PE2, and PE3 reside.

```
[edit]
user@switch# set routing-options router-id 10.25.0.2
user@switch# set routing-options autonomous-system 100
```

4. Enable MPLS on the loopback interface and interfaces ge-3/1/5.0 and ge-3/1/8.0.

```
[edit]
user@switch# set protocols mpls interface lo0.0
```



```

user@switch# set protocols mpls interface ge-3/1/5.0
user@switch# set protocols mpls interface ge-3/1/8.0

```

5. Configure an IBGP overlay that includes PE1, PE2, and PE3.

```

[edit]
user@switch# set protocols bgp local-address 10.25.0.2
user@switch# set protocols bgp peer-as 100
user@switch# set protocols bgp local-as 100
user@switch# set protocols bgp group evpn-mes type internal
user@switch# set protocols bgp group evpn-mes family evpn signaling
user@switch# set protocols bgp group evpn-mes peer-as 100
user@switch# set protocols bgp group evpn-mes neighbor 10.25.0.1
user@switch# set protocols bgp group evpn-mes neighbor 10.25.0.3

```

6. Configure OSPF as the internal routing protocol for EVPN by specifying an area ID and interfaces on which EVPN-MPLS is enabled.

```

[edit]
user@switch# set protocols ospf traffic-engineering
user@switch# set protocols ospf area 0.0.0.0 interface ge-3/1/5.0
user@switch# set protocols ospf area 0.0.0.0 interface lo0.0
user@switch# set protocols ospf area 0.0.0.0 interface fxp0.0 disable
user@switch# set protocols ospf area 0.0.0.0 interface ge-3/1/8.0

```

7. Configure the LDP on the loopback interface and the interfaces on which EVPN-MPLS is enabled.

```

[edit]
user@switch# set protocols ldp interface lo0.0
user@switch# set protocols ldp interface ge-3/1/5.0
user@switch# set protocols ldp interface ge-3/1/8.0

```

8. Configure a virtual switch routing instance for VLAN v100, and include the interfaces and other entities associated with the VLAN.

```

[edit]
user@switch# set routing-instances evpn1 instance-type virtual-switch
user@switch# set routing-instances evpn1 interface ge-108/0/25.0
user@switch# set routing-instances evpn1 interface ge-108/0/27.0

```

```

user@switch# set routing-instances evpn1 interface ge-3/1/9.0
user@switch# set routing-instances evpn1 route-distinguisher 10.25.0.2:1
user@switch# set routing-instances evpn1 vrf-target target:100:1
user@switch# set routing-instances evpn1 protocols evpn label-allocation per-instance
user@switch# set routing-instances evpn1 protocols evpn extended-vlan-list 100
user@switch# set routing-instances evpn1 protocols evpn mclag bgp-peer 10.25.0.1
user@switch# set routing-instances evpn1 switch-options service-id 2
user@switch# set routing-instances evpn1 vlans v100 vlan-id 100
user@switch# set routing-instances evpn1 vlans v100 l3-interface irb.0
user@switch# set routing-instances evpn1 vlans v100 no-arp-suppression

```

PE3 Configuration

IN THIS SECTION

- [CLI Quick Configuration | 1383](#)
- [PE3: Configuring EVPN-MPLS | 1384](#)

CLI Quick Configuration

PE3: EVPN-MPLS Configuration

```

set interfaces lo0 unit 0 family inet address 10.25.0.3/32
set interfaces ge-0/3/5 unit 0 family inet address 10.0.1.2/30
set interfaces ge-0/3/5 unit 0 family mpls
set interfaces ge-0/3/7 unit 0 family inet address 10.0.4.1/30
set interfaces ge-0/3/7 unit 0 family mpls
set interfaces ge-0/0/46 unit 0 esi 00:01:02:03:04:00:01:02:04:12
set interfaces ge-0/0/46 unit 0 esi all-active
set interfaces ge-0/0/46 unit 0 family ethernet-switching vlan members 100
set routing-options router-id 10.25.0.3
set routing-options autonomous-system 100
set routing-options forwarding-table export evpn-pplb
set policy-options policy-statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pplb then load-balance per-packet
set protocols mpls interface lo0.0
set protocols mpls interface ge-0/3/5.0
set protocols mpls interface ge-0/3/7.0

```

```

set protocols bgp local-address 10.25.0.3
set protocols bgp peer-as 100
set protocols bgp local-as 100
set protocols bgp group evpn-mes type internal
set protocols bgp group evpn-mes family evpn signaling
set protocols bgp group evpn-mes peer-as 100
set protocols bgp group evpn-mes neighbor 10.25.0.2
set protocols bgp group evpn-mes neighbor 10.25.0.1
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-0/3/5.0
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface ge-0/3/7.0
set protocols ldp interface lo0.0
set protocols ldp interface ge-0/3/5.0
set protocols ldp interface ge-0/3/7.0
set routing-instances evpn1 instance-type virtual-switch
set routing-instances evpn1 interface ge-0/0/46.0
set routing-instances evpn1 route-distinguisher 10.25.0.3:1
set routing-instances evpn1 vrf-target target:100:1
set routing-instances evpn1 protocols evpn label-allocation per-instance
set routing-instances evpn1 protocols evpn extended-vlan-list 100
set routing-instances evpn1 switch-options service-id 2
set routing-instances evpn1 vlans v100 vlan-id 100

```

PE3: Configuring EVPN-MPLS

Step-by-Step Procedure

1. Configure the interfaces on EVPN-MPLS interworking occurs.

```

[edit]
user@switch# set interfaces lo0 unit 0 family inet address 10.25.0.3/32
user@switch# set interfaces ge-0/3/5 unit 0 family inet address 10.0.1.2/30
user@switch# set interfaces ge-0/3/5 unit 0 family mpls
user@switch# set interfaces ge-0/3/7 unit 0 family inet address 10.0.4.1/30
user@switch# set interfaces ge-0/3/7 unit 0 family mpls

```

2. Configure interface ge-0/0/46 with EVPN multihoming in active-active mode, an ESI, and map the ports to VLAN v100..

```
[edit]
user@switch# set interfaces ge-0/0/46 unit 0 esi 00:01:02:03:04:00:01:02:04:12
user@switch# set interfaces ge-0/0/46 unit 0 esi all-active
user@switch# set interfaces ge-0/0/46 unit 0 family ethernet-switching vlan members 100
```

3. Assign a router ID and the autonomous system in which the PE1, PE2, and PE3 reside.

```
[edit]
user@switch# set routing-options router-id 10.25.0.2
user@switch# set routing-options autonomous-system 100
```

4. Enable per-packet load-balancing for EVPN routes when EVPN multihoming active-active mode is used.

```
[edit]
user@switch# set routing-options forwarding-table export evpn-pplb
user@switch# set policy-options policy-statement evpn-pplb from protocol evpn
user@switch# set policy-options policy-statement evpn-pplb then load-balance per-packet
```

5. Enable MPLS on the loopback interface and interfaces ge-0/3/5.0 and ge-0/3/7.0.

```
[edit]
user@switch# set protocols mpls interface lo0.0
user@switch# set protocols mpls interface ge-0/3/5.0
user@switch# set protocols mpls interface ge-0/3/7.0
```

6. Configure an IBGP overlay that includes PE1, PE2, and PE3.

```
[edit]
user@switch# set protocols bgp local-address 10.25.0.3
user@switch# set protocols bgp peer-as 100
user@switch# set protocols bgp local-as 100
user@switch# set protocols bgp group evpn-mes type internal
user@switch# set protocols bgp group evpn-mes family evpn signaling
user@switch# set protocols bgp group evpn-mes peer-as 100
```

```

user@switch# set protocols bgp group evpn-mes neighbor 10.25.0.2
user@switch# set protocols bgp group evpn-mes neighbor 10.25.0.1

```

7. Configure OSPF as the internal routing protocol for EVPN by specifying an area ID and interfaces on which EVPN-MPLS is enabled.

```

[edit]
user@switch# set protocols ospf traffic-engineering
user@switch# set protocols ospf area 0.0.0.0 interface ge-0/3/5.0
user@switch# set protocols ospf area 0.0.0.0 interface lo0.0
user@switch# set protocols ospf area 0.0.0.0 interface fxp0.0 disable
user@switch# set protocols ospf area 0.0.0.0 interface ge-0/3/7.0

```

8. Configure the LDP on the loopback interface and the interfaces on which EVPN-MPLS is enabled.

```

[edit]
user@switch# set protocols ldp interface lo0.0
user@switch# set protocols ldp interface ge-0/3/5.0
user@switch# set protocols ldp interface ge-0/3/7.0

```

9. Configure a virtual switch routing instance for VLAN v100, and include the interfaces and other entities associated with the VLAN.

```

[edit]
user@switch# set routing-instances evpn1 instance-type virtual-switch
user@switch# set routing-instances evpn1 interface ge-0/0/46.0
user@switch# set routing-instances evpn1 route-distinguisher 10.25.0.3:1
user@switch# set routing-instances evpn1 vrf-target target:100:1
user@switch# set routing-instances evpn1 protocols evpn label-allocation per-instance
user@switch# set routing-instances evpn1 protocols evpn extended-vlan-list 100
user@switch# set routing-instances evpn1 switch-options service-id 2
user@switch# set routing-instances evpn1 vlans v100 vlan-id 100

```

RELATED DOCUMENTATION

[Understanding EVPN-MPLS Interworking with Junos Fusion Enterprise and MC-LAG](#) | 1359

Example: EVPN-MPLS Interworking With an MC-LAG Topology

IN THIS SECTION

- [Requirements | 1387](#)
- [Overview and Topology | 1388](#)
- [PE1 and PE2 Configuration | 1390](#)
- [PE3 Configuration | 1407](#)

This example shows how to use Ethernet VPN (EVPN) to extend a multichassis link aggregation (MC-LAG) network over an MPLS network to a data center network or geographically distributed campus network.

EVPN-MPLS interworking is supported with an MC-LAG topology in which two MX Series routers, two EX9200 switches, or a mix of the two Juniper Networks devices function as MC-LAG peers, which use the Inter-Chassis Control Protocol (ICCP) and an interchassis link (ICL) to connect and maintain the topology. The MC-LAG peers are connected to a provider edge (PE) device in an MPLS network. The PE device can be either an MX Series router or an EX9200 switch.

This example shows how to configure the MC-LAG peers and PE device in the MPLS network to interwork with each other.

Requirements

This example uses the following hardware and software components:

- Three EX9200 switches:
 - PE1 and PE2, which both function as MC-LAG peers in the MC-LAG topology and EVPN BGP peers in the EVPN-MPLS overlay network.
 - PE3, which functions as an EVPN BGP peer in the EVPN-MPLS overlay network.
- The EX9200 switches are running Junos OS Release 17.4R1 or later software.

NOTE: Although the MC-LAG topology includes two customer edge (CE) devices, this example focuses on the configuration of the PE1, PE2, and PE3.

The topology in [Figure 140 on page 1388](#) also includes PE3 at the edge of an MPLS network. PE3 functions as the gateway between the MC-LAG network and either a data center or a geographically distributed campus network. PE1, PE2, and PE3 run EVPN, which enables hosts in the MC-LAG network

to communicate with hosts in the data center or other campus network by way of an intervening MPLS network.

From the perspective of the EVPN-MPLS interworking feature, PE3 functions solely as an EVPN BGP peer, and PE1 and PE2 in the MC-LAG topology have dual roles:

- MC-LAG peers in the MC-LAG network.
- EVPN BGP peers in the EVPN-MPLS network.

Because of the dual roles, PE1 and PE2 are configured with MC-LAG, EVPN, BGP, and MPLS attributes.

[Table 46 on page 1389](#) outlines key MC-LAG and EVPN (BGP and MPLS) attributes configured on PE1, PE2, and PE3.

Table 46: Key MC-LAG and EVPN (BGP and MPLS) Attributes Configured on PE1, PE2, and PE3

Key Attributes	PE1	PE2	PE3
MC-LAG Attributes			
Interfaces	ICL: aggregated Ethernet interface ae1, which is comprised of xe-2/1/1 and xe-2/1/2 ICCP: xe-2/1/0	ICL: aggregated Ethernet interface ae1, which is comprised of xe-2/1/1 and xe-2/1/2 ICCP: xe-2/1/0	Not applicable
EVPN-MPLS			
Interfaces	Connection to PE3: xe-2/0/0 Connection to PE2: xe-2/0/2	Connection to PE3: xe-2/0/2 Connection to PE1: xe-2/0/0	Connection to PE1: xe-2/0/2 Connection to PE2: xe-2/0/3
IP addresses	BGP peer address: 198.51.100.1	BGP peer address: 198.51.100.2	BGP peer address: 198.51.100.3
Autonomous system	65000	65000	65000

Table 46: Key MC-LAG and EVPN (BGP and MPLS) Attributes Configured on PE1, PE2, and PE3
(Continued)

Key Attributes	PE1	PE2	PE3
Virtual switch routing instances	evpn1, evpn2, evpn3	evpn1, evpn2, evpn3	evpn1, evpn2, evpn3

Note the following about the EVPN-MPLS interworking feature and its configuration:

- You must configure Ethernet segment identifiers (ESIs) on the dual-homed interfaces in the MC-LAG topology. The ESIs enable EVPN to identify the dual-homed interfaces.
- The only type of routing instance that is supported is the virtual switch instance (set routing-instances *name* instance-type virtual-switch).
- On the MC-LAG peers, you must include the bgp-peer configuration statement in the [edit routing-instances *name* protocols evpn mclag] hierarchy level. This configuration statement enables the interworking of EVPN-MPLS with MC-LAG on the MC-LAG peers.
- Address Resolution Protocol (ARP) suppression is not supported.

PE1 and PE2 Configuration

IN THIS SECTION

- [CLI Quick Configuration | 1391](#)
- [PE1: Configuring MC-LAG | 1397](#)
- [PE1: Configuring EVPN-MPLS | 1399](#)
- [PE2: Configuring MC-LAG | 1402](#)
- [PE2: Configuring EVPN-MPLS | 1404](#)

To configure PE1 and PE2, perform these tasks:

CLI Quick Configuration

PE1: MC-LAG Configuration

```

set chassis aggregated-devices ethernet device-count 3
set interfaces xe-2/0/1 gigether-options 802.3ad ae0
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp periodic fast
set interfaces ae0 aggregated-ether-options lacp system-id 00:00:11:11:11:11
set interfaces ae0 aggregated-ether-options lacp admin-key 1
set interfaces ae0 aggregated-ether-options mc-ae mc-ae-id 1
set interfaces ae0 aggregated-ether-options mc-ae redundancy-group 2
set interfaces ae0 aggregated-ether-options mc-ae chassis-id 0
set interfaces ae0 aggregated-ether-options mc-ae mode active-active
set interfaces ae0 aggregated-ether-options mc-ae status-control active
set interfaces ae0 unit 1 esi 00:11:22:33:44:55:66:77:88:99
set interfaces ae0 unit 1 esi all-active
set interfaces ae0 unit 1 family ethernet-switching interface-mode trunk
set interfaces ae0 unit 1 family ethernet-switching vlan members 1
set interfaces ae0 unit 2 esi 00:11:11:11:11:11:11:11:11:11
set interfaces ae0 unit 2 esi all-active
set interfaces ae0 unit 2 family ethernet-switching interface-mode trunk
set interfaces ae0 unit 2 family ethernet-switching vlan members 2
set interfaces ae0 unit 3 esi 00:11:22:22:22:22:22:22:22:22
set interfaces ae0 unit 3 esi all-active
set interfaces ae0 unit 3 family ethernet-switching interface-mode trunk
set interfaces ae0 unit 3 family ethernet-switching vlan members 3
set interfaces xe-2/0/6 enable
set interfaces xe-2/0/6 flexible-vlan-tagging
set interfaces xe-2/0/6 encapsulation flexible-ethernet-services
set interfaces xe-2/0/6 unit 1 family ethernet-switching interface-mode trunk
set interfaces xe-2/0/6 unit 1 family ethernet-switching vlan members 1
set interfaces xe-2/0/6 unit 2 family ethernet-switching interface-mode trunk
set interfaces xe-2/0/6 unit 2 family ethernet-switching vlan members 2
set interfaces xe-2/0/6 unit 3 family ethernet-switching interface-mode trunk
set interfaces xe-2/0/6 unit 3 family ethernet-switching vlan members 3
set interfaces xe-2/1/0 unit 0 family inet address 203.0.113.1/24
set interfaces xe-2/1/1 gigether-options 802.3ad ae1
set interfaces xe-2/1/2 gigether-options 802.3ad ae1
set interfaces ae1 flexible-vlan-tagging

```

```

set interfaces ae1 encapsulation flexible-ethernet-services
set interfaces ae1 aggregated-ether-options lacp active
set interfaces ae1 unit 1 family ethernet-switching interface-mode trunk
set interfaces ae1 unit 1 family ethernet-switching vlan members 1
set interfaces ae1 unit 2 family ethernet-switching interface-mode trunk
set interfaces ae1 unit 2 family ethernet-switching vlan members 2
set interfaces ae1 unit 3 family ethernet-switching interface-mode trunk
set interfaces ae1 unit 3 family ethernet-switching vlan members 3
set multi-chassis multi-chassis-protection 203.0.113.2 interface ae1
set protocols iccp local-ip-addr 203.0.113.1
set protocols iccp peer 203.0.113.2 session-establishment-hold-time 600
set protocols iccp peer 203.0.113.2 redundancy-group-id-list 2
set protocols iccp peer 203.0.113.2 liveness-detection minimum-interval 10000
set protocols iccp peer 203.0.113.2 liveness-detection multiplier 3

```

PE1: EVPN-MPLS Configuration

```

set interfaces lo0 unit 0 family inet address 198.51.100.1/32 primary
set interfaces xe-2/0/0 unit 0 family inet address 192.0.2.2/24
set interfaces xe-2/0/0 unit 0 family mpls
set interfaces xe-2/0/2 unit 0 family inet address 192.0.2.111/24
set interfaces xe-2/0/2 unit 0 family mpls
set interfaces irb unit 1 family inet address 10.2.1.1/24 virtual-gateway-address 10.2.1.254
set interfaces irb unit 2 family inet address 10.2.2.1/24 virtual-gateway-address 10.2.2.254
set interfaces irb unit 3 family inet address 10.2.3.1/24 virtual-gateway-address 10.2.3.254
set routing-options router-id 198.51.100.1
set routing-options autonomous-system 65000
set routing-options forwarding-table export evpn-pplb
set protocols mpls interface xe-2/0/0.0
set protocols mpls interface xe-2/0/2.0
set protocols bgp group evpn type internal
set protocols bgp group evpn local-address 198.51.100.1
set protocols bgp group evpn family evpn signaling
set protocols bgp group evpn local-as 65000
set protocols bgp group evpn neighbor 198.51.100.2
set protocols bgp group evpn neighbor 198.51.100.3
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols ospf area 0.0.0.0 interface xe-2/0/0.0
set protocols ospf area 0.0.0.0 interface xe-2/0/2.0
set protocols ldp interface xe-2/0/0.0
set protocols ldp interface xe-2/0/2.0
set protocols ldp interface lo0.0

```

```
set policy-options policy-statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pplb then load-balance per-packet
set routing-instances evpn1 instance-type virtual-switch
set routing-instances evpn1 interface xe-2/0/6.1
set routing-instances evpn1 interface ae0.1
set routing-instances evpn1 interface ae1.1
set routing-instances evpn1 route-distinguisher 1:10
set routing-instances evpn1 vrf-target target:1:5
set routing-instances evpn1 protocols evpn extended-vlan-list 1
set routing-instances evpn1 protocols evpn mclag bgp-peer 198.51.100.2
set routing-instances evpn1 switch-options service-id 1
set routing-instances evpn1 vlans v1 vlan-id 1
set routing-instances evpn1 vlans v1 l3-interface irb.1
set routing-instances evpn1 vlans v1 no-arp-suppression
set routing-instances evpn2 instance-type virtual-switch
set routing-instances evpn2 interface xe-2/0/6.2
set routing-instances evpn2 interface ae0.2
set routing-instances evpn2 interface ae1.2
set routing-instances evpn2 route-distinguisher 1:20
set routing-instances evpn2 vrf-target target:1:6
set routing-instances evpn2 protocols evpn extended-vlan-list 2
set routing-instances evpn2 protocols evpn mclag bgp-peer 198.51.100.2
set routing-instances evpn2 switch-options service-id 2
set routing-instances evpn2 vlans v1 vlan-id 2
set routing-instances evpn2 vlans v1 l3-interface irb.2
set routing-instances evpn2 vlans v1 no-arp-suppression
set routing-instances evpn3 instance-type virtual-switch
set routing-instances evpn3 interface xe-2/0/6.3
set routing-instances evpn3 interface ae0.3
set routing-instances evpn3 interface ae1.3
set routing-instances evpn3 route-distinguisher 1:30
set routing-instances evpn3 vrf-target target:1:7
set routing-instances evpn3 protocols evpn extended-vlan-list 3
set routing-instances evpn3 protocols evpn mclag bgp-peer 198.51.100.2
set routing-instances evpn3 switch-options service-id 3
set routing-instances evpn3 vlans v1 vlan-id 3
set routing-instances evpn3 vlans v1 l3-interface irb.3
set routing-instances evpn3 vlans v1 no-arp-suppression
```

PE2: MC-LAG Configuration

```
set chassis aggregated-devices ethernet device-count 3
set interfaces xe-2/0/1 gigether-options 802.3ad ae0
set interfaces xe-2/0/6 enable
set interfaces xe-2/0/6 flexible-vlan-tagging
set interfaces xe-2/0/6 encapsulation flexible-ethernet-services
set interfaces xe-2/0/6 unit 1 family ethernet-switching interface-mode trunk
set interfaces xe-2/0/6 unit 1 family ethernet-switching vlan members 1
set interfaces xe-2/0/6 unit 2 family ethernet-switching interface-mode trunk
set interfaces xe-2/0/6 unit 2 family ethernet-switching vlan members 2
set interfaces xe-2/0/6 unit 3 family ethernet-switching interface-mode trunk
set interfaces xe-2/0/6 unit 3 family ethernet-switching vlan members 3
set interfaces xe-2/1/0 unit 0 family inet address 203.0.113.2/24
set interfaces xe-2/1/1 gigether-options 802.3ad ae1
set interfaces xe-2/1/2 gigether-options 802.3ad ae1
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp periodic fast
set interfaces ae0 aggregated-ether-options lacp system-id 00:00:11:11:11:11
set interfaces ae0 aggregated-ether-options lacp admin-key 1
set interfaces ae0 aggregated-ether-options mc-ae mc-ae-id 1
set interfaces ae0 aggregated-ether-options mc-ae redundancy-group 2
set interfaces ae0 aggregated-ether-options mc-ae chassis-id 1
set interfaces ae0 aggregated-ether-options mc-ae mode active-active
set interfaces ae0 aggregated-ether-options mc-ae status-control standby
set interfaces ae0 unit 1 esi 00:11:22:33:44:55:66:77:88:99
set interfaces ae0 unit 1 esi all-active
set interfaces ae0 unit 1 family ethernet-switching interface-mode trunk
set interfaces ae0 unit 1 family ethernet-switching vlan members 1
set interfaces ae0 unit 2 esi 00:11:11:11:11:11:11:11:11:11
set interfaces ae0 unit 2 esi all-active
set interfaces ae0 unit 2 family ethernet-switching interface-mode trunk
set interfaces ae0 unit 2 family ethernet-switching vlan members 2
set interfaces ae0 unit 3 esi 00:11:22:22:22:22:22:22:22:22
set interfaces ae0 unit 3 esi all-active
set interfaces ae0 unit 3 family ethernet-switching interface-mode trunk
set interfaces ae0 unit 3 family ethernet-switching vlan members 3
set interfaces ae1 flexible-vlan-tagging
set interfaces ae1 encapsulation flexible-ethernet-services
set interfaces ae1 aggregated-ether-options lacp active
```

```

set interfaces ae1 unit 1 family ethernet-switching interface-mode trunk
set interfaces ae1 unit 1 family ethernet-switching vlan members 1
set interfaces ae1 unit 2 family ethernet-switching interface-mode trunk
set interfaces ae1 unit 2 family ethernet-switching vlan members 2
set interfaces ae1 unit 3 family ethernet-switching interface-mode trunk
set interfaces ae1 unit 3 family ethernet-switching vlan members 3
set multi-chassis multi-chassis-protection 203.0.113.1 interface ae1
set protocols iccp local-ip-addr 203.0.113.2
set protocols iccp peer 203.0.113.1 session-establishment-hold-time 600
set protocols iccp peer 203.0.113.1 redundancy-group-id-list 2
set protocols iccp peer 203.0.113.1 liveness-detection minimum-interval 10000
set protocols iccp peer 203.0.113.1 liveness-detection multiplier 3

```

PE2: EVPN-MPLS Configuration

```

set interfaces xe-2/0/0 unit 0 family inet address 192.0.2.222/24
set interfaces xe-2/0/0 unit 0 family mpls
set interfaces xe-2/0/2 unit 0 family inet address 192.0.2.22/24
set interfaces xe-2/0/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 198.51.100.2/32 primary
set interfaces irb unit 1 family inet address 10.2.1.2/24 virtual-gateway-address 10.2.1.254
set interfaces irb unit 2 family inet address 10.2.2.2/24 virtual-gateway-address 10.2.2.254
set interfaces irb unit 3 family inet address 10.2.3.2/24 virtual-gateway-address 10.2.3.254
set routing-options router-id 198.51.100.2
set routing-options autonomous-system 65000
set routing-options forwarding-table export evpn-pplb
set protocols mpls interface xe-2/0/2.0
set protocols mpls interface xe-2/0/0.0
set protocols bgp group evpn type internal
set protocols bgp group evpn local-address 198.51.100.2
set protocols bgp group evpn family evpn signaling
set protocols bgp group evpn local-as 65000
set protocols bgp group evpn neighbor 198.51.100.1
set protocols bgp group evpn neighbor 198.51.100.3
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols ospf area 0.0.0.0 interface xe-2/0/0.0
set protocols ospf area 0.0.0.0 interface xe-2/0/2.0
set protocols ldp interface xe-2/0/0.0
set protocols ldp interface xe-2/0/2.0
set protocols ldp interface lo0.0
set policy-options policy-statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pplb then load-balance per-packet

```

```
set routing-instances evpn1 instance-type virtual-switch
set routing-instances evpn1 interface xe-2/0/6.1
set routing-instances evpn1 interface ae0.1
set routing-instances evpn1 interface ae1.1
set routing-instances evpn1 route-distinguisher 1:11
set routing-instances evpn1 vrf-target target:1:5
set routing-instances evpn1 protocols evpn extended-vlan-list 1
set routing-instances evpn1 protocols evpn mlag bgp-peer 198.51.100.1
set routing-instances evpn1 switch-options service-id 1
set routing-instances evpn1 vlans v1 vlan-id 1
set routing-instances evpn1 vlans v1 l3-interface irb.1
set routing-instances evpn1 vlans v1 no-arp-suppression
set routing-instances evpn2 instance-type virtual-switch
set routing-instances evpn2 interface xe-2/0/6.2
set routing-instances evpn2 interface ae0.2
set routing-instances evpn2 interface ae1.2
set routing-instances evpn2 route-distinguisher 1:21
set routing-instances evpn2 vrf-target target:1:6
set routing-instances evpn2 protocols evpn extended-vlan-list 2
set routing-instances evpn2 protocols evpn mlag bgp-peer 198.51.100.1
set routing-instances evpn2 switch-options service-id 2
set routing-instances evpn2 vlans v1 vlan-id 2
set routing-instances evpn2 vlans v1 l3-interface irb.2
set routing-instances evpn2 vlans v1 no-arp-suppression
set routing-instances evpn3 instance-type virtual-switch
set routing-instances evpn3 interface xe-2/0/6.3
set routing-instances evpn3 interface ae0.3
set routing-instances evpn3 interface ae1.3
set routing-instances evpn3 route-distinguisher 1:31
set routing-instances evpn3 vrf-target target:1:7
set routing-instances evpn3 protocols evpn extended-vlan-list 3
set routing-instances evpn3 protocols evpn mlag bgp-peer 198.51.100.1
set routing-instances evpn3 switch-options service-id 3
set routing-instances evpn3 vlans v1 vlan-id 3
set routing-instances evpn3 vlans v1 l3-interface irb.3
set routing-instances evpn3 vlans v1 no-arp-suppression
```

PE1: Configuring MC-LAG

Step-by-Step Procedure

1. Set the number of aggregated Ethernet interfaces on PE1.

```
[edit]
user@switch# set chassis aggregated-devices ethernet device-count 3
```

2. Configure aggregated Ethernet interface ae0 on interface xe-2/0/1, and configure LACP and MC-LAG on ae0. Divide aggregated Ethernet interface ae0 into three logical interfaces (ae0.1, ae0.2, and ae0.3). For each logical interface, specify an ESI, place the logical interface in MC-LAG active-active mode, and map the logical interface to a VLAN.

```
[edit]
user@switch# set interfaces xe-2/0/1 gigether-options 802.3ad ae0
user@switch# set interfaces ae0 flexible-vlan-tagging
user@switch# set interfaces ae0 encapsulation flexible-ethernet-services
user@switch# set interfaces ae0 aggregated-ether-options lacp active
user@switch# set interfaces ae0 aggregated-ether-options lacp periodic fast
user@switch# set interfaces ae0 aggregated-ether-options lacp system-id 00:00:11:11:11:11
user@switch# set interfaces ae0 aggregated-ether-options lacp admin-key 1
user@switch# set interfaces ae0 aggregated-ether-options mc-ae mc-ae-id 1
user@switch# set interfaces ae0 aggregated-ether-options mc-ae redundancy-group 2
user@switch# set interfaces ae0 aggregated-ether-options mc-ae chassis-id 0
user@switch# set interfaces ae0 aggregated-ether-options mc-ae mode active-active
user@switch# set interfaces ae0 aggregated-ether-options mc-ae status-control active
user@switch# set interfaces ae0 unit 1 esi 00:11:22:33:44:55:66:77:88:99
user@switch# set interfaces ae0 unit 1 esi all-active
user@switch# set interfaces ae0 unit 1 family ethernet-switching interface-mode trunk
user@switch# set interfaces ae0 unit 1 family ethernet-switching vlan members 1
user@switch# set interfaces ae0 unit 2 esi 00:11:11:11:11:11:11:11:11:11
user@switch# set interfaces ae0 unit 2 esi all-active
user@switch# set interfaces ae0 unit 2 family ethernet-switching interface-mode trunk
user@switch# set interfaces ae0 unit 2 family ethernet-switching vlan members 2
user@switch# set interfaces ae0 unit 3 esi 00:11:22:22:22:22:22:22:22:22
user@switch# set interfaces ae0 unit 3 esi all-active
user@switch# set interfaces ae0 unit 3 family ethernet-switching interface-mode trunk
user@switch# set interfaces ae0 unit 3 family ethernet-switching vlan members 3
```


3. Configure physical interface xe-2/0/6, and divide it into three logical interfaces (xe-2/0/6.1, xe-2/0/6.2, and xe-2/0/6.3). Map each logical interface to a VLAN.

```
[edit]
user@switch# set interfaces xe-2/0/6 enable
user@switch# set interfaces xe-2/0/6 flexible-vlan-tagging
user@switch# set interfaces xe-2/0/6 encapsulation flexible-ethernet-services
user@switch# set interfaces xe-2/0/6 unit 1 family ethernet-switching interface-mode trunk
user@switch# set interfaces xe-2/0/6 unit 1 family ethernet-switching vlan members 1
user@switch# set interfaces xe-2/0/6 unit 2 family ethernet-switching interface-mode trunk
user@switch# set interfaces xe-2/0/6 unit 2 family ethernet-switching vlan members 2
user@switch# set interfaces xe-2/0/6 unit 3 family ethernet-switching interface-mode trunk
user@switch# set interfaces xe-2/0/6 unit 3 family ethernet-switching vlan members 3
```

4. Configure physical interface xe-2/1/0 as a Layer 3 interface, on which you configure ICCP. Specify the interface with the IP address of 203.0.113.2 on PE2 as the ICCP peer to PE1.

```
[edit]
user@switch# set interfaces xe-2/1/0 unit 0 family inet address 203.0.113.1/24
user@switch# set protocols iccp local-ip-addr 203.0.113.1
user@switch# set protocols iccp peer 203.0.113.2 session-establishment-hold-time 600
user@switch# set protocols iccp peer 203.0.113.2 redundancy-group-id-list 2
user@switch# set protocols iccp peer 203.0.113.2 liveness-detection minimum-interval 10000
user@switch# set protocols iccp peer 203.0.113.2 liveness-detection multiplier 3
```

5. Configure aggregated Ethernet interface ae1 on interfaces xe-2/1/1 and xe-2/1/2, and configure LACP on ae1. Divide aggregated Ethernet interface ae1 into three logical interfaces (ae1.1, ae1.2, and ae1.3), and map each logical interface to a VLAN. Specify ae1 as the multichassis protection link between PE1 and PE2.

```
[edit]
user@switch# set interfaces xe-2/1/1 gigether-options 802.3ad ae1
user@switch# set interfaces xe-2/1/2 gigether-options 802.3ad ae1
user@switch# set interfaces ae1 flexible-vlan-tagging
user@switch# set interfaces ae1 encapsulation flexible-ethernet-services
user@switch# set interfaces ae1 aggregated-ether-options lacp active
user@switch# set interfaces ae1 unit 1 family ethernet-switching interface-mode trunk
user@switch# set interfaces ae1 unit 1 family ethernet-switching vlan members 1
user@switch# set interfaces ae1 unit 2 family ethernet-switching interface-mode trunk
user@switch# set interfaces ae1 unit 2 family ethernet-switching vlan members 2
```

```

user@switch# set interfaces ae1 unit 3 family ethernet-switching interface-mode trunk
user@switch# set interfaces ae1 unit 3 family ethernet-switching vlan members 3
user@switch# set multi-chassis multi-chassis-protection 203.0.113.2 interface ae1

```

PE1: Configuring EVPN-MPLS

Step-by-Step Procedure

1. Configure the loopback interface, and the interfaces connected to the other PE devices.

```

[edit]
user@switch# set interfaces lo0 unit 0 family inet address 198.51.100.1/32 primary
user@switch# set interfaces xe-2/0/0 unit 0 family inet address 192.0.2.2/24
user@switch# set interfaces xe-2/0/0 unit 0 family mpls
user@switch# set interfaces xe-2/0/2 unit 0 family inet address 192.0.2.111/24
user@switch# set interfaces xe-2/0/2 unit 0 family mpls

```

2. Configure IRB interfaces irb.1, irb.2, and irb.3.

```

[edit]
user@switch# set interfaces irb unit 1 family inet address 10.2.1.1/24 virtual-gateway-address 10.2.1.254
user@switch# set interfaces irb unit 2 family inet address 10.2.2.1/24 virtual-gateway-address 10.2.2.254
user@switch# set interfaces irb unit 3 family inet address 10.2.3.1/24 virtual-gateway-address 10.2.3.254

```

3. Assign a router ID and the autonomous system in which PE1, PE2, and PE3 reside.

```

[edit]
user@switch# set routing-options router-id 198.51.100.1
user@switch# set routing-options autonomous-system 65000

```

4. Enable per-packet load-balancing for EVPN routes when EVPN multihoming active-active mode is used.

```

[edit]
user@switch# set routing-options forwarding-table export evpn-pplb

```

```

user@switch# set policy-options policy-statement evpn-pplb from protocol evpn
user@switch# set policy-options policy-statement evpn-pplb then load-balance per-packet

```

5. Enable MPLS on interfaces xe-2/0/0.0 and xe-2/0/2.0.

```

[edit]
user@switch# set protocols mpls interface xe-2/0/0.0
user@switch# set protocols mpls interface xe-2/0/2.0

```

6. Configure an IBGP overlay that includes PE1, PE2, and PE3.

```

[edit]
user@switch# set protocols bgp group evpn type internal
user@switch# set protocols bgp group evpn local-address 198.51.100.1
user@switch# set protocols bgp group evpn family evpn signaling
user@switch# set protocols bgp group evpn local-as 65000
user@switch# set protocols bgp group evpn neighbor 198.51.100.2
user@switch# set protocols bgp group evpn neighbor 198.51.100.3

```

7. Configure OSPF as the internal routing protocol for EVPN by specifying an area ID and interfaces on which EVPN-MPLS is enabled.

```

[edit]
user@switch# set protocols ospf area 0.0.0.0 interface lo0.0
user@switch# set protocols ospf area 0.0.0.0 interface xe-2/0/0.0
user@switch# set protocols ospf area 0.0.0.0 interface xe-2/0/2.0

```

8. Configure the Label Distribution Protocol (LDP) on the loopback interface and the interfaces on which EVPN-MPLS is enabled.

```

[edit]
user@switch# set protocols ldp interface lo0.0
user@switch# set protocols ldp interface xe-2/0/0.0
user@switch# set protocols ldp interface xe-2/0/2.0

```

9. Configure virtual switch routing instances for VLAN v1, which is assigned VLAN IDs of 1, 2, and 3, and include the interfaces and other entities associated with the VLAN.

```
[edit]
user@switch# set routing-instances evpn1 instance-type virtual-switch
user@switch# set routing-instances evpn1 interface xe-2/0/6.1
user@switch# set routing-instances evpn1 interface ae0.1
user@switch# set routing-instances evpn1 interface ae1.1
user@switch# set routing-instances evpn1 route-distinguisher 1:10
user@switch# set routing-instances evpn1 vrf-target target:1:5
user@switch# set routing-instances evpn1 protocols evpn extended-vlan-list 1
user@switch# set routing-instances evpn1 protocols evpn mclag bgp-peer 198.51.100.2
user@switch# set routing-instances evpn1 switch-options service-id 1
user@switch# set routing-instances evpn1 vlans v1 vlan-id 1
user@switch# set routing-instances evpn1 vlans v1 l3-interface irb.1
user@switch# set routing-instances evpn1 vlans v1 no-arp-suppression
user@switch# set routing-instances evpn2 instance-type virtual-switch
user@switch# set routing-instances evpn2 interface xe-2/0/6.2
user@switch# set routing-instances evpn2 interface ae0.2
user@switch# set routing-instances evpn2 interface ae1.2
user@switch# set routing-instances evpn2 route-distinguisher 1:20
user@switch# set routing-instances evpn2 vrf-target target:1:6
user@switch# set routing-instances evpn2 protocols evpn extended-vlan-list 2
user@switch# set routing-instances evpn2 protocols evpn mclag bgp-peer 198.51.100.2
user@switch# set routing-instances evpn2 switch-options service-id 2
user@switch# set routing-instances evpn2 vlans v1 vlan-id 2
user@switch# set routing-instances evpn2 vlans v1 l3-interface irb.2
user@switch# set routing-instances evpn2 vlans v1 no-arp-suppression
user@switch# set routing-instances evpn3 instance-type virtual-switch
user@switch# set routing-instances evpn3 interface xe-2/0/6.3
user@switch# set routing-instances evpn3 interface ae0.3
user@switch# set routing-instances evpn3 interface ae1.3
user@switch# set routing-instances evpn3 route-distinguisher 1:30
user@switch# set routing-instances evpn3 vrf-target target:1:7
user@switch# set routing-instances evpn3 protocols evpn extended-vlan-list 3
user@switch# set routing-instances evpn3 protocols evpn mclag bgp-peer 198.51.100.2
user@switch# set routing-instances evpn3 switch-options service-id 3
user@switch# set routing-instances evpn3 vlans v1 vlan-id 3
user@switch# set routing-instances evpn3 vlans v1 l3-interface irb.3
user@switch# set routing-instances evpn3 vlans v1 no-arp-suppression
```

PE2: Configuring MC-LAG

Step-by-Step Procedure

1. Set the number of aggregated Ethernet interfaces on PE2.

```
[edit]
user@switch# set chassis aggregated-devices ethernet device-count 3
```

2. Configure aggregated Ethernet interface ae0 on interface xe-2/0/1, and configure LACP and MC-LAG on ae0. Divide aggregated Ethernet interface ae0 into three logical interfaces (ae0.1, ae0.2, and ae0.3). For each logical interface, specify an ESI, place the logical interface in MC-LAG active-active mode, and map the logical interface to a VLAN.

```
[edit]
user@switch# set interfaces xe-2/0/1 gigether-options 802.3ad ae0
user@switch# set interfaces ae0 flexible-vlan-tagging
user@switch# set interfaces ae0 encapsulation flexible-ethernet-services
user@switch# set interfaces ae0 aggregated-ether-options lacp active
user@switch# set interfaces ae0 aggregated-ether-options lacp periodic fast
user@switch# set interfaces ae0 aggregated-ether-options lacp system-id 00:00:11:11:11:11
user@switch# set interfaces ae0 aggregated-ether-options lacp admin-key 1
user@switch# set interfaces ae0 aggregated-ether-options mc-ae mc-ae-id 1
user@switch# set interfaces ae0 aggregated-ether-options mc-ae redundancy-group 2
user@switch# set interfaces ae0 aggregated-ether-options mc-ae chassis-id 1
user@switch# set interfaces ae0 aggregated-ether-options mc-ae mode active-active
user@switch# set interfaces ae0 aggregated-ether-options mc-ae status-control standby
user@switch# set interfaces ae0 unit 1 esi 00:11:22:33:44:55:66:77:88:99
user@switch# set interfaces ae0 unit 1 esi all-active
user@switch# set interfaces ae0 unit 1 family ethernet-switching interface-mode trunk
user@switch# set interfaces ae0 unit 1 family ethernet-switching vlan members 1
user@switch# set interfaces ae0 unit 2 esi 00:11:11:11:11:11:11:11:11:11
user@switch# set interfaces ae0 unit 2 esi all-active
user@switch# set interfaces ae0 unit 2 family ethernet-switching interface-mode trunk
user@switch# set interfaces ae0 unit 2 family ethernet-switching vlan members 2
user@switch# set interfaces ae0 unit 3 esi 00:11:22:22:22:22:22:22:22:22
user@switch# set interfaces ae0 unit 3 esi all-active
user@switch# set interfaces ae0 unit 3 family ethernet-switching interface-mode trunk
user@switch# set interfaces ae0 unit 3 family ethernet-switching vlan members 3
```

3. Configure physical interface xe-2/0/6, and divide it into three logical interfaces (xe-2/0/6.1, xe-2/0/6.2, and xe-2/0/6.3). Map each logical interface to a VLAN.

```
[edit]
set interfaces xe-2/0/6 enable
set interfaces xe-2/0/6 flexible-vlan-tagging
set interfaces xe-2/0/6 encapsulation flexible-ethernet-services
set interfaces xe-2/0/6 unit 1 family ethernet-switching interface-mode trunk
set interfaces xe-2/0/6 unit 1 family ethernet-switching vlan members 1
set interfaces xe-2/0/6 unit 2 family ethernet-switching interface-mode trunk
set interfaces xe-2/0/6 unit 2 family ethernet-switching vlan members 2
set interfaces xe-2/0/6 unit 3 family ethernet-switching interface-mode trunk
set interfaces xe-2/0/6 unit 3 family ethernet-switching vlan members 3
```

4. Configure physical interface xe-2/1/0 as a Layer 3 interface, on which you configure ICCP. Specify the interface with the IP address of 203.0.113.1 on PE1 as the ICCP peer to PE2.

```
[edit]
set interfaces xe-2/1/0 unit 0 family inet address 203.0.113.2/24
set protocols iccp local-ip-addr 203.0.113.2
set protocols iccp peer 203.0.113.1 session-establishment-hold-time 600
set protocols iccp peer 203.0.113.1 redundancy-group-id-list 2
set protocols iccp peer 203.0.113.1 liveness-detection minimum-interval 10000
set protocols iccp peer 203.0.113.1 liveness-detection multiplier 3
```

5. Configure aggregated Ethernet interface ae1 on interfaces xe-2/1/1 and xe-2/1/2, and configure LACP on ae1. Divide aggregated Ethernet interface ae1 into three logical interfaces (ae1.1, ae1.2, and ae1.3), and map each logical interface to a VLAN. Specify ae1 as the multichassis protection link between PE1 and PE2.

```
[edit]
set interfaces xe-2/1/1 gigether-options 802.3ad ae1
set interfaces xe-2/1/2 gigether-options 802.3ad ae1
set interfaces ae1 flexible-vlan-tagging
set interfaces ae1 encapsulation flexible-ethernet-services
set interfaces ae1 aggregated-ether-options lacp active
set interfaces ae1 unit 1 family ethernet-switching interface-mode trunk
set interfaces ae1 unit 1 family ethernet-switching vlan members 1
set interfaces ae1 unit 2 family ethernet-switching interface-mode trunk
set interfaces ae1 unit 2 family ethernet-switching vlan members 2
```

```

set interfaces ae1 unit 3 family ethernet-switching interface-mode trunk
set interfaces ae1 unit 3 family ethernet-switching vlan members 3
set multi-chassis multi-chassis-protection 203.0.113.1 interface ae1

```

PE2: Configuring EVPN-MPLS

Step-by-Step Procedure

1. Configure the loopback interface, and the interfaces connected to the other PE devices.

```

[edit]
user@switch# set interfaces lo0 unit 0 family inet address 198.51.100.2/32 primary
user@switch# set interfaces xe-2/0/0 unit 0 family inet address 192.0.2.222/24
user@switch# set interfaces xe-2/0/0 unit 0 family mpls
user@switch# set interfaces xe-2/0/2 unit 0 family inet address 192.0.2.22/24
user@switch# set interfaces xe-2/0/2 unit 0 family mpls

```

2. Configure IRB interfaces irb.1, irb.2, and irb.3.

```

[edit]
user@switch# set interfaces irb unit 1 family inet address 10.2.1.2/24 virtual-gateway-
address 10.2.1.254
user@switch# set interfaces irb unit 2 family inet address 10.2.2.2/24 virtual-gateway-
address 10.2.2.254
user@switch# set interfaces irb unit 3 family inet address 10.2.3.2/24 virtual-gateway-
address 10.2.3.254

```

3. Assign a router ID and the autonomous system in which PE1, PE2, and PE3 reside.

```

[edit]
user@switch# set routing-options router-id 198.51.100.2
user@switch# set routing-options autonomous-system 65000

```

4. Enable per-packet load-balancing for EVPN routes when EVPN multihoming active-active mode is used.

```

[edit]
user@switch# set routing-options forwarding-table export evpn-pplb

```

```

user@switch# set policy-options policy-statement evpn-pplb from protocol evpn
user@switch# set policy-options policy-statement evpn-pplb then load-balance per-packet

```

5. Enable MPLS on interfaces xe-2/0/0.0 and xe-2/0/2.0.

```

[edit]
user@switch# set protocols mpls interface xe-2/0/0.0
user@switch# set protocols mpls interface xe-2/0/2.0

```

6. Configure an IBGP overlay that includes PE1, PE2, and PE3.

```

[edit]
user@switch# set protocols bgp group evpn type internal
user@switch# set protocols bgp group evpn local-address 198.51.100.2
user@switch# set protocols bgp group evpn family evpn signaling
user@switch# set protocols bgp group evpn local-as 65000
user@switch# set protocols bgp group evpn neighbor 198.51.100.1
user@switch# set protocols bgp group evpn neighbor 198.51.100.3

```

7. Configure OSPF as the internal routing protocol for EVPN by specifying an area ID and interfaces on which EVPN-MPLS is enabled.

```

[edit]
user@switch# set protocols ospf area 0.0.0.0 interface lo0.0
user@switch# set protocols ospf area 0.0.0.0 interface xe-2/0/0.0
user@switch# set protocols ospf area 0.0.0.0 interface xe-2/0/2.0

```

8. Configure the Label Distribution Protocol (LDP) on the loopback interface and the interfaces on which EVPN-MPLS is enabled.

```

[edit]
user@switch# set protocols ldp interface lo0.0
user@switch# set protocols ldp interface xe-2/0/0.0
user@switch# set protocols ldp interface xe-2/0/2.0

```


9. Configure virtual switch routing instances for VLAN v1, which is assigned VLAN IDs of 1, 2, and 3, and include the interfaces and other entities associated with the VLAN.

```
[edit]
user@switch# set routing-instances evpn1 instance-type virtual-switch
user@switch# set routing-instances evpn1 interface xe-2/0/6.1
user@switch# set routing-instances evpn1 interface ae0.1
user@switch# set routing-instances evpn1 interface ae1.1
user@switch# set routing-instances evpn1 route-distinguisher 1:11
user@switch# set routing-instances evpn1 vrf-target target:1:5
user@switch# set routing-instances evpn1 protocols evpn extended-vlan-list 1
user@switch# set routing-instances evpn1 protocols evpn mclag bgp-peer 198.51.100.1
user@switch# set routing-instances evpn1 switch-options service-id 1
user@switch# set routing-instances evpn1 vlans v1 vlan-id 1
user@switch# set routing-instances evpn1 vlans v1 l3-interface irb.1
user@switch# set routing-instances evpn1 vlans v1 no-arp-suppression
user@switch# set routing-instances evpn2 instance-type virtual-switch
user@switch# set routing-instances evpn2 interface xe-2/0/6.2
user@switch# set routing-instances evpn2 interface ae0.2
user@switch# set routing-instances evpn2 interface ae1.2
user@switch# set routing-instances evpn2 route-distinguisher 1:21
user@switch# set routing-instances evpn2 vrf-target target:1:6
user@switch# set routing-instances evpn2 protocols evpn extended-vlan-list 2
user@switch# set routing-instances evpn2 protocols evpn mclag bgp-peer 198.51.100.1
user@switch# set routing-instances evpn2 switch-options service-id 2
user@switch# set routing-instances evpn2 vlans v1 vlan-id 2
user@switch# set routing-instances evpn2 vlans v1 l3-interface irb.2
user@switch# set routing-instances evpn2 vlans v1 no-arp-suppression
user@switch# set routing-instances evpn3 instance-type virtual-switch
user@switch# set routing-instances evpn3 interface xe-2/0/6.3
user@switch# set routing-instances evpn3 interface ae0.3
user@switch# set routing-instances evpn3 interface ae1.3
user@switch# set routing-instances evpn3 route-distinguisher 1:31
user@switch# set routing-instances evpn3 vrf-target target:1:7
user@switch# set routing-instances evpn3 protocols evpn extended-vlan-list 3
user@switch# set routing-instances evpn3 protocols evpn mclag bgp-peer 198.51.100.1
user@switch# set routing-instances evpn3 switch-options service-id 3
user@switch# set routing-instances evpn3 vlans v1 vlan-id 3
user@switch# set routing-instances evpn3 vlans v1 l3-interface irb.3
user@switch# set routing-instances evpn3 vlans v1 no-arp-suppression
```

PE3 Configuration

IN THIS SECTION

- [CLI Quick Configuration | 1407](#)
- [PE3: Configuring EVPN-MPLS | 1409](#)

CLI Quick Configuration

PE3: EVPN-MPLS Configuration

```

set interfaces lo0 unit 0 family inet address 198.51.100.3/32 primary
set interfaces xe-2/0/2 unit 0 family inet address 192.0.2.1/24
set interfaces xe-2/0/2 unit 0 family mpls
set interfaces xe-2/0/3 unit 0 family inet address 192.0.2.11/24
set interfaces xe-2/0/3 unit 0 family mpls
set interfaces xe-2/0/6 enable
set interfaces xe-2/0/6 flexible-vlan-tagging
set interfaces xe-2/0/6 encapsulation flexible-ethernet-services
set interfaces xe-2/0/6 unit 1 family ethernet-switching interface-mode trunk
set interfaces xe-2/0/6 unit 1 family ethernet-switching vlan members 1
set interfaces xe-2/0/6 unit 2 family ethernet-switching interface-mode trunk
set interfaces xe-2/0/6 unit 2 family ethernet-switching vlan members 2
set interfaces xe-2/0/6 unit 3 family ethernet-switching interface-mode trunk
set interfaces xe-2/0/6 unit 3 family ethernet-switching vlan members 3
set interfaces irb unit 1 family inet address 10.2.1.3/24 virtual-gateway-address 10.2.1.254
set interfaces irb unit 2 family inet address 10.2.2.3/24 virtual-gateway-address 10.2.2.254
set interfaces irb unit 3 family inet address 10.2.3.3/24 virtual-gateway-address 10.2.3.254
set routing-options router-id 198.51.100.3
set routing-options autonomous-system 65000
set routing-options forwarding-table export evpn-pplb
set protocols mpls interface xe-2/0/2.0
set protocols mpls interface xe-2/0/3.0
set protocols bgp group evpn type internal
set protocols bgp group evpn local-address 198.51.100.3
set protocols bgp group evpn family evpn signaling
set protocols bgp group evpn local-as 65000
set protocols bgp group evpn neighbor 198.51.100.1
set protocols bgp group evpn neighbor 198.51.100.2

```

```
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols ospf area 0.0.0.0 interface xe-2/0/2.0
set protocols ospf area 0.0.0.0 interface xe-2/0/3.0
set protocols ldp interface lo0.0
set protocols ldp interface xe-2/0/2.0
set protocols ldp interface xe-2/0/3.0
set policy-options policy-statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pplb then load-balance per-packet
set routing-instances evpn1 instance-type virtual-switch
set routing-instances evpn1 interface xe-2/0/6.1
set routing-instances evpn1 route-distinguisher 1:12
set routing-instances evpn1 vrf-target target:1:5
set routing-instances evpn1 protocols evpn extended-vlan-list 1
set routing-instances evpn1 switch-options service-id 1
set routing-instances evpn1 vlans v1 vlan-id 1
set routing-instances evpn1 vlans v1 l3-interface irb.1
set routing-instances evpn1 vlans v1 no-arp-suppression
set routing-instances evpn2 instance-type virtual-switch
set routing-instances evpn2 interface xe-2/0/6.2
set routing-instances evpn2 route-distinguisher 1:22
set routing-instances evpn2 vrf-target target:1:6
set routing-instances evpn2 protocols evpn extended-vlan-list 2
set routing-instances evpn2 switch-options service-id 2
set routing-instances evpn2 vlans v1 vlan-id 2
set routing-instances evpn2 vlans v1 l3-interface irb.2
set routing-instances evpn2 vlans v1 no-arp-suppression
set routing-instances evpn3 instance-type virtual-switch
set routing-instances evpn3 interface xe-2/0/6.3
set routing-instances evpn3 route-distinguisher 1:32
set routing-instances evpn3 vrf-target target:1:7
set routing-instances evpn3 protocols evpn extended-vlan-list 3
set routing-instances evpn3 switch-options service-id 3
set routing-instances evpn3 vlans v1 vlan-id 3
set routing-instances evpn3 vlans v1 l3-interface irb.3
set routing-instances evpn3 vlans v1 no-arp-suppression
```

PE3: Configuring EVPN-MPLS

Step-by-Step Procedure

1. Configure the loopback interface, and the interfaces connected to the other PE devices.

```
[edit]
user@switch# set interfaces lo0 unit 0 family inet address 198.51.100.3/32 primary
user@switch# set interfaces xe-2/0/2 unit 0 family inet address 192.0.2.1/24
user@switch# set interfaces xe-2/0/2 unit 0 family mpls
user@switch# set interfaces xe-2/0/3 unit 0 family inet address 192.0.2.11/24
user@switch# set interfaces xe-2/0/3 unit 0 family mpls
```

2. Configure interface xe-2/0/6, which is connected to the host.

```
[edit]
user@switch# set interfaces xe-2/0/6 enable
user@switch# set interfaces xe-2/0/6 flexible-vlan-tagging
user@switch# set interfaces xe-2/0/6 encapsulation flexible-ethernet-services
user@switch# set interfaces xe-2/0/6 unit 1 family ethernet-switching interface-mode trunk
user@switch# set interfaces xe-2/0/6 unit 1 family ethernet-switching vlan members 1
user@switch# set interfaces xe-2/0/6 unit 2 family ethernet-switching interface-mode trunk
user@switch# set interfaces xe-2/0/6 unit 2 family ethernet-switching vlan members 2
user@switch# set interfaces xe-2/0/6 unit 3 family ethernet-switching interface-mode trunk
user@switch# set interfaces xe-2/0/6 unit 3 family ethernet-switching vlan members 3
```

3. Configure IRB interfaces irb.1, irb.2, and irb.3.

```
[edit]
user@switch# set interfaces irb unit 1 family inet address 10.2.1.3/24 virtual-gateway-
address 10.2.1.254
user@switch# set interfaces irb unit 2 family inet address 10.2.2.3/24 virtual-gateway-
address 10.2.2.254
user@switch# set interfaces irb unit 3 family inet address 10.2.3.3/24 virtual-gateway-
address 10.2.3.254
```

4. Assign a router ID and the autonomous system in which PE1, PE2, and PE3 reside.

```
[edit]
user@switch# set routing-options router-id 198.51.100.3
user@switch# set routing-options autonomous-system 65000
```

5. Enable per-packet load-balancing for EVPN routes when EVPN multihoming active-active mode is used.

```
[edit]
user@switch# set routing-options forwarding-table export evpn-pplb
user@switch# set policy-options policy-statement evpn-pplb from protocol evpn
user@switch# set policy-options policy-statement evpn-pplb then load-balance per-packet
```

6. Enable MPLS on interfaces xe-2/0/2.0 and xe-2/0/3.0.

```
[edit]
user@switch# set protocols mpls interface xe-2/0/2.0
user@switch# set protocols mpls interface xe-2/0/3.0
```

7. Configure an IBGP overlay that includes PE1, PE2, and PE3.

```
[edit]
user@switch# set protocols bgp group evpn type internal
user@switch# set protocols bgp group evpn local-address 198.51.100.3
user@switch# set protocols bgp group evpn family evpn signaling
user@switch# set protocols bgp group evpn local-as 65000
user@switch# set protocols bgp group evpn neighbor 198.51.100.1
user@switch# set protocols bgp group evpn neighbor 198.51.100.2
```

8. Configure OSPF as the internal routing protocol for EVPN by specifying an area ID and interfaces on which EVPN-MPLS is enabled.

```
[edit]
user@switch# set protocols ospf area 0.0.0.0 interface lo0.0
user@switch# set protocols ospf area 0.0.0.0 interface xe-2/0/2.0
user@switch# set protocols ospf area 0.0.0.0 interface xe-2/0/3.0
```

9. Configure the LDP on the loopback interface and the interfaces on which EVPN-MPLS is enabled.

```
[edit]
user@switch# set protocols ldp interface lo0.0
user@switch# set protocols ldp interface xe-2/0/2.0
user@switch# set protocols ldp interface xe-2/0/3.0
```

10. Configure virtual switch routing instances for VLAN v1, which is assigned VLAN IDs of 1, 2, and 3, and include the interfaces and other entities associated with the VLAN.

```
[edit]
user@switch# set routing-instances evpn1 instance-type virtual-switch
user@switch# set routing-instances evpn1 interface xe-2/0/6.1
user@switch# set routing-instances evpn1 route-distinguisher 1:12
user@switch# set routing-instances evpn1 vrf-target target:1:5
user@switch# set routing-instances evpn1 protocols evpn extended-vlan-list 1
user@switch# set routing-instances evpn1 switch-options service-id 1
user@switch# set routing-instances evpn1 vlans v1 vlan-id 1
user@switch# set routing-instances evpn1 vlans v1 l3-interface irb.1
user@switch# set routing-instances evpn1 vlans v1 no-arp-suppression
user@switch# set routing-instances evpn2 instance-type virtual-switch
user@switch# set routing-instances evpn2 interface xe-2/0/6.2
user@switch# set routing-instances evpn2 route-distinguisher 1:22
user@switch# set routing-instances evpn2 vrf-target target:1:6
user@switch# set routing-instances evpn2 protocols evpn extended-vlan-list 2
user@switch# set routing-instances evpn2 switch-options service-id 2
user@switch# set routing-instances evpn2 vlans v1 vlan-id 2
user@switch# set routing-instances evpn2 vlans v1 l3-interface irb.2
user@switch# set routing-instances evpn2 vlans v1 no-arp-suppression
user@switch# set routing-instances evpn3 instance-type virtual-switch
user@switch# set routing-instances evpn3 interface xe-2/0/6.3
user@switch# set routing-instances evpn3 route-distinguisher 1:32
user@switch# set routing-instances evpn3 vrf-target target:1:7
user@switch# set routing-instances evpn3 protocols evpn extended-vlan-list 3
user@switch# set routing-instances evpn3 switch-options service-id 3
user@switch# set routing-instances evpn3 vlans v1 vlan-id 3
user@switch# set routing-instances evpn3 vlans v1 l3-interface irb.3
user@switch# set routing-instances evpn3 vlans v1 no-arp-suppression
```

RELATED DOCUMENTATION

| [Understanding EVPN-MPLS Interworking with Junos Fusion Enterprise and MC-LAG](#) | 1359

7

PART

PBB-EVPN

[Configuring PBB-EVPN Integration | 1414](#)

[Configuring MAC Pinning for PBB-EVPNs | 1520](#)

Configuring PBB-EVPN Integration

IN THIS CHAPTER

- [Provider Backbone Bridging \(PBB\) and EVPN Integration Overview | 1414](#)
- [Example: Configuring PBB with Single-Homed EVPN | 1450](#)
- [Example: Configuring PBB with Multihomed EVPN | 1478](#)

Provider Backbone Bridging (PBB) and EVPN Integration Overview

IN THIS SECTION

- [Technology Overview of PBB-EVPN Integration | 1415](#)
- [Implementation Overview of PBB-EVPN Integration | 1437](#)
- [Configuration Overview of PBB-EVPN Integration | 1444](#)
- [Supported and Unsupported Features on PBB-EVPN | 1449](#)

Ethernet VPN (EVPN) provides a solution for multipoint Layer 2 VPN services with advanced multihoming capabilities using BGP for distributing MAC address reachability information over the core MPLS or IP network. However, with EVPN, several thousands of MAC addresses are carried from each virtual routing and forwarding (VRF) instance, requiring frequent updates on newly learned MAC routes and withdrawn routes. This increases the overhead on the provider network.

Provider backbone bridging (PBB) extends Layer 2 Ethernet switching to provide enhanced scalability, quality-of-service (QoS) features, and carrier-class reliability. With the integration of PBB with EVPN, instead of sending the customer MAC (C-MAC) addresses as control plane learning, the backbone MAC (B-MAC) addresses are distributed in the EVPN core. This simplifies the control plane learning across the core and allows a huge number of Layer 2 services, such as data center connectivity, to transit the network in a simple manner.

The following sections describe the technology and implementation overview of PBB-EVPN integration:

Technology Overview of PBB-EVPN Integration

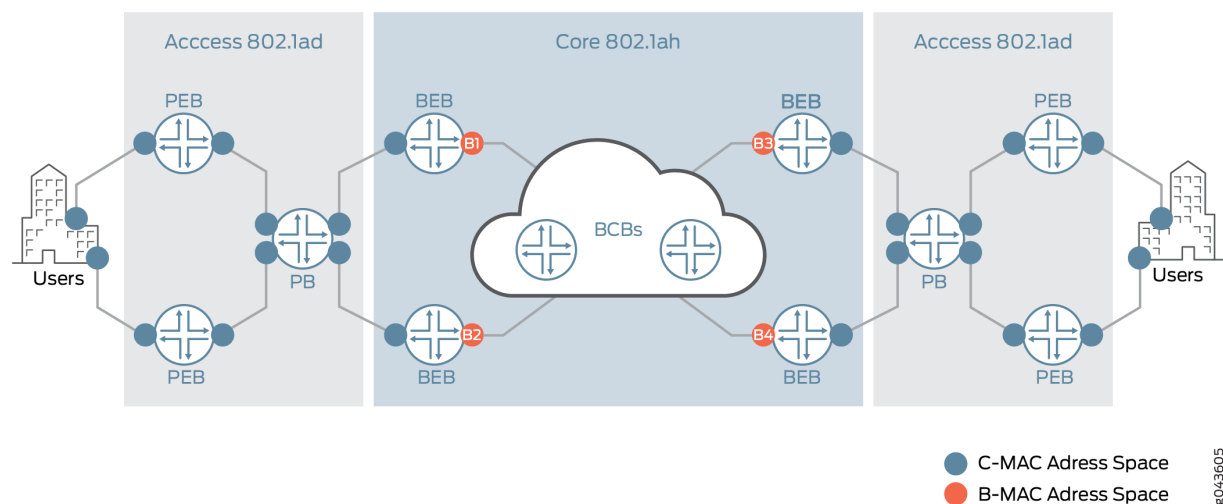
Understanding Provider Backbone Bridging (PBB)

Provider backbone bridging (PBB) was originally defined as IEEE 802.1ah standard, and operates in exactly the same manner as IEEE 802.1ad standard. However, instead of multiplexing VLANs, PBB duplicates the MAC layer of the customer frame and separates it from the provider domain, by encapsulating it in a 24 bit instance service identifier (I-SID). This allows for complete transparency between the customer network and the provider network.

When operating on customer MAC (C-MAC) and service MAC (S-MAC) addresses, PBB uses a new backbone MAC (B-MAC) address. The B-MAC address is added at the edge of the PBB network, that is administered by a carrier VPN or a carrier-of-carriers VPN. With the use of an I-SID for the customer routing instance (I-component) service group, PBB improves the scalability of Ethernet services.

Figure 141 on page 1415 illustrates a PBB network, describing the PBB network elements and MAC address spaces.

Figure 141: PBB Network Elements



The PBB terms are:

- **PB**—Provider bridge (802.1ad)
- **PEB**—Provider edge bridge (802.1ad)
- **BEB**—Backbone edge bridge (802.1ah)
- **BCB**—Backbone core bridge (802.1ah)

The BEB device is the first immediate point of interest within PBB, and forms the boundary between the access network and the core. This introduces two key components—the I-component and B-component in PBB.

- **I-component**

The I-component forms the customer or access facing interface or routing instance. The I-component is responsible for mapping customer Ethernet traffic to the appropriate I-SID. At first, the customer Ethernet traffic is mapped to a customer bridge domain. Then each customer bridge domain is mapped to an I-SID. This service mapping can be per port, per port with service VLAN (S-VLAN), or per port with S-VLAN and customer VLAN (C-VLAN). The I-component is used to learn and forward frames based on the C-MAC addresses, and maintains a C-MAC-to-B-MAC mapping table that is based on instance tag (I-TAG).

Within the I-component there are two ports:

- Customer Instance Port (CIP)

These ports are customer service instances at the customer-facing interfaces. Service definitions can be per port, per port with S-VLAN, or per port with S-VLAN and C-VLAN.

- Provider Instance Port (PIP)

This port performs PBB encapsulation, such as pushing the I-TAG, source and destination B-MAC addresses, and PBB decapsulation, such as popping the I-SID, learning source B-MAC-to-C-MAC mapping, in the ingress direction.

- **B-component**

the B-component is the backbone-facing PBB core instance. The B-component is used to learn and forward packets based on the B-MAC addresses. The B-component is then responsible for mapping the I-SIDs to the appropriate B-VLANs (in the case of PBB networks) or pushing and popping service MPLS labels for MPLS-based networks.

Within the B-component there are two ports:

- Customer Backbone Port (CBP)

These ports are backbone edge ports that can receive and transmit instance-tagged frames from multiple customers, and assign backbone VLAN IDs (B-VIDs) and translate the I-SID on the basis of the received I-SID.

- Provider Backbone Port (PBP)

These ports provide connectivity to the other bridges within and attached to the backbone. These are provider-facing ports. These ports support the S-VLAN component.

Figure 142 on page 1417 illustrates the key components of PBB. Figure 143 on page 1418 illustrates the PBB packet format.

Figure 142: PBB Key Components

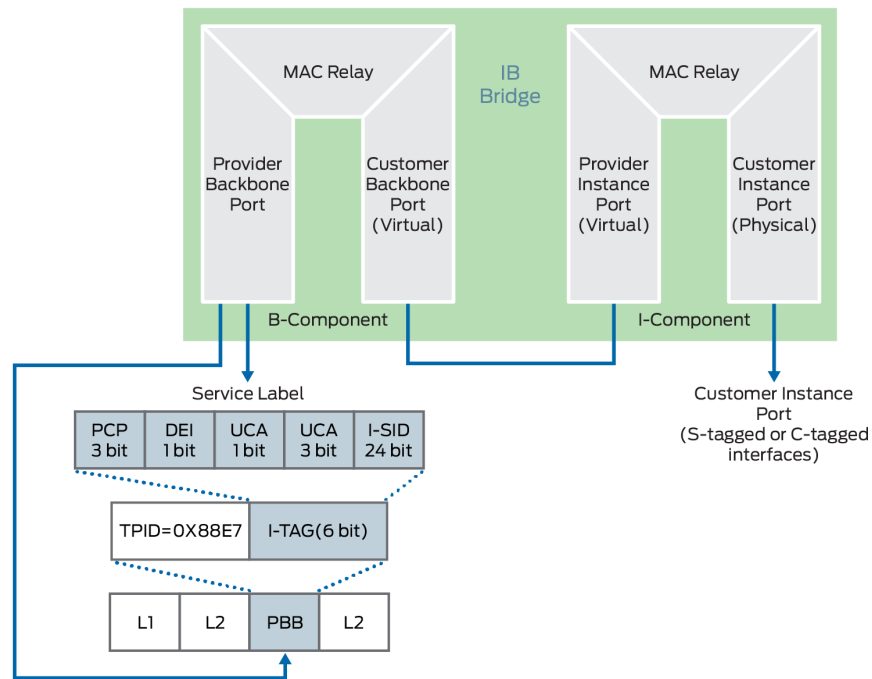
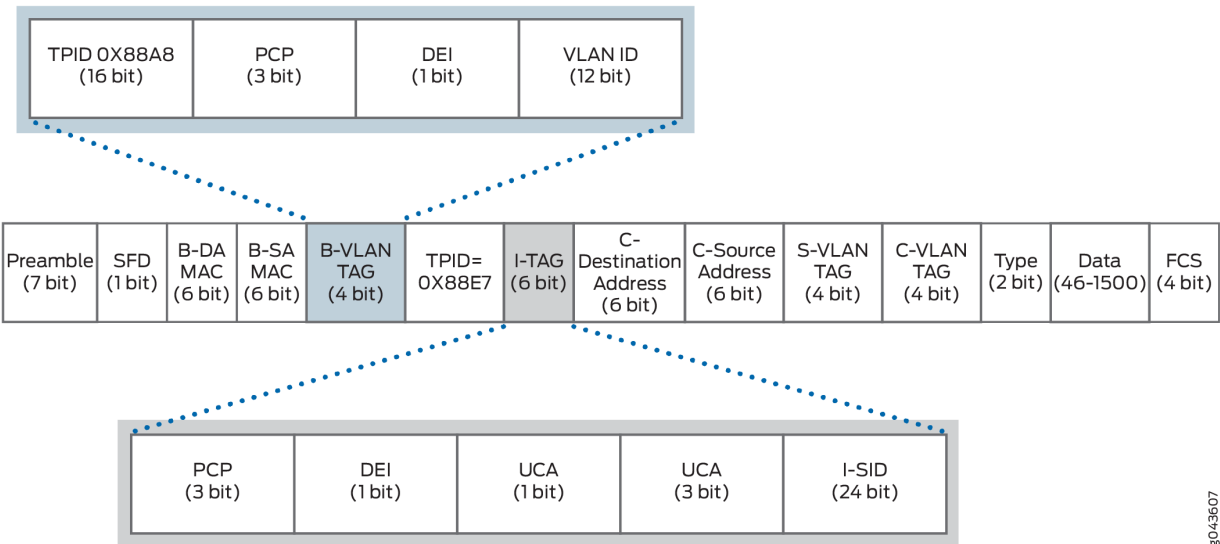


Figure 143: PBB Packet Format



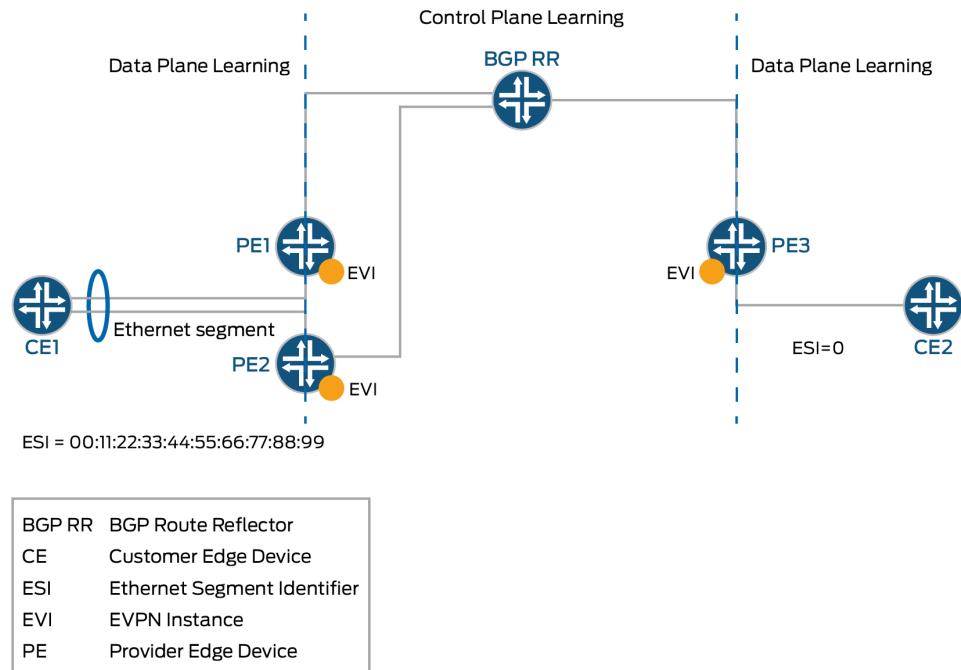
8043607

Understanding EVPN

EVPN is a new standards-based technology that provides virtual multipoint bridged connectivity between different Layer 2 domains over an IP or IP/MPLS backbone network. Similar to other VPN technologies, such as IPVPN and VPLS, EVPN instances (EVIs) are configured on PE routers to maintain logical service separation between customers. The provider edge (PE) devices connect to customer edge (CE) devices, which can be a router, switch, or host. The PE devices then exchange reachability information using MultiProtocol BGP (MP-BGP) and encapsulated traffic is forwarded between them.

Because elements of the architecture are common with other VPN technologies, EVPN can be seamlessly introduced and integrated into existing service environments.

Figure 144: EVPN Overview



The EVPN technology provides mechanisms for next-generation DCI by adding extended control plane procedures to exchange the Layer 2 (MAC address) and Layer 3 (IP address) information among the participating Data Center Border Routers (DCBRs). These features help address some of the DCI challenges, such as seamless VM mobility and optimal IP routing. Seamless VM mobility refers to the challenge of Layer 2 extension and maintaining connectivity in the face of VM mobility, and optimal IP routing refers to the challenge of supporting default gateway behavior for a VM's outbound traffic and triangular routing avoidance of a VM's inbound traffic.

The EVPN technology is used by the data center operator to offer multi-tenancy, flexible, and resilient services that can be extended on demand. This flexibility and resiliency can require using compute resources among different physical data centers for a single service (Layer 2 extension) and VM motion.

EVPN supports all-active multihoming, which allows a CE device to connect to two or more PE devices such that traffic is forwarded using all of the links between the devices. This enables the CE device to load-balance traffic to the multiple PE devices. More importantly it allows a remote PE device to load-balance traffic to the multihomed PEs across the core network. This load balancing of traffic flows between data centers is known as aliasing. EVPN also has mechanisms that prevent the looping of broadcast, unknown unicast, and multicast (BUM) traffic in an all-active multihomed topology.

Multihoming provides redundancy in the event that an access link or a PE device fails. In either case, traffic flows from the CE device toward the PE device use the remaining active links. For traffic in the other direction, the remote PE device updates its forwarding table to send traffic to the remaining active PE devices connected to the multihomed Ethernet segment. EVPN provides a fast convergence mechanism so that the time it takes to make this adjustment is independent of the number of MAC addresses learned by the PE device.

EVPN's MP-BGP control plane allows live virtual machines to be dynamically moved from one data center to another, also known as VM motion. After a VM is moved to a destination server/hypervisor, it transmits a gratuitous ARP that updates the Layer 2 forwarding table of the PE device at the destination data center. The PE device then transmits a MAC route update to all remote PE devices which in turn update their forwarding tables. In this manner, an EVPN tracks the movement of the VM, also known as MAC Mobility. EVPN also has mechanisms to detect and stop MAC flapping.

The EVPN technology, similar to Layer 3 MPLS VPN, introduces the concept of routing MAC addresses using MP-BGP over the MPLS core. Some of the important benefits of using EVPNs include:

- Ability to have a dual-active multihomed edge device
- Provides load balancing across dual-active links
- Provides MAC address mobility
- Provides multi-tenancy
- Provides aliasing
- Enables fast convergence

PBB-EVPN Integration

The integration of PBB with EVPN is described in the following sections:

Integrating the PBB and EVPN Network Elements

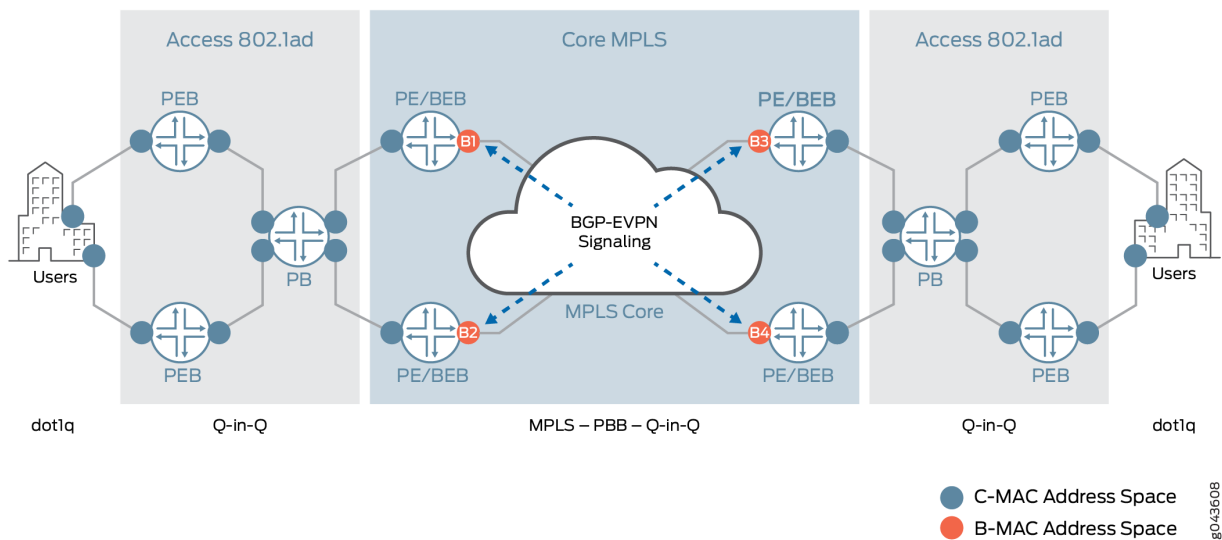
In a PBB network, a huge amount of customer MAC (C-MAC) addresses are hid behind a drastically smaller number of backbone MAC (B-MAC) addresses, without the devices in the core having to learn and process all the individual customer states. The I-SID creates an encapsulation that enables a large number of services to be deployed. However, unlike modern networks that have a simple MPLS core composed of PE and provider devices, the devices in the PBB core need to act as switches, called the backbone core bridge (BCB), performing forwarding decisions based on B-MAC addresses. This causes incompatibility issues with modern MPLS networks, where packets are switched between edge loopback addresses using MPLS labels and recursion.

With the integration of PBB with EVPN, the BCB element in the PBB core is replaced with MPLS, while retaining the service scaling properties of the BEB PBB edge device. The B-component is signaled using

EVPN BGP signaling and encapsulated inside MPLS using PE and provider devices. As a result, the vast scale of PBB is combined with the simplicity of a traditional basic MPLS core network and the amount of network-wide state information is significantly reduced, as opposed to regular PBB.

Figure 145 on page 1421 illustrates the PBB-EVPN integration using the different elements of a PBB and EVPN network.

Figure 145: PBB-EVPN Integration

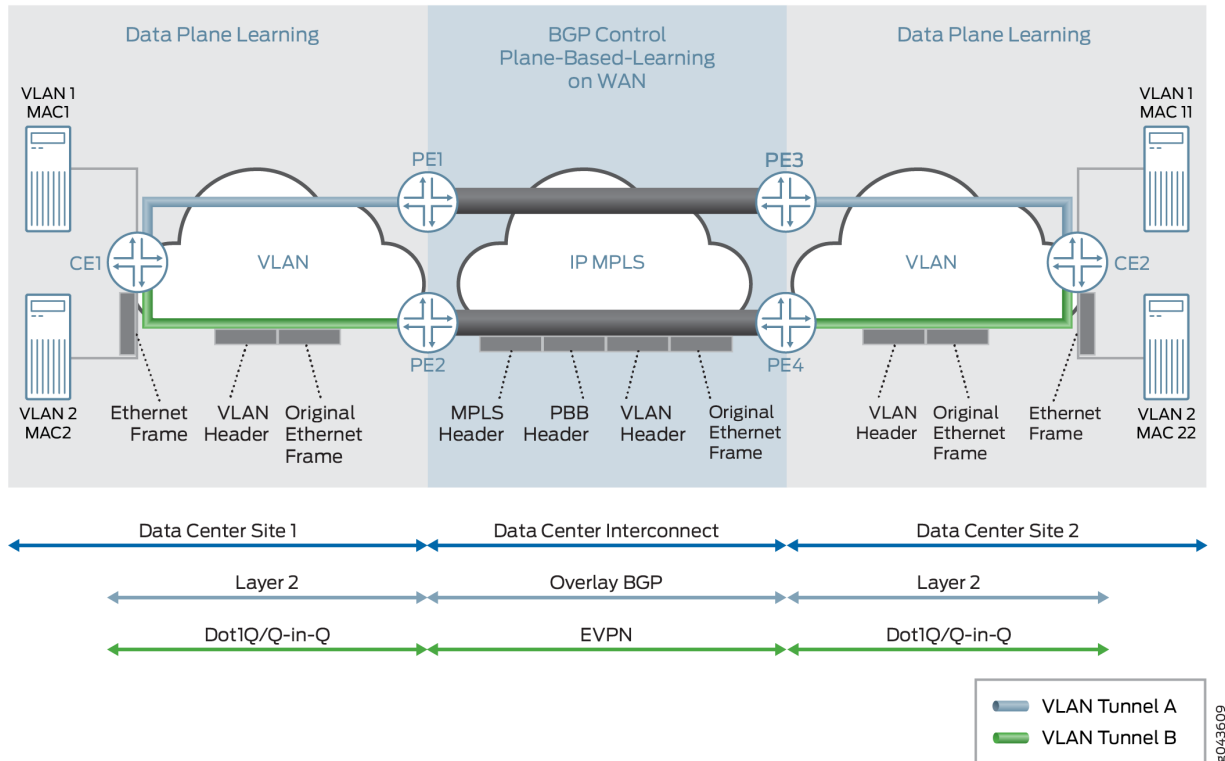


PBB-EVPN Control Plane Initialization

In a PBB-EVPN network, B-MAC addresses are distributed over the EVPN core and C-MAC addresses are learned in the data plane and aggregated behind the B-MAC addresses.

Figure 146 on page 1422 illustrates the control plane handling in a sample PBB-EVPN network with four PE devices and two top-of-rack devices across two data centers.

Figure 146: PBB-EVPN Control Plane Handling



The control plane handling at the Data Center Site 1 is as follows:

1. The C-MAC address lookup occurs and the C-MAC address is learned.
2. The B-MAC source address and I-SID are pushed on the packet.
3. Destination address lookup of C-MAC to B-MAC is done in the I-SID table. If the MAC address is present, the packet is routed using an EVPN MAC route; otherwise a multicast route is used.
4. This route gives the service label for the packet, which has PBB and also the original frame.

The control plane handling at the Data Center Site 2 is as follows:

1. At the destination PE device, the packet is received with a single service label, indicating that it is a PBB frame.
2. The source address allocation of C-MAC to B-MAC is learned in the I-SID table.
3. The source address of the C-MAC is learned in the customer bridge domain (C-BD) MAC table.

Discovery of EVPN Routes in PBB-EVPN

PBB with Dot1ah functionality is implemented at the PE devices. In the case of PBB-EVPN, the PE devices implement the instance and backbone bridge functionality. Only B-MAC addresses are distributed in the control plane, the C-MAC addresses are learned in the data plane. The following EVPN routes are discovered on the different PE devices:

VPN Autodiscovery

When an EVPN instance (EVI) is configured on different PE devices, autodiscovery of the VPN happens first for discovering the EVPN endpoints. Each PE device that is configured with an EVI sends the inclusive multicast route.

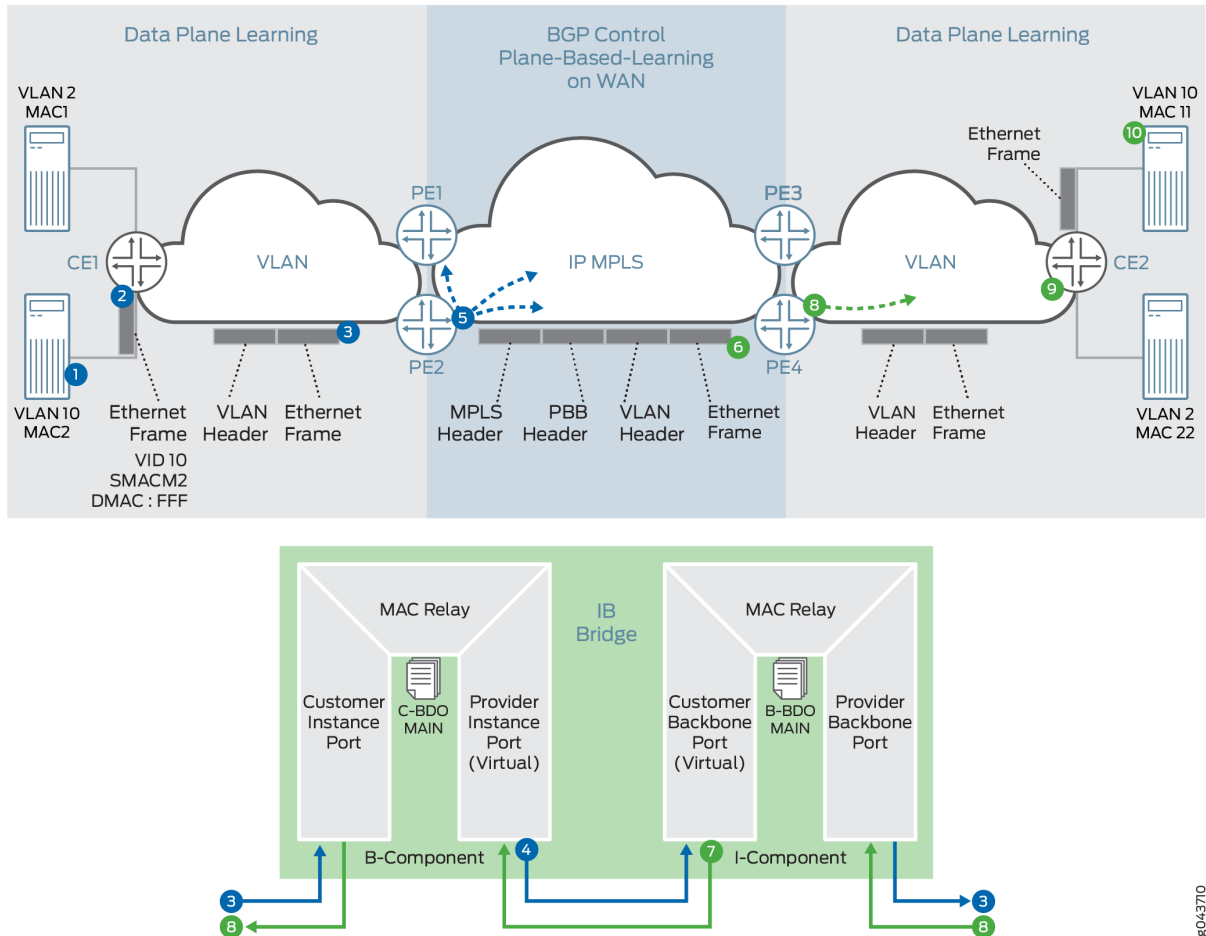
The inclusive multicast route fields are as follows:

- **RD**—Unique route distinguisher value per advertising PE device per EVI. The significance of the RD is local to a PE device.
- **TAG ID**—Service ID equivalent to the I-SID value. One I-SID is assigned to one bridge domain under an EVI when service is supported. The TAG ID is set to 0 for the I-SID bundle service, where multiple I-SIDs are mapped to one bridge domain.
- **Originating IP addr**—Loopback IP address.
- **P-Multicast Service Interface (PMSI) Attributes**—Attributes required for transmitting the BUM traffic. There are two type of tunnels—point-to-multipoint LSP and ingress replication. In the case of ingress replication, the multicast label for BUM traffic is downstream assigned. In the case of point-to-multipoint LSP, the PMSI attribute includes the point-to-multipoint LSP identifier. If the multicast tree is shared or aggregated among multiple EVIs, the PE device uses the upstream assigned label to associate or bind to the EVI.
- **RT Extended Community**—Route target associated with an EVI. This attribute is of global significance in EVPN.

In [Figure 147 on page 1424](#), each PE device sends the inclusive multicast route to each BGP neighbor. Device PE1 sends an inclusive multicast route to Devices PE2, PE3, and PE4 for VPN autodiscovery. The

handling of BUM traffic is also illustrated in the figure. During the startup sequence, Devices PE1, PE2, PE3, and PE4 send inclusive multicast route that include the multicast label.

Figure 147: VPN Autodiscovery



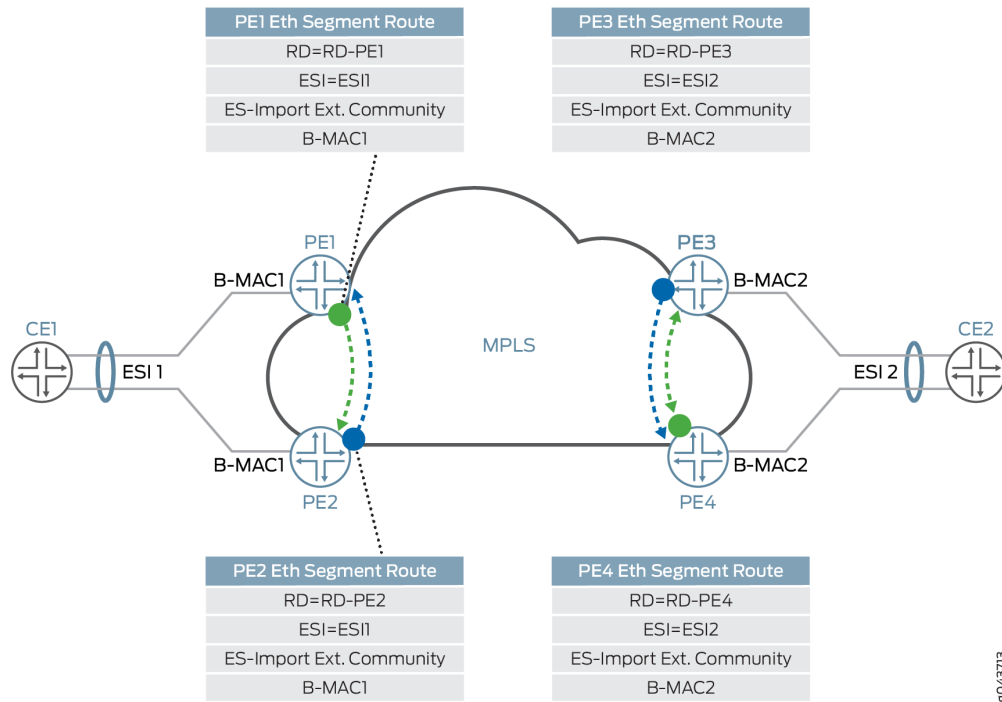
Ethernet Segment Discovery

The Ethernet segment route is encoded in the EVPN NLRI using the Route Type of value 4. This NLRI is used for discovering the multihomed Ethernet segment and for DF election.

ES-import route target, a transitive route target extended community, is also carried with the Ethernet segment route. ES-import extended community enables all the PE devices connected to the same multihomed site to import the Ethernet segment routes. The import of this route is done by the PE device that has the ESI configured. All other PE devices discard this Ethernet segment route.

Figure 148 on page 1425 provide details about the procedure of Ethernet segment route for multihomed Ethernet segment autodiscovery.

Figure 148: Ethernet Segment Discovery



In this figure, Devices PE1 and PE2 are connected to a multihomed segment with ESI value of ESI1 and B-MAC address of B-MAC1. In the case of an active-active multihomed segment, this B-MAC should be the on Devices PE1 and PE2. Similarly, Devices PE3 and PE4 are active/active multihomed for ESI2 with B-MAC address of B-MAC2. Devices PE1 and PE2 send the Ethernet segment route for ESI1, which is received by Devices PE3 and PE4, but is ignored because the devices are not configured for ESI1. Only Devices PE1 and PE2 are in one redundant group and the DF election is performed in this group. Similarly, Devices PE3 and PE4 are in another redundant group and either Device PE3 or PE4 is selected as the DF.

Ethernet MAC Routes Discovery

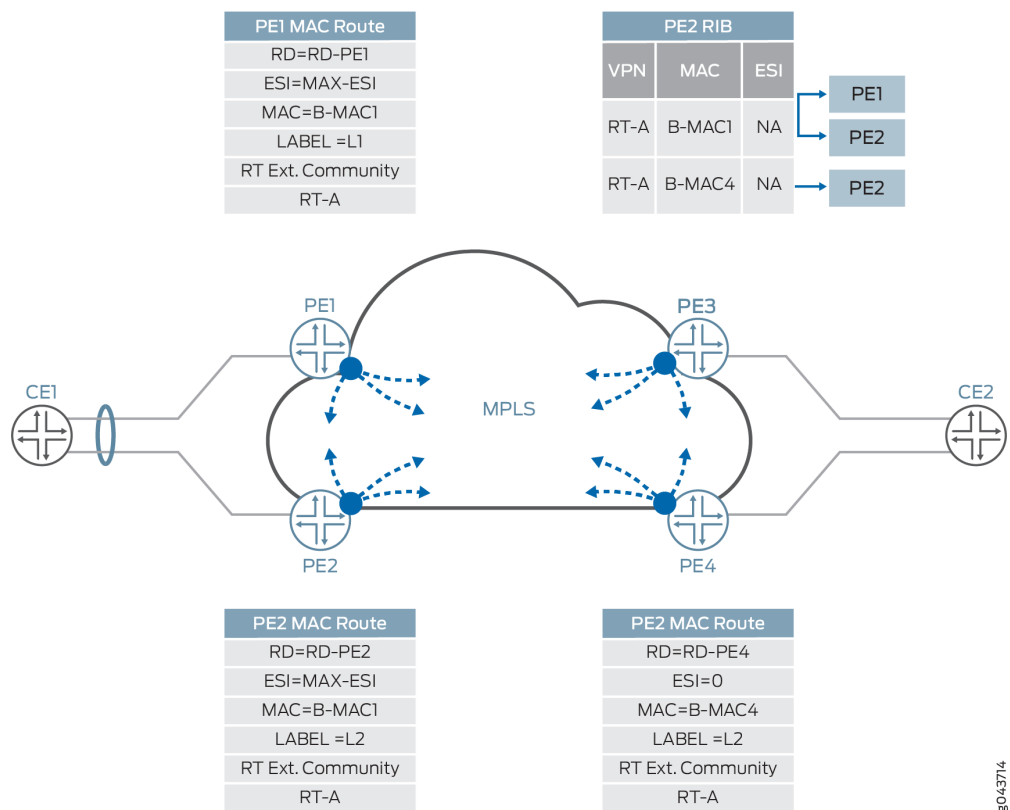
Ethernet MAC Advertisement route is used for distributing B-MAC addresses of PE nodes. The MAC advertisement route is encoded with following fields:

- MAC address field contains B-MAC address.
- Ethernet Tag field is set to 0.

- Ethernet segment identifier field must be set to 0 (for single-homed segments or multihomed segments with per-I-SID load balancing) or to MAX-ESI (for multihomed segment with per-flow load balancing).
- Label is associated with unicast forwarding of traffic from different PE devices.
- RT (route target) extended community associated with its EVI.

Figure 149 on page 1426 illustrates the MAC route advertisement in PBB-EVPN.

Figure 149: Ethernet MAC Routes Discovery



Differences Between PBB-EVPN and EVPN

Table 47 on page 1427 and Table 48 on page 1427 list the differences between PBB-EVPN and pure EVPN for Layer 2 networks in the context of different route types and route attributes, respectively.

Table 47: Route Differences Between PBB-EVPN and EVPN

Route	Usage	Applicability
Ethernet autodiscovery route	<ul style="list-style-type: none"> • MAC mass withdrawal • Aliasing • Advertising split horizon labels 	EVPN only
MAC advertisement route	<ul style="list-style-type: none"> • Advertise MAC address reachability • Advertise IP and MAC bindings 	EVPN PBB-EVPN
Inclusive multicast route	Multicast tunnel endpoint discovery	EVPN PBB-EVPN
Ethernet segment route	<ul style="list-style-type: none"> • Redundancy • Designated forwarder (DF) election 	EVPN PBB-EVPN

Table 48: Route Attributes and Route Usage Differences Between PBB-EVPN and EVPN

Attribute	Usage	Applicability
ESI MPLS lable extended community	<ul style="list-style-type: none"> • Encode split horizon label for the Ethernet segment. • Indicate redundancy mode (active/standby or active/active). 	Ethernet autodiscovery route
ES-import extended community	Limit the import scope of the Ethernet segment routes.	Ethernet segment route
MAC mobility extended community	<ul style="list-style-type: none"> • EVPN: Indicates that a MAC address has moved from one segment to another across PE devices. • PBB-EVPN: Signal C-MAC address flush notification. 	MAC advertisement route

Table 48: Route Attributes and Route Usage Differences Between PBB-EVPN and EVPN *(Continued)*

Attribute	Usage	Applicability
Default gateway extended community	Indicate the MAC or IP bindings of a gateway.	MAC advertisement route

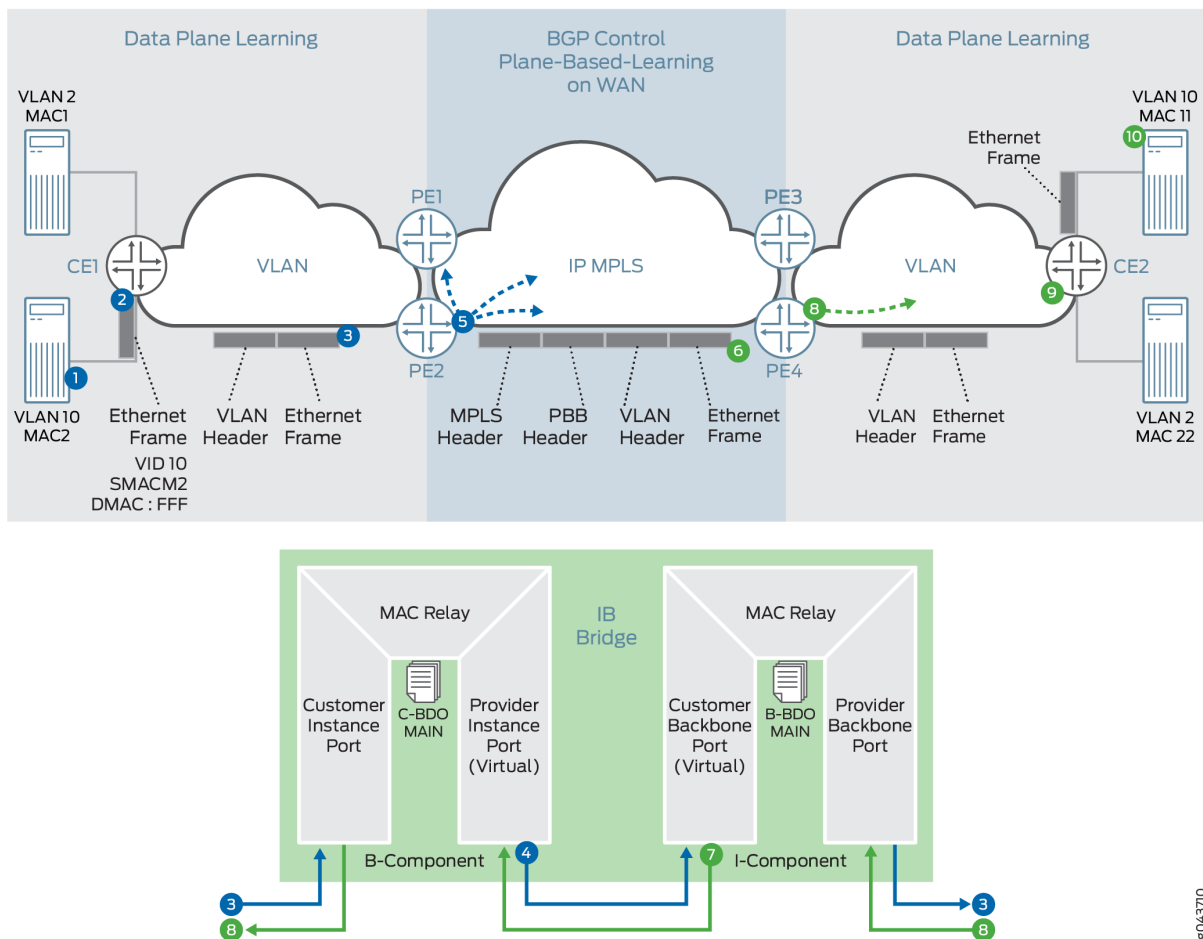
PBB-EVPN Packet Walkthrough

Based on the PBB and EVPN configuration on different PE devices of the network, the Ethernet segment, B-MAC address reachability, and multicast routes are already programmed on different PE devices in the EVPN cloud. The packet walkthrough of PBB-EVPN includes the handling of the following traffic types:

Handling BUM Traffic in PBB-EVPN

Figure 150 on page 1429 illustrates the PBB-EVPN control plane and BUM traffic handling.

Figure 150: PBB-EVPN BUM Traffic Handling



The PBB-EVPN handling of BUM traffic over the EVPN cloud is as follows:

1. After Server A boots up, Server A attempts to send traffic to Server B. Server A does not have the ARP binding in the ARP table for Server B, so Server A builds an ARP broadcast request and sends it. The contents of the ARP packets are VLAN 10, S-MAC=M2 (server A interface MAC), Destination MAC=ff.ff.ff.ff.ff, source IP address=IP address of Server A or VM IP address, and destination IP address=IP address of Server B. Ether type of packet for ARP is 0x0806. Layer 2 frame is sent to Device CE1.
2. Device CE1 does the Layer 2 switching operation on this frame. Because this is a Layer 2 broadcast frame, the frame is classified to an interface and based on the bridge domain configuration for this

service and broadcast behavior. The packet is forwarded to all the members of the bridge domain except to the one on which it was received. There might be VLAN translation, such as push, pop, or translate, done on this frame. This frame is sent over to Device PE2. This frame might be untagged, single tagged, or Q-in-Q.

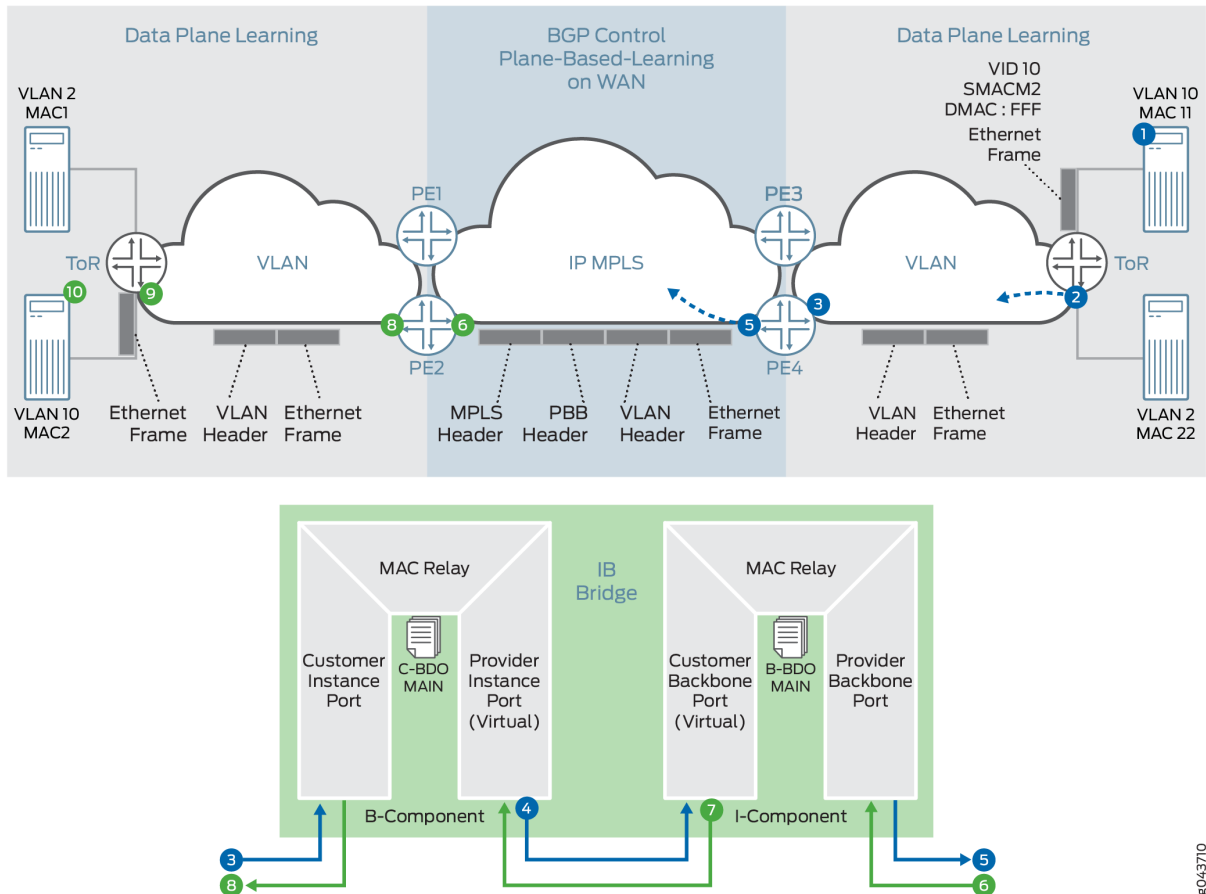
3. After Device PE2 receives this frame, at first it goes through the classification engine to classify the frame into a service. Based on the classification result interface (that is, the Customer Instance Port [CIP]) the service is identified. The source MAC address is learned (if it is not present in the MAC table). This classification results in the C-BD. Because this frame is a broadcast frame, it is sent to all the member interfaces of this bridge domain. One of the member interfaces of this bridge domain is the Provider Instance Port (PIP) interface. Now the packet is formed based on the I-SID configured for this PIP interface. The outer header of the packet at the PIP egress interface is formed based on the following information:
 - I-SID—Configured I-SID value on this PIP interface.
 - Source MAC address—B-MAC address configured or autogenerated for this frame.
 - Destination MAC address—Based on the per-I-SID mapping table that is built based on the source C-MAC-to-B-MAC address learning and the destination C-MAC-to-B-MAC address. For BUM traffic, the default value of the bridge destination address (B-DA) is the Backbone Service Instance Group address. When the B-DA of a frame is a Backbone Service Instance Group address, the normal behavior is to deliver the frame to all Customer Backbone Ports (CBPs) reachable within the backbone VLAN (B-VLAN) to which the backbone service instance is mapped. Filtering based on I-SID by the egress CBP ensures that frames are not transmitted by CBPs that are not part of the backbone service instance.
 - Layer 2 Ethernet type—0x88E7.
 - Payload—Customer frame.
4. The I-SID-formed packet is sent to the CBP for identifying the backbone bridge domain (B-BD) associated with the I-SID.
5. Lookup in the B-BD is done to send the packet to the right destination. Because this frame is a broadcast frame and the destination B-MAC is a multicast address (00-1E-83-<ISID value>), the packet needs to be handled as the ingress replication (that is, VPLS edge flood next hop) for EVPN. This next hop pushes the service label (multicast MPLS label associated with B-VLAN per peer ID and bridge VLAN ID). The MPLS packet is formed and sent over the MPLS cloud for Devices PE1, PE3 and PE4.
6. The frame is received by Device PE4 as a MPLS packet. The bridge domain identification is accomplished by doing an MPLS label L1 lookup in the mpls.0 routing table. The MPLS lookup points to the table next hop for the bridge domain next hop. After the bridge domain is identified, the packet is identified as a broadcast packet. The BUM composite flood next hop is executed, and this next hop points to the CBP.

7. The egress interfaces are identified. One of the egress interfaces is a PIP interface where the I-SID is configured, and I-SID based filtering (MAC filtering) is applied for filtering the frame. The source C-MAC-to-B-MAC address is learned for the I-SID MAC mapping table. This table is used for building the destination B-MAC address for unicast traffic. The outer I-SID header is popped from the customer Layer 2 frame. The customer bridge domain (C-BD) is found based on the I-SID classification to the PIP interface.
8. The source C-MAC address is learned. The destination C-MAC lookup is done. This is a broadcast frame, and based on the BUM handling (flood next hop), the frame is forwarded to all the member of the C-BD, except the member interface on which this frame was received.
9. Device CE2 receives this frame. Service classification is done based on the frame VLAN. Based on the classification, the bridge domain forwarding service is found and MAC learning is done. Because the frame is a broadcast frame, it is handled by flood next hop.
10. Server B receives the ARP request packet and sends the ARP reply to Server A.

Handling Unicast Traffic in PBB-EVPN

Figure 151 on page 1432 illustrates the PBB-EVPN control plane and unicast traffic handling in the form of an ARP reply from Server B.

Figure 151: PBB-EVPN Unicast Traffic Handling



For the unicast traffic flow, it is assumed that both the data and control plane MAC learning has already happened.

1. Server B generates an ARP reply. The contents of the ARP packets are VLAN 10, S-MAC=MAC11 (Server B interface MAC), destination MAC=MACA, source IP address=IP address of Server B or VM IP address, and destination IP address=IP address of Server A. This frame is forwarded to top-of-rack B.
2. After receiving the frame, the CE device classifies the incoming frame. Based on the interface family, the bridge domain associated with the interface is identified. Source MAC address learning

takes place on the bridge domain. Next the bridge domain destination MAC (MACA) lookup is done, and the lookup provides the Layer 2 egress interface. The output feature of the egress interface is applied before the CE device sends the frame to the egress interface.

3. Layer 2 encapsulated frame is received by Device PE4. The Layer 2 service classification is done to identify the customer bridge domain (C-BD) associated with this frame. The source MAC address (MAC11) learning is done in the context of the C-BD on the CIP interface.
4. Destination MAC lookup in the context of C-BD points to the PIP interface. At this point, the PIP interface egress feature list is executed. Based on the feature list, the outer I-SID header is pushed on the original Ethernet frame.
 - Source MAC—B-MAC of Device PE4
 - Destination MAC—B-MAC of Device PE2 (lookup result in I-SID C-MAC-to-B-MAC table)
 - I-SID—Configured value of I-SID
 - Layer 2 Ether typ—0x88E7
5. The destination MAC address (B-MAC of Device PE2) lookup is done in the B-BD MAC address table. This lookup results in a unicast next hop (that is, an EVPN next hop). This next hop contains a unicast MPLS service label. This label is distributed through the multiprotocol BGP (MP-BGP) control plane. The downstream peer allocates this MPLS service label. Allocation of this label can be per EVI; per EVI and VLAN; per EVI, VLAN, and attachment circuit; or per MAC address. Based on the information in the next hop, the MPLS packet is formed and forwarded on the MPLS network.
6. Device PE2 receives the frame. It is identified as an MPLS packet. The MPLS label lookup is done in the mpls.0 routing table. This lookup results in the table next hop. This lookup results in the B-BD table. The B-MAC rule (that is, source B-MAC is destination B-MAC) and I-SID filtering (CBP configured ISID=packet ISID) rules are applied. Based on the received frame I-SID, CBP is identified and B-VLAN is popped.
7. The frame header is passed to the PIP interface for further processing. The C-MAC address (M11 to B-MAC-PE2) to B-MAC mapping is learned in the I-SID table. The outer I-SID header is popped.
8. The inner source MAC address is learned on the PIP interface in the context of C-BD. The inner destination MAC address lookup is done, resulting in the egress CIP interface.
9. The CE device receives the Layer 2 frame, and Layer 2 forwarding is done.
10. Server A receives the unicast ARP reply packet from Server B.

Handling Path Forwarding in PBB-EVPN

In a PBB-EVPN network, a frame can come from either the customer edge (CE) side (bridge interface) or the MPLS-enabled interface (core-facing interface).

The packet flow for packets received from the CE side is as follows:

1. If the frame is received from a CE interface, the interface belongs to the bridge family, and the MAC address lookup and learning are done in the customer bridge domain (C-BD) context. The result of the lookup is a unicast MAC route or a flood MAC route.
2. The next lookup is done in the I-SID MAC table to determine the destination B-MAC associated with the destination C-MAC.
3. The I-SID header is prepended to the packet.
4. The next lookup is done in the B-BD because the PIP interface belongs to the bridge family.
5. The B-BD lookup points to either the unicast MAC route or the flood MAC route and this route points to either the EVPN indirect multicast next hop or the unicast indirect next hop.

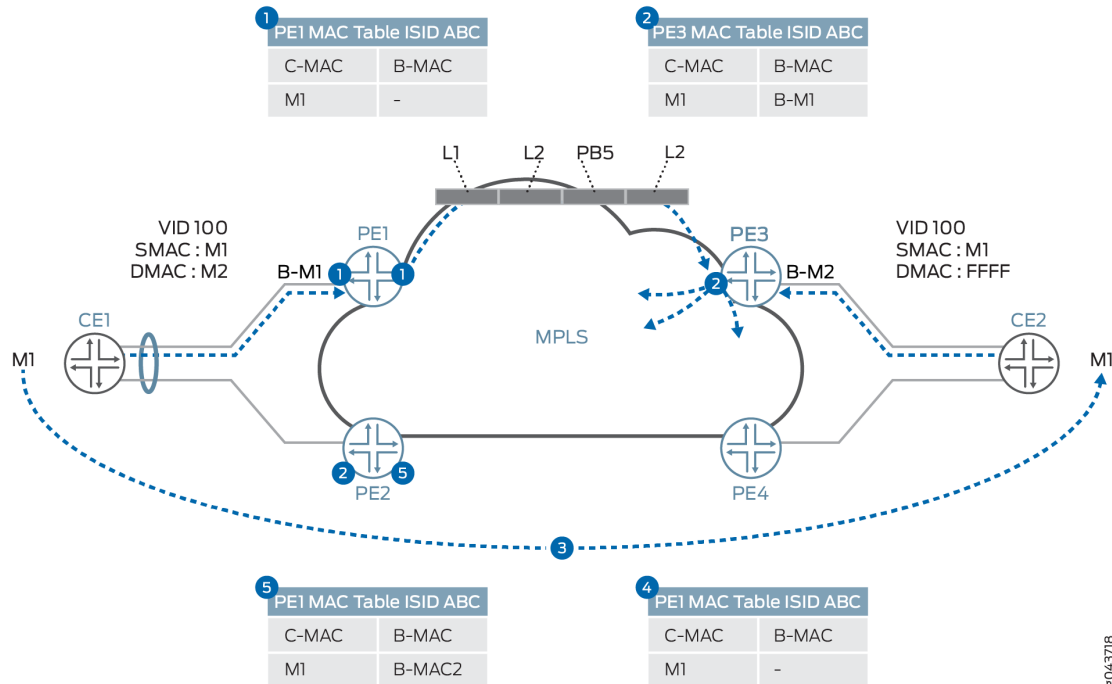
The packet flow for packets received from the core side is as follows:

1. If the frame is received from the core-facing interface, the interface belongs to the MPLS family, and the MPLS label lookup is done in the mpls.0 routing table next hop. The result of this lookup is the routing instance context.
2. The next lookup is done based on the I-SID from the packet to the BBD lookup.
3. If the BBD is found, then I-SID based filtering rules are applied, where the I-SID configured MAC should match the packet source B-MAC, then the frame is dropped.
4. The I-SID MAC table is updated for the destination B-MAC associated with the destination C-MAC for building the C-MAC-to-B-MAC association.
5. The I-SID header is removed and C-BD is found based on the PIP interface.
6. The next lookup is done in the C-BD because the PIP interface belongs to the bridge family.
7. The C-BD lookup points to either a unicast MAC route or a flood MAC route, and this route points to a CE interface or flood route.

Handling MAC Mobility in PBB-EVPN

Figure 152 on page 1435 illustrates PBB-EVPN MAC mobility from the point of the forwarding and control plane.

Figure 152: PBB-EVPN MAC Mobility Handling



MAC mobility from the point of the forwarding and control plane is handled as follows:

1. Device PE1 learns the C-MAC address, M1, on the local port and forwards it across the core according to the destination-C-MAC-to-remote-B-MAC mapping. This mapping is either statically configured or learned through the data plane. If the destination-C-MAC-to-remote-B-MAC mapping is not found in the I-SID mapping table, then the remote B-MAC is derived by using the I-SID.
2. Device PE3 learns the C-MAC address, M1, through the B-MAC address, B-M1, from the dataplane.
3. Customer M1 is moved from Device CE1 to behind Device CE2.
4. When Customer M1 wants to communicate with a customer behind Device CE1, a broadcast traffic with VID: 100, Source MAC: M1, and destination MAC: ff.ff.ff.ff.ff is sent. Device PE3 learns the MAC M1 in the C-BD MAC table and updates the M1 location in the I-SID mapping table.
5. Device PE1 receives the packet and M1 is learned and updated in the I-SID mapping table as reachable through the remote MAC is B-M2.

Handling End-to-End OAM for PBB-EVPN

You can run provider-level Operation, Administration, and Maintenance (OAM) by running connectivity fault management (CFM) on the inward or outward facing maintenance endpoints (MEPs) either on the PIP interface or over the EVPN service.

Currently, the designated forwarder (DF) election is decided based on the DF election algorithm and it is the local decision of the PE devices. This is useful in an end-to-end service handling scenario, where the decision of DF election can be done with operator's consent as well and vice versa. Another scenario in which it might be useful to influence the DF role per service basis or propagating the DF to a CE device is for multihomed networks, where there is no direct link between the CE and PE devices.

Handling QoS and Firewall Policer Support for PBB-EVPN

[Table 49 on page 1436](#) provides details about the QoS and firewall features that are supported in the context of PBB-EVPN integration.

Table 49: Firewall and QoS Feature Support in PBB-EVPN

Feature	Description	Support on round-trip time (RTT)	Support on CE Interface	Support on Core Interface
Classification	Fixed classification to one FC	Yes	Yes	Yes
	Behavior aggregate (BA) and multifield classifier (MF) classification for inner output vlan .1p bit	Yes	Yes	Yes
	BA and MF classification based on DEI and PCP	No	Not required	No
	BA and MF classification based on Exp	No	No	Yes
CoS Marking	.1P to I-SID PCP and DEI: Customer VLAN .1p	No	By default, .1P is mapped to PCP and DEI.	Yes

Table 49: Firewall and QoS Feature Support in PBB-EVPN (Continued)

Feature	Description	Support on round-trip time (RTT)	Support on CE Interface	Support on Core Interface
	.1P to Exp: Customer VLAN .1p	No	No	Yes
	MPLS EXP to I-SID PCP and DEI	No	Default behavior	No
	EXP to .1P	No	Yes	No
QoS shaping	Hierarchical scheduling and shaping on ingress device	No	Yes	Yes
	Hierarchical scheduling and shaping on egress device	No	Yes	Yes
Firewall filter	Filtering BUM traffic	Unknown traffic only	Broadcast and multicast traffic only	Broadcast and multicast traffic only
	I-SID-based firewall filter	No	No	No
	Customer VLAN-based filter	No	Yes	Yes
Policer (2 rate 3 color)	Ingress direction	No	Yes	Yes
	Egress direction	No	Yes	Yes

Implementation Overview of PBB-EVPN Integration

The following sections provide use case scenarios for PBB-EVPN integration for DCI.

PBB-EVPN Failure Scenarios

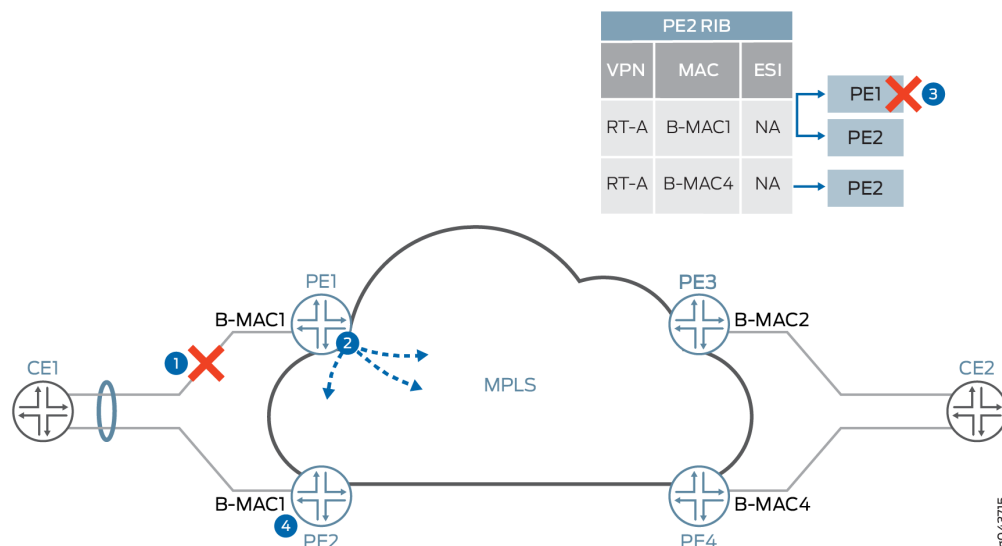
There are different PBB-EVPN failure scenarios that should be taken care of while providing an end-to-end solution. These failure scenarios can be of the following types:

Segment Failure

A segment, or CE-facing link, failure is handled in the active/active and active/standby multihoming redundancy modes.

Figure 153 on page 1438 shows the handling of segment failure for flow-based load balancing at Device CE1.

Figure 153: PBB-EVPN Segment Failure



A segment failure in PBB-EVPN is handled as follows:

1. Ethernet link between Devices CE1 and PE1 failed due to fiber cut or the interface is down. Device PE1 detects the failed segment.
2. Device PE1 withdraws the B-MAC address that is advertised for the failed segment (B-M1).
3. The CE1-facing link goes down. If the link failure happens in the single-active redundancy mode or no redundancy case, the C-MAC flush is also done.

The C-MAC address flushing happens in two ways:

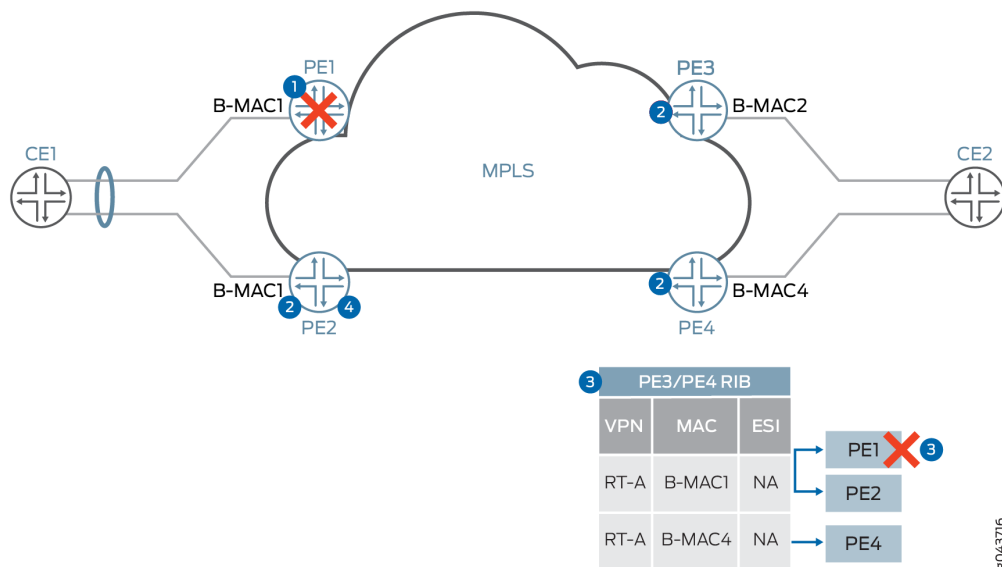
- If Device PE2 uses the shared B-MAC address for multiple I-SIDs, it notifies the remote PE device by readvertising the B-MAC address with the MAC mobility extended community attribute by incrementing the value of counter. This causes the remote PE device to flush all C-MAC addresses associated with the B-MAC address for Device PE1.
 - If Device PE2 uses the dedicated B-MAC address, then it withdraws the B-MAC address associated with the failed segment and sends it to Devices PE2, PE3, and PE4.
4. After receiving the B-MAC withdraw from Device PE1, Device PE3 removes PE1 reachability for B-MAC1 from its forwarding table. Reachability of B-MAC1 through Device PE2 still exists.
 5. The DF election is rerun on Device PE2 for all the I-SIDs for the Ethernet segment ESI.

Node Failure

A node, or PE device, failure scenario is similar to a segment failure from the point of view of CE side failure handling, but it is different from core side failure handling. In the case of core side failure handling, EVPN depends on the BGP session timeout for clearing the state of the EVPN sessions on affected PE devices.

Figure 154 on page 1439 illustrates a node failure scenario for node failure handling.

Figure 154: PBB-EVPN Node Failure



1. Device PE1 failed and the CE side switchover to Device PE2 is done by an interface down event.
2. Devices PE2, PE3, and PE4, or BGP route reflector, detect the BGP session timeout with Device PE1.

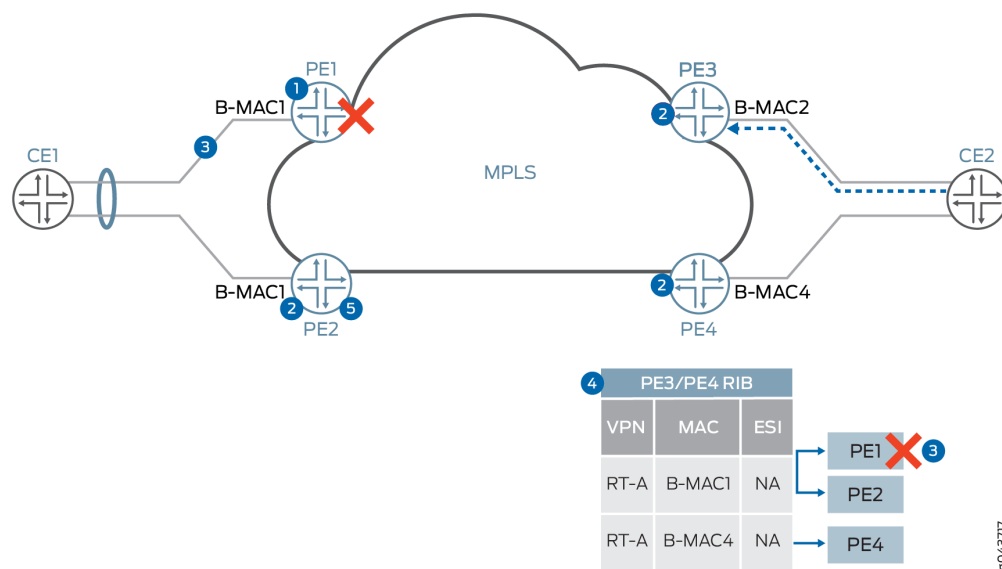
3. As soon as the BGP session timeout happens, Devices PE3 and PE4 remove Device PE1 from the forwarding table by marking the Device PE1 next hop as unreachable or deleted. In the case of single-active redundancy mode, the I-SID table for C-MAC-to-B-MAC mapping table has to be flushed or updated. In the case of active/active redundancy mode, it is not required to flush the I-SID table, because the same B-MAC address is used for both Devices PE1 and PE2 for a given EVI.
4. At Device PE2, after a BGP timeout, the DF election algorithm is rerun and Device PE2 becomes the DF for all I-SIDs on an affected Ethernet segment.

Core Failure

The handling of core side isolation in the EVPN network is similar to the PE side failure, with some differences in handling of the CE device or Ethernet segment.

Figure 155 on page 1440 provides details about the handling of core isolation.

Figure 155: PBB-EVPN Core Failure



Core isolation in PBB-EVPN is handled as follows:

1. Device PE1 loses connectivity to the core.
2. Devices PE2, PE3, and PE4, or BGP route reflector, detect the BGP session timeout with Device PE1.
3. Device PE1 sends an LACP OUT_OF_SYNC message to Device CE1 to take the port out of the bundle.
4. Device PE2, or BGP route reflector, detects the BGP session timeout with Device PE1.

5. Device PE2 reruns the DF election and is selected as the DF for all the I-SIDs on the segment.

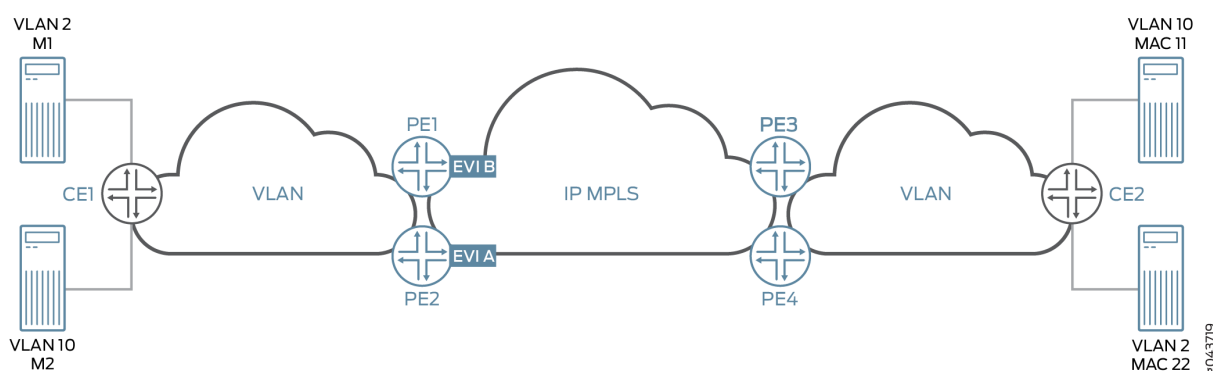
PBB-EVPN I-SID Use Case Scenarios

I-SID Base Service

In the case of the I-SID base service, there is one-to-one mapping between a bridge domain and an EVI. In this case, there is no need to carry the I-SID in the MAC advertisement route, because the bridge domain ID can be derived from the route target (RT) associated with this route. The MPLS label allocation is done on a per-EVI basis.

Figure 156 on page 1441 provides an overview of the I-SID base use case scenario.

Figure 156: I-SID Base Service Use Case



In the case of I-SID load balancing, where load balancing of traffic is done on a per-service basis from the originating CE link aggregation group (LAG) configuration, there are two models for B-MAC addresses:

- Shared source B-MAC

In this model, all I-SIDs from an Ethernet segment share one source B-MAC address. This model has limitations from the point of view of B-MAC withdrawal because of service failure. The remote PE device needs to flush B-MAC-to-C-MAC mapping for all I-SIDs. This creates problem for convergence because MAC flush is done for all I-SIDs.

- Unique source B-MAC per I-SID

Unique unicast B-MAC addresses (one per I-SID) are allocated per multihomed Ethernet segment. The DF filtering is applied to unicast and multicast traffic, in both the core-to-segment and segment-to-core directions.

I-SID-Aware Service

In the case of I-SID-aware service, multiple I-SIDs can be mapped to the same EVI. But there is one-to-one mapping between a bridge domain and an I-SID. The Ethernet Tag ID must be set to the I-SID in the BGP routes advertisements. The MPLS label allocation is done on a per-EVI or per-EVI/I-SID basis so that the PBB can be terminated at the ingress PE device and recreated at the egress PE device.

VPLS Integration with PBB-EVPN Use Case Scenario

In this use case scenario, VPLS is one cloud that is getting integrated with PBB-EVPN by using logical tunnel interfaces. The logical tunnel interface is being terminated into the customer bridge domain (C-BD). MAC address learning from the VPLS cloud is happening in the context of the C-BD. The C-bridge domain gets mapped to the backbone bridge domain and going to the EVPN cloud.

NOTE: PBB-VPLS interworking with PBB-EVPN under an EVI is not supported in Junos OS Release 17.2R1.

PBB-EVPN Redundancy Use Case Scenarios

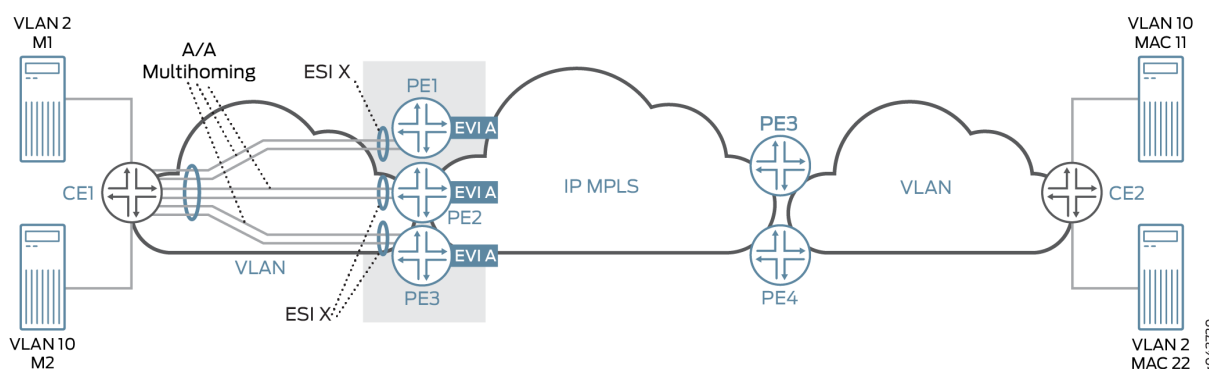
Single Active Redundancy Use Case Scenario

In this use case scenario, a CE device is multihomed to multiple PE devices. The Ethernet segment for this scenario is defined by configuring the same ESI ID on multiple physical interfaces or aggregated Ethernet interfaces on the PE devices along with the mode of operation. In this mode of operation, only one PE device (that is, the DF) is allowed to forward the traffic to and from this Ethernet segment for BUM traffic. The DF election is done based on each ESI in an EVI and by taking the lowest configured I-SID into consideration. This also depends on this number of PE devices to which a segment is multihomed. The service carving is achieved by putting different I-SIDs in different EVIs. The DF election is similar to the DF election in the VLAN-based EVPN. A default timer of 3 seconds is used for reception of Ethernet segment routes from other PE nodes, and this timer can be configured with the same command as used in EVPN—the `designated-forwarder-election hold-time` statement.

In case of PBB-EVPN, the Ethernet autodiscovery route is not used. This mode is specified in the B-MAC advertisement route. In a single-homed or multihomed setup, with the single-active mode, the ESI field must be set to 0 in the B-MAC route advertisement. If ESI 0 is used in the MAC advertisement route, then I-SID-based load balancing is performed. An I-SID value can serve as a single home, an active/standby scenario, or an active/active scenario. It cannot, however, be used in a mixed mode of operation.

Figure 157 on page 1443 provides the use case scenario for active/standby multihoming along with DF election.

Figure 157: PBB-EVPN Redundancy Use Case



Active/Standby Redundancy Use Case Scenario

In the case of the active/active redundancy use case scenario, the DF election is used for handling the BUM traffic. In the case of PBB-EVPN, split horizon of EVPN is not used for filtering the BUM traffic. Instead, BUM traffic is filtered by filtering the destination B-MAC, where the configured B-MAC is the same as the received packet B-MAC; therefore that packet is from the same segment.

The aliasing approach is the same as the EVPN, but the B-MAC route is advertised by setting the ESI field to MAX-ESI. When the remote PE device receives the B-MAC route with the MAX-ESI value, then the remote PE device does the load balancing between Devices PE1 and PE2.

Active/Active Redundancy Use Case Scenario

In a PBB-EVPN active-active multihomed network, all the multihomed PE devices require identical MAC installations. For this purpose, BGP is used to sync the source C-MAC addresses (on the CE side) or the remote C-MAC addresses (from the core) across the multihomed PE devices for same ESI.

To enable MAC sync:

1. For the source C-MAC address sync:
 - Configure per-packet-load-balancing on the CE device.
 - Ensure that there are minimum flows per source C-MAC, so that each source C-MAC takes both links toward the multihomed PE devices at least once. This ensures that both the multihomed PE devices learn each source C-MAC.

2. For remote C-MAC address sync:

- Ensure that there are minimum flows per remote C-MAC, so that each remote C-MAC takes both links(aliasing) towards the multihomed PE devices at least once while traversing the core. This ensures that both the multihomed PE devices learn each remote C-MAC.

Configuration Overview of PBB-EVPN Integration

The PBB-EVPN configuration is done using the following models:

- One-to-one mapping between the I-SID and the bridge domain

In this configuration model, there is a shared EVPN instance (EVI) between different services, although there is one-to-one mapping between the bridge domain and the I-SID.

- Many-to-one mapping between the I-SID and the bridge domain

In this configuration model, virtual switch configuration is used to allow multiple I-SIDs to be mapped to one bridge domain. This model allows only one bridge domain in a given EVI, and all other bridge domains are mapped to the other Layer 2 services.

Sample PBB-EVPN Port Configuration:

- Provider Backbone Port (PBP) Configuration:

```
routing-instances {
  evpnA {
    instance-type virtual-switch;
    route-distinguisher 192.0.2.1;
    vrf-target target:65221:111;
    protocols {
      evpn {
        label-allocation per-instance;
        extended-isid-list [10000 10401 20000-20001]
      }
    }
  }
}
```

- Customer Backbone Port (CBP) Configuration:

```
interfaces {
  cbp {
```

```

flexible-valn-tagging;
unit 0 {
    family bridge {
        interface-mode trunk;
        isid-list [10000 10401 20000-20001] [all];
    }
}
}

```

- Provider Instance Port (PIP) Configuration:

```

interfaces {
    pip {
        flexible-valn-tagging;
        unit 0 {
            family bridge {
                interface-mode trunk;
                isid-list [20000 20001];
            }
        }
        unit 1 {
            family bridge {
                interface-mode trunk;
                isid-list [10000 10401];
            }
        }
    }
}

```

- Customer Instance Port (CIP) Configuration:

```

interfaces {
    ge-4/0/0 {
        flexible-valn-tagging;
    }
    esi 1 primary;
    unit 0 {
        family bridge {
            interface-mode trunk;
            vlan-id-list [1-399 500 800];
        }
    }
}

```



```

    }
    ge-7/0/0/ {
        flexible-vlan-tagging;
        unit 0 {
            family bridge {
                interface-mode trunk;
                vlan-id-list [1-401];
            }
        }
    }
    ge-8/0/0/ {
        flexible-vlan-tagging;
        unit 0 {
            family bridge {
                interface-mode trunk;
                vlan-id-list [500-501];
            }
        }
    }
}

```

Sample PBB-EVPN Routing Instance Configuration:

- Provider Routing Instance Configuration:

```

routing-instances {
    EVPN A {
        instance-type virtual-switch;
        route-distinguisher 192.0.2.1;
        vrf-target target:65221:111;
        protocols {
            pbb-evpn {
                label-allocation per-instance;
                extended-isid-list [10000 10401 20000] [all];
            }
            evpn {
                label-allocation per-instance;
                extended-vlan-list 1000;
            }
        }
        interface cbp.0;
        bridge-domains {

```

```

        B-BD1 {
            isid-list 10000;
        }
        B-BD2 {
            isid-list 10401;
        }
        B-BD3 {
            isid-list 20000;
        }
        B-BD4 {
            isid-list 1000;
        }
    }
    pbb-options {
        vlan-id 1000 isid-list [20001];
        default-bvlan 22;
    }
}

```

- Customer Routing Instance Configuration:

```

routing-instances {
    PBN-1 {
        instance-type virtual-switch;
        interface ge-4/0/0.0;
        interface pip.0;
        bridge-domains {
            customer-BDs {
                vlan-id-list [1-399 500 800];
            }
        }
        pbb-options {
            peer-instance EVPN A;
            source-bmac <mac-address>;
            service-groups {
                pbb-1-isid {
                    source-bmac <mac-address>;
                    isid {
                        isid-1 {
                            isid 20000 vlan-id-list [1-399 500] [all];
                            source-bmac <mac-address>;
                        }
                    }
                }
            }
        }
    }
}

```

```

        map-dest-bmac-to-dest-cmac <b-mac> <c-mac>;
    }
    isid-2 {
        isid 20001 vlan-id-list [800] [all];
    }
    }
    }
    }
}
PBN-2 {
    instance-type virtual-switch;
    interface ge-7/0/0.0;
    interface pip.1;
    bridge-domains {
        customer-BDs {
            vlan-id-list [1-401];
        }
    }
    pbb-options {
        peer-instance EVPN A;
        default-isid <i-sid>;
        service-groups {
            pbb-2-isid-1 {
                isid {
                    isid-1 {
                        isid 10000 vlan-id-list [1-400];
                    }
                }
            }
            pbb-2-isid-2 {
                isid {
                    isid-1 {
                        isid 10401 vlan-id-list [401];
                    }
                }
            }
        }
    }
}
}

```

Supported and Unsupported Features on PBB-EVPN

Junos OS supports the following features with PBB-EVPN:

- Graceful Routing Engine switchover (GRES), unified in-service software upgrade (ISSU), and nonstop software upgrade (NSSU).
- Nonstop active routing (NSR) for BGP peers configured with EVPN family.

NSR on PBB-EVPN replicates and recreates backbone MAC (B-MAC) routes, inclusive multicast routes, and Ethernet segment identifier (ESI) routes.

- Feature support on 64-bit platforms.
- IEEE assigned standard ether type as 0x88E7 for I-SID frames. In addition to this, 802.1x can be used.

The following security considerations are supported:

- Packet destined to the Layer 2 ether type as 0x88E7 is processed only if PBB is enabled on the ingress core PE device.
- Packet received from the core is processed only if the I-SID is known and configured on the ingress PE device; otherwise, the frame is dropped.

Junos OS does not support the following features for PBB-EVPN integration:

- Complete support of EVPN NSR
- Integrated routing and bridging (IRB) interfaces
- IPv6 IP addresses
- Logical systems

RELATED DOCUMENTATION

[Example: Configuring PBB with Multihomed EVPN | 1478](#)

[Example: Configuring PBB with Single-Homed EVPN | 1450](#)

[pbb-evpn-core | 1640](#)

Example: Configuring PBB with Single-Homed EVPN

IN THIS SECTION

- [Requirements | 1450](#)
- [Overview and Topology | 1450](#)
- [Configuration | 1451](#)
- [Verification | 1470](#)

This example shows how to integrate provider backbone bridging (PBB) with Ethernet VPN (EVPN). With this integration, the control plane operations in the core are simplified, providing faster convergence and scalability enhancements than regular EVPN. The PBB-EVPN applications include Data Center Interconnect (DCI) and carrier Ethernet E-LAN services.

Requirements

This example uses the following hardware and software components:

- Three provider edge (PE) devices each connected to single-homed customer sites.
- Four customer edge (CE) devices that are single-homed to the PE devices.
- Junos OS Release 17.2R1 or later running on all the PE routers.

Before you begin:

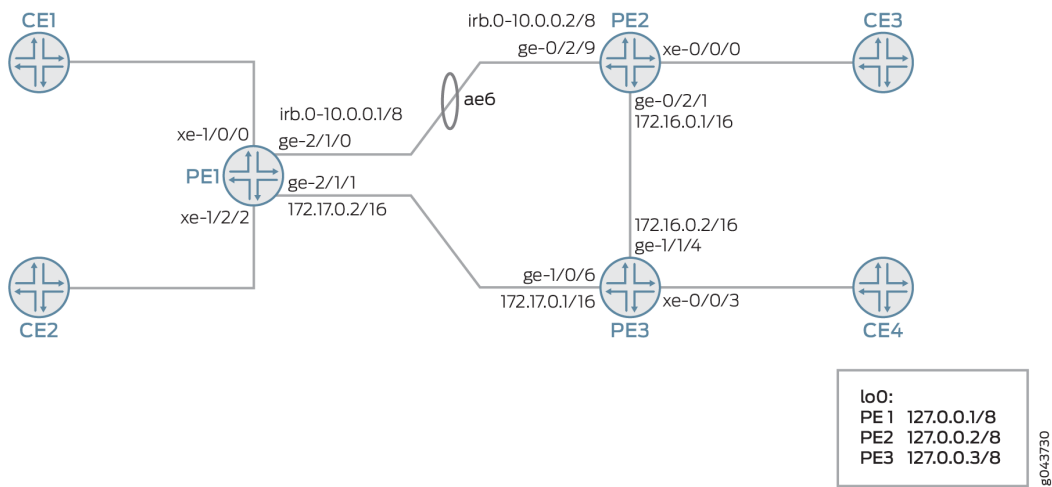
- Configure the device interfaces.
- Configure an IGP, such as OSPF, on all the PE devices.
- Establish an internal BGP session between the PE devices.
- Enable RSVP on the PE devices.
- Configure MPLS and label-switched paths (LSPs) between the PE devices.

Overview and Topology

Starting in Junos OS Release 17.2R1, PBB is integrated with Ethernet VPN (EVPN) to enable significant reduction in the control plane learning across the core, allowing a huge number of Layer 2 services, such as data center connectivity, to transit the network in a simplified manner.

In a PBB-EVPN network, the backbone core bridge (BCB) device in the PBB core is replaced with MPLS, while retaining the service scaling properties of the PBB backbone edge bridge (BEB). The B-component (provider routing instance) is signalled using EVPN BGP signaling and encapsulated inside MPLS using provider edge (PE) and provider (P) devices. Thus, PBB-EVPN combines the vast scaling property of PBB with the simplicity of a traditional basic MPLS core network, resulting in significant reduction in the amount of network-wide state information, as opposed to regular PBB.

Figure 158: PBB with Single-Homed EVPN



In [Figure 158 on page 1451](#), PBB is integrated with EVPN, where the CE devices are single-homed to Devices PE1, PE2, and PE3.

Configuration

IN THIS SECTION

- [CLI Quick Configuration | 1451](#)
- [Procedure | 1458](#)
- [Results | 1464](#)

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the [edit] hierarchy level, and then enter `commit` from configuration mode.

PE1

```

set chassis aggregated-devices ethernet device-count 16
set chassis network-services enhanced-ip
set interfaces xe-1/0/0 flexible-vlan-tagging
set interfaces xe-1/0/0 encapsulation flexible-ethernet-services
set interfaces xe-1/0/0 unit 0 encapsulation vlan-bridge
set interfaces xe-1/0/0 unit 0 vlan-id 10
set interfaces xe-1/0/0 unit 1 encapsulation vlan-bridge
set interfaces xe-1/0/0 unit 1 vlan-id 20
set interfaces xe-1/2/2 flexible-vlan-tagging
set interfaces xe-1/2/2 encapsulation flexible-ethernet-services
set interfaces xe-1/2/2 unit 0 encapsulation vlan-bridge
set interfaces xe-1/2/2 unit 0 vlan-id 10
set interfaces xe-1/2/2 unit 0 family bridge filter input BRI
set interfaces xe-1/2/2 unit 1 encapsulation vlan-bridge
set interfaces xe-1/2/2 unit 1 vlan-id 20
set interfaces xe-1/2/2 unit 2 encapsulation vlan-bridge
set interfaces xe-1/2/2 unit 2 vlan-id 11
set interfaces xe-1/2/2 unit 2 family bridge
set interfaces xe-1/2/2 unit 3 encapsulation vlan-bridge
set interfaces xe-1/2/2 unit 3 vlan-id 21
set interfaces xe-1/2/2 unit 3 family bridge
set interfaces ge-2/1/0 gigether-options 802.3ad ae6
set interfaces ge-2/1/1 unit 0 family inet address 10.0.0.1/8
set interfaces ge-2/1/1 unit 0 family iso
set interfaces ge-2/1/1 unit 0 family mpls
set interfaces ae6 encapsulation ethernet-bridge
set interfaces ae6 unit 0 family bridge
set interfaces cbp0 unit 0 family bridge interface-mode trunk
set interfaces cbp0 unit 0 family bridge bridge-domain-type bvlan
set interfaces cbp0 unit 0 family bridge isid-list all
set interfaces cbp0 unit 1 family bridge interface-mode trunk
set interfaces cbp0 unit 1 family bridge bridge-domain-type bvlan
set interfaces cbp0 unit 1 family bridge isid-list all
set interfaces irb unit 0 family inet address 10.0.0.1/8
set interfaces irb unit 0 family iso
set interfaces irb unit 0 family mpls
set interfaces lo0 unit 0 family inet address 127.0.0.1/8 primary
set interfaces pip0 unit 0 family bridge interface-mode trunk
set interfaces pip0 unit 0 family bridge bridge-domain-type svlan
set interfaces pip0 unit 0 family bridge isid-list all-service-groups

```

```

set interfaces pip0 unit 1 family bridge interface-mode trunk
set interfaces pip0 unit 1 family bridge bridge-domain-type svlan
set interfaces pip0 unit 1 family bridge isid-list all-service-groups
set routing-options router-id 127.0.0.1
set routing-options autonomous-system 65221
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path PE1toPE2 from 127.0.0.1
set protocols mpls label-switched-path PE1toPE2 to 127.0.0.2
set protocols mpls label-switched-path PE1toPE3 from 127.0.0.1
set protocols mpls label-switched-path PE1toPE3 to 127.0.0.3
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 127.0.0.1
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 127.0.0.2
set protocols bgp group ibgp neighbor 127.0.0.3
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set routing-instances pbbn1 instance-type virtual-switch
set routing-instances pbbn1 interface cbp0.0
set routing-instances pbbn1 route-distinguisher 127.0.0.1:100
set routing-instances pbbn1 vrf-target target:100:100
set routing-instances pbbn1 protocols evpn pbb-evpn-core
set routing-instances pbbn1 protocols evpn extended-isid-list 1000
set routing-instances pbbn1 protocols evpn extended-isid-list 2000
set routing-instances pbbn1 bridge-domains bda vlan-id 100
set routing-instances pbbn1 bridge-domains bda isid-list 1000
set routing-instances pbbn1 bridge-domains bda vlan-id-scope-local
set routing-instances pbbn1 bridge-domains bdb vlan-id 200
set routing-instances pbbn1 bridge-domains bdb isid-list 2000
set routing-instances pbbn1 bridge-domains bdb vlan-id-scope-local
set routing-instances pbn1 instance-type virtual-switch
set routing-instances pbn1 interface pip0.0
set routing-instances pbn1 bridge-domains bda domain-type bridge
set routing-instances pbn1 bridge-domains bda vlan-id 10
set routing-instances pbn1 bridge-domains bda interface xe-1/2/2.0
set routing-instances pbn1 bridge-domains bda interface xe-1/0/0.0
set routing-instances pbn1 bridge-domains bdb domain-type bridge
set routing-instances pbn1 bridge-domains bdb vlan-id 20
set routing-instances pbn1 bridge-domains bdb interface xe-1/2/2.1

```



```

set routing-instances pbn1 bridge-domains bdb interface xe-1/0/0.1
set routing-instances pbn1 bridge-domains bdc domain-type bridge
set routing-instances pbn1 bridge-domains bdc vlan-id 11
set routing-instances pbn1 bridge-domains bdc interface xe-1/2/2.2
set routing-instances pbn1 bridge-domains bdd domain-type bridge
set routing-instances pbn1 bridge-domains bdd vlan-id 21
set routing-instances pbn1 bridge-domains bdd interface xe-1/2/2.3
set routing-instances pbn1 pbb-options peer-instance pbbn1
set routing-instances pbn1 service-groups sga service-type elan
set routing-instances pbn1 service-groups sga pbb-service-options isid 1000 vlan-id-list 10
set routing-instances pbn1 service-groups sga pbb-service-options isid 1000 vlan-id-list 11
set routing-instances pbn1 service-groups sga pbb-service-options source-bmac 00:50:50:50:50:50
set routing-instances pbn1 service-groups sgb service-type elan
set routing-instances pbn1 service-groups sgb pbb-service-options isid 2000 vlan-id-list 20
set routing-instances pbn1 service-groups sgb pbb-service-options isid 2000 vlan-id-list 21
set bridge-domains bd vlan-id none
set bridge-domains bd interface ae6.0
set bridge-domains bd routing-interface irb.0

```

PE2

```

set chassis aggregated-devices ethernet device-count 3
set chassis network-services enhanced-ip
set interfaces xe-0/0/0 flexible-vlan-tagging
set interfaces xe-0/0/0 encapsulation flexible-ethernet-services
set interfaces xe-0/0/0 unit 0 encapsulation vlan-bridge
set interfaces xe-0/0/0 unit 0 vlan-id 10
set interfaces xe-0/0/0 unit 1 encapsulation vlan-bridge
set interfaces xe-0/0/0 unit 1 vlan-id 20
set interfaces xe-0/0/0 unit 2 encapsulation vlan-bridge
set interfaces xe-0/0/0 unit 2 vlan-id 11
set interfaces xe-0/0/0 unit 2 family bridge
set interfaces xe-0/0/0 unit 3 encapsulation vlan-bridge
set interfaces xe-0/0/0 unit 3 vlan-id 21
set interfaces xe-0/0/0 unit 3 family bridge
set interfaces ge-0/2/1 unit 0 family inet address 172.16.0.1/16
set interfaces ge-0/2/1 unit 0 family mpls
set interfaces ge-0/2/9 gigether-options 802.3ad ae6
set interfaces ae6 encapsulation ethernet-bridge
set interfaces ae6 unit 0 family bridge
set interfaces cbp0 unit 0 family bridge interface-mode trunk
set interfaces cbp0 unit 0 family bridge bridge-domain-type bvlan

```

```

set interfaces cbp0 unit 0 family bridge isid-list all
set interfaces cbp0 unit 1 family bridge interface-mode trunk
set interfaces cbp0 unit 1 family bridge bridge-domain-type bvlan
set interfaces cbp0 unit 1 family bridge isid-list all
set interfaces irb unit 0 family inet address 10.0.0.2/8
set interfaces irb unit 0 family mpls
set interfaces lo0 unit 0 family inet address 127.0.0.2/8 primary
set interfaces pip0 unit 0 family bridge interface-mode trunk
set interfaces pip0 unit 0 family bridge bridge-domain-type svlan
set interfaces pip0 unit 0 family bridge isid-list all-service-groups
set interfaces pip0 unit 1 family bridge interface-mode trunk
set interfaces pip0 unit 1 family bridge bridge-domain-type svlan
set interfaces pip0 unit 1 family bridge isid-list all-service-groups
set routing-options router-id 127.0.0.2
set routing-options autonomous-system 65221
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path PE2toPE1 from 127.0.0.2
set protocols mpls label-switched-path PE2toPE1 to 127.0.0.1
set protocols mpls label-switched-path PE2toPE3 from 127.0.0.2
set protocols mpls label-switched-path PE2toPE3 to 127.0.0.3
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 127.0.0.2
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 127.0.0.1
set protocols bgp group ibgp neighbor 127.0.0.3
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set routing-instances pbbn1 instance-type virtual-switch
set routing-instances pbbn1 interface cbp0.0
set routing-instances pbbn1 route-distinguisher 127.0.0.2:100
set routing-instances pbbn1 vrf-target target:100:100
set routing-instances pbbn1 protocols evpn pbb-evpn-core
set routing-instances pbbn1 protocols evpn extended-isid-list 1000
set routing-instances pbbn1 protocols evpn extended-isid-list 2000
set routing-instances pbbn1 bridge-domains bda vlan-id 100
set routing-instances pbbn1 bridge-domains bda isid-list 1000
set routing-instances pbbn1 bridge-domains bda vlan-id-scope-local
set routing-instances pbbn1 bridge-domains bdb vlan-id 200
set routing-instances pbbn1 bridge-domains bdb isid-list 2000

```

```

set routing-instances pbbn1 bridge-domains bdb vlan-id-scope-local
set routing-instances pbn1 instance-type virtual-switch
set routing-instances pbn1 interface pip0.0
set routing-instances pbn1 bridge-domains bda domain-type bridge
set routing-instances pbn1 bridge-domains bda vlan-id 10
set routing-instances pbn1 bridge-domains bda interface xe-0/0/0.0
set routing-instances pbn1 bridge-domains bdb domain-type bridge
set routing-instances pbn1 bridge-domains bdb vlan-id 20
set routing-instances pbn1 bridge-domains bdb interface xe-0/0/0.1
set routing-instances pbn1 bridge-domains bdc domain-type bridge
set routing-instances pbn1 bridge-domains bdc vlan-id 11
set routing-instances pbn1 bridge-domains bdc interface xe-0/0/0.2
set routing-instances pbn1 bridge-domains bdd domain-type bridge
set routing-instances pbn1 bridge-domains bdd vlan-id 21
set routing-instances pbn1 bridge-domains bdd interface xe-0/0/0.3
set routing-instances pbn1 pbb-options peer-instance pbbn1
set routing-instances pbn1 service-groups sga service-type elan
set routing-instances pbn1 service-groups sga pbb-service-options isid 1000 vlan-id-list 10
set routing-instances pbn1 service-groups sga pbb-service-options isid 1000 vlan-id-list 11
set routing-instances pbn1 service-groups sga pbb-service-options source-bmac 00:51:51:51:51:51
set routing-instances pbn1 service-groups sgb service-type elan
set routing-instances pbn1 service-groups sgb pbb-service-options isid 2000 vlan-id-list 20
set routing-instances pbn1 service-groups sgb pbb-service-options isid 2000 vlan-id-list 21
set bridge-domains bd vlan-id none
set bridge-domains bd interface ae6.0
set bridge-domains bd routing-interface irb.0

```

PE3

```

set chassis aggregated-devices ethernet device-count 16
set chassis network-services enhanced-ip
set interfaces xe-0/0/3 flexible-vlan-tagging
set interfaces xe-0/0/3 encapsulation flexible-ethernet-services
set interfaces xe-0/0/3 unit 0 encapsulation vlan-bridge
set interfaces xe-0/0/3 unit 0 vlan-id 10
set interfaces xe-0/0/3 unit 1 encapsulation vlan-bridge
set interfaces xe-0/0/3 unit 1 vlan-id 20
set interfaces xe-0/0/3 unit 2 encapsulation vlan-bridge
set interfaces xe-0/0/3 unit 2 vlan-id 11
set interfaces xe-0/0/3 unit 2 family bridge
set interfaces xe-0/0/3 unit 3 encapsulation vlan-bridge
set interfaces xe-0/0/3 unit 3 vlan-id 21

```

```

set interfaces xe-0/0/3 unit 3 family bridge
set interfaces ge-1/0/6 unit 0 family inet address 172.17.0.1/16
set interfaces ge-1/0/6 unit 0 family mpls
set interfaces ge-1/1/4 unit 0 family inet address 172.16.0.2/16
set interfaces ge-1/1/4 unit 0 family mpls
set interfaces cbp0 unit 0 family bridge interface-mode trunk
set interfaces cbp0 unit 0 family bridge bridge-domain-type bvlan
set interfaces cbp0 unit 0 family bridge isid-list all
set interfaces cbp0 unit 1 family bridge interface-mode trunk
set interfaces cbp0 unit 1 family bridge bridge-domain-type bvlan
set interfaces cbp0 unit 1 family bridge isid-list all
set interfaces lo0 unit 0 family inet address 127.0.0.3/8 primary
set interfaces pip0 unit 0 family bridge interface-mode trunk
set interfaces pip0 unit 0 family bridge bridge-domain-type svlan
set interfaces pip0 unit 0 family bridge isid-list all-service-groups
set interfaces pip0 unit 1 family bridge interface-mode trunk
set interfaces pip0 unit 1 family bridge bridge-domain-type svlan
set interfaces pip0 unit 1 family bridge isid-list all-service-groups
set routing-options router-id 127.0.0.3
set routing-options autonomous-system 65221
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path PE3toPE1 from 127.0.0.3
set protocols mpls label-switched-path PE3toPE1 to 127.0.0.1
set protocols mpls label-switched-path PE3toPE2 from 127.0.0.3
set protocols mpls label-switched-path PE3toPE2 to 127.0.0.2
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 127.0.0.3
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 127.0.0.1
set protocols bgp group ibgp neighbor 127.0.0.2
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set routing-instances pbbn1 instance-type virtual-switch
set routing-instances pbbn1 interface cbp0.0
set routing-instances pbbn1 route-distinguisher 127.0.0.3:100
set routing-instances pbbn1 vrf-target target:100:100
set routing-instances pbbn1 protocols evpn pbb-evpn-core
set routing-instances pbbn1 protocols evpn extended-isid-list 1000
set routing-instances pbbn1 protocols evpn extended-isid-list 2000

```

```

set routing-instances pbbn1 bridge-domains bda vlan-id 100
set routing-instances pbbn1 bridge-domains bda isid-list 1000
set routing-instances pbbn1 bridge-domains bda vlan-id-scope-local
set routing-instances pbbn1 bridge-domains bdb vlan-id 200
set routing-instances pbbn1 bridge-domains bdb isid-list 2000
set routing-instances pbbn1 bridge-domains bdb vlan-id-scope-local
set routing-instances pbn1 instance-type virtual-switch
set routing-instances pbn1 interface pip0.0
set routing-instances pbn1 bridge-domains bda domain-type bridge
set routing-instances pbn1 bridge-domains bda vlan-id 10
set routing-instances pbn1 bridge-domains bda interface xe-0/0/3.0
set routing-instances pbn1 bridge-domains bdb domain-type bridge
set routing-instances pbn1 bridge-domains bdb vlan-id 20
set routing-instances pbn1 bridge-domains bdb interface xe-0/0/3.1
set routing-instances pbn1 bridge-domains bdc domain-type bridge
set routing-instances pbn1 bridge-domains bdc vlan-id 11
set routing-instances pbn1 bridge-domains bdc interface xe-0/0/3.2
set routing-instances pbn1 bridge-domains bdd domain-type bridge
set routing-instances pbn1 bridge-domains bdd vlan-id 21
set routing-instances pbn1 bridge-domains bdd interface xe-0/0/3.3
set routing-instances pbn1 pbb-options peer-instance pbbn1
set routing-instances pbn1 service-groups sga service-type elan
set routing-instances pbn1 service-groups sga pbb-service-options isid 1000 vlan-id-list 10
set routing-instances pbn1 service-groups sga pbb-service-options isid 1000 vlan-id-list 11
set routing-instances pbn1 service-groups sga pbb-service-options source-bmac 00:52:52:52:52:52
set routing-instances pbn1 service-groups sgb service-type elan
set routing-instances pbn1 service-groups sgb pbb-service-options isid 2000 vlan-id-list 20
set routing-instances pbn1 service-groups sgb pbb-service-options isid 2000 vlan-id-list 21

```

Procedure

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the [CLI User Guide](#).

To configure Device PE1:

1. Set the number of aggregated Ethernet interfaces on Device PE1.

```
[edit chassis]
user@PE1# set aggregated-devices ethernet device-count 16
```

2. Set Device PE1's network services to enhanced Internet Protocol and use enhanced mode capabilities.

```
[edit chassis]
user@PE1# set chassis network-services enhanced-ip
```

3. Configure the CE-facing interfaces of Device PE1.

```
[edit interfaces]
user@PE1# set xe-1/0/0 flexible-vlan-tagging
user@PE1# set xe-1/0/0 encapsulation flexible-ethernet-services
user@PE1# set xe-1/0/0 unit 0 encapsulation vlan-bridge
user@PE1# set xe-1/0/0 unit 0 vlan-id 10
user@PE1# set xe-1/0/0 unit 1 encapsulation vlan-bridge
user@PE1# set xe-1/0/0 unit 1 vlan-id 20
user@PE1# set xe-1/2/2 flexible-vlan-tagging
user@PE1# set xe-1/2/2 encapsulation flexible-ethernet-services
user@PE1# set xe-1/2/2 unit 0 encapsulation vlan-bridge
user@PE1# set xe-1/2/2 unit 0 vlan-id 10
user@PE1# set xe-1/2/2 unit 0 family bridge filter input BRI
user@PE1# set xe-1/2/2 unit 1 encapsulation vlan-bridge
user@PE1# set xe-1/2/2 unit 1 vlan-id 20
user@PE1# set xe-1/2/2 unit 2 encapsulation vlan-bridge
user@PE1# set xe-1/2/2 unit 2 vlan-id 11
user@PE1# set xe-1/2/2 unit 2 family bridge
user@PE1# set xe-1/2/2 unit 3 encapsulation vlan-bridge
user@PE1# set xe-1/2/2 unit 3 vlan-id 21
user@PE1# set xe-1/2/2 unit 3 family bridge
```

4. Configure the interfaces connecting Device PE1 with the other PE devices.

```
[edit interfaces]
user@PE1# set ge-2/1/0 gigether-options 802.3ad ae6
user@PE1# set ge-2/1/1 unit 0 family inet address 10.0.0.1/8
```

```

user@PE1# set ge-2/1/1 unit 0 family iso
user@PE1# set ge-2/1/1 unit 0 family mpls

```

5. Configure the aggregated Ethernet bundle ae6.

```

[edit interfaces]
user@PE1# set ae6 encapsulation ethernet-bridge
user@PE1# set ae6 unit 0 family bridge

```

6. Configure the loopback interface of Device PE1.

```

[edit interfaces]
user@PE1# set interfaces lo0 unit 0 family inet address 127.0.0.1/8 primary

```

7. Configure the integrated routing and bridging (IRB) interfaces for Device PE1.

```

[edit interfaces]
user@PE1# set interfaces irb unit 0 family inet address 10.0.0.1/8
user@PE1# set interfaces irb unit 0 family iso
user@PE1# set interfaces irb unit 0 family mpls

```

8. Configure the customer backbone port (CBP) interfaces on Device PE1.

```

[edit interfaces]
user@PE1# set cbp0 unit 0 family bridge interface-mode trunk
user@PE1# set cbp0 unit 0 family bridge bridge-domain-type bvlan
user@PE1# set cbp0 unit 0 family bridge isid-list all
user@PE1# set cbp0 unit 1 family bridge interface-mode trunk
user@PE1# set cbp0 unit 1 family bridge bridge-domain-type bvlan
user@PE1# set cbp0 unit 1 family bridge isid-list all

```

9. Configure the Provider Instance Port (PIP) on Device PE1.

```

[edit interfaces]
user@PE1# set pip0 unit 0 family bridge interface-mode trunk
user@PE1# set pip0 unit 0 family bridge bridge-domain-type svlan
user@PE1# set pip0 unit 0 family bridge isid-list all-service-groups
user@PE1# set pip0 unit 1 family bridge interface-mode trunk

```

```

user@PE1# set pip0 unit 1 family bridge bridge-domain-type svlan
user@PE1# set pip0 unit 1 family bridge isid-list all-service-groups

```

10. Configure the router ID and autonomous system number for Device PE1.

```

[edit routing-options]
user@PE1# set router-id 127.0.0.1
user@PE1# set autonomous-system 65221

```

11. Configure RSVP on all the interfaces of Device PE1, excluding the management interface.

```

[edit protocols]
user@PE1# set rsvp interface all
user@PE1# set rsvp interface fxp0.0 disable

```

12. Configure MPLS on all the interfaces of Device PE1, excluding the management interface.

```

[edit protocols]
user@PE1# set mpls interface all
user@PE1# set mpls interface fxp0.0 disable

```

13. Configure LSPs from Device PE1 to all other PE devices.

```

[edit protocols]
user@PE1# set mpls label-switched-path PE1toPE2 from 127.0.0.1
user@PE1# set mpls label-switched-path PE1toPE2 to 127.0.0.2
user@PE1# set mpls label-switched-path PE1toPE3 from 127.0.0.1
user@PE1# set mpls label-switched-path PE1toPE3 to 127.0.0.3

```

14. Configure an internal BGP session under family EVPN from Device PE1 to all other PE devices.

```

[edit protocols]
user@PE1# set bgp group ibgp type internal
user@PE1# set bgp group ibgp local-address 127.0.0.1
user@PE1# set bgp group ibgp family evpn signaling
user@PE1# set bgp group ibgp neighbor 127.0.0.2
user@PE1# set bgp group ibgp neighbor 127.0.0.3

```


15. Configure OSPF on all the interfaces of Device of PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set ospf traffic-engineering
user@PE1# set ospf area 0.0.0.0 interface all
user@PE1# set ospf area 0.0.0.0 interface fxp0.0 disable
```

16. Configure a customer routing instance (I-component) on Device PE1 with type virtual switch. Assign the CBP interface, route-distinguisher, and virtual routing and forwarding (VRF) target values to the PBBN routing instance.

```
[edit routing-instances]
user@PE1# set pbbn1 instance-type virtual-switch
user@PE1# set pbbn1 interface cbp0.0
user@PE1# set pbbn1 route-distinguisher 127.0.0.1:100
user@PE1# set pbbn1 vrf-target target:100:100
```

17. Configure PBB-EVPN integration from the customer routing instance. Assign the extended I-SID list and bridge domains to the routing instance.

```
[edit routing-instances]
user@PE1# set pbbn1 protocols evpn pbb-evpn-core
user@PE1# set pbbn1 protocols evpn extended-isid-list 1000
user@PE1# set pbbn1 protocols evpn extended-isid-list 2000
user@PE1# set pbbn1 bridge-domains bda vlan-id 100
user@PE1# set pbbn1 bridge-domains bda isid-list 1000
user@PE1# set pbbn1 bridge-domains bda vlan-id-scope-local
user@PE1# set pbbn1 bridge-domains bdb vlan-id 200
user@PE1# set pbbn1 bridge-domains bdb isid-list 2000
user@PE1# set pbbn1 bridge-domains bdb vlan-id-scope-local
```

18. Configure a provider routing instance on Device PE1 with type virtual switch. Assign the PBP interface and bridge domains to the routing instance.

```
[edit routing-instances]
user@PE1# set pbn1 instance-type virtual-switch
user@PE1# set pbn1 interface pip0.0
user@PE1# set pbn1 bridge-domains bda domain-type bridge
user@PE1# set pbn1 bridge-domains bda vlan-id 10
```

```

user@PE1# set pbn1 bridge-domains bda interface xe-1/2/2.0
user@PE1# set pbn1 bridge-domains bda interface xe-1/0/0.0
user@PE1# set pbn1 bridge-domains bdb domain-type bridge
user@PE1# set pbn1 bridge-domains bdb vlan-id 20
user@PE1# set pbn1 bridge-domains bdb interface xe-1/2/2.1
user@PE1# set pbn1 bridge-domains bdb interface xe-1/0/0.1
user@PE1# set pbn1 bridge-domains bdc domain-type bridge
user@PE1# set pbn1 bridge-domains bdc vlan-id 11
user@PE1# set pbn1 bridge-domains bdc interface xe-1/2/2.2
user@PE1# set pbn1 bridge-domains bdd domain-type bridge
user@PE1# set pbn1 bridge-domains bdd vlan-id 21
user@PE1# set pbn1 bridge-domains bdd interface xe-1/2/2.3

```

19. Configure the peer PBBN routing instance in the customer routing instance.

```

[edit routing-instances]
user@PE1# set pbn1 pbb-options peer-instance pbbn1

```

20. Configure the service groups to be supported in the customer routing instance.

```

[edit routing-instances]
user@PE1# set pbn1 service-groups sga service-type elan
user@PE1# set pbn1 service-groups sga pbb-service-options isid 1000 vlan-id-list 10
user@PE1# set pbn1 service-groups sga pbb-service-options isid 1000 vlan-id-list 11
user@PE1# set pbn1 service-groups sga pbb-service-options source-bmac 00:50:50:50:50:50
user@PE1# set pbn1 service-groups sgb service-type elan
user@PE1# set pbn1 service-groups sgb pbb-service-options isid 2000 vlan-id-list 20
user@PE1# set pbn1 service-groups sgb pbb-service-options isid 2000 vlan-id-list 21

```

21. Configure the bridge domains on Device PE1.

```

[edit bridge-domains]
user@PE1# set bd vlan-id none
user@PE1# set bd interface ae6.0
user@PE1# set bd routing-interface irb.0

```

Results

From configuration mode, confirm your configuration by entering the `show chassis`, `show interfaces`, `show routing-options`, `show protocols`, `show routing-instances`, and `show bridge-domains` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show chassis
aggregated-devices {
    ethernet {
        device-count 16;
    }
}
network-services enhanced-ip;
```

```
user@PE1# show interfaces
xe-1/0/0 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 0 {
        encapsulation vlan-bridge;
        vlan-id 10;
    }
    unit 1 {
        encapsulation vlan-bridge;
        vlan-id 20;
    }
}
xe-1/2/2 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 0 {
        encapsulation vlan-bridge;
        vlan-id 10;
        family bridge {
            filter {
                input BRI; ## reference 'BRI' not found
            }
        }
    }
    unit 1 {
```

```

        encapsulation vlan-bridge;
        vlan-id 20;
    }
    unit 2 {
        encapsulation vlan-bridge;
        vlan-id 11;
        family bridge;
    }
    unit 3 {
        encapsulation vlan-bridge;
        vlan-id 21;
        family bridge;
    }
}
ge-2/1/0 {
    together-options {
        802.3ad ae6;
    }
}
ge-2/1/1 {
    unit 0 {
        family inet {
            address 10.0.0.1/8;
        }
        family iso;
        family mpls;
    }
}
ae6 {
    encapsulation ethernet-bridge;
    unit 0 {
        family bridge;
    }
}
cbp0 {
    unit 0 {
        family bridge {
            interface-mode trunk;
            bridge-domain-type bvlan;
            isid-list all;
        }
    }
    unit 1 {

```

```

        family bridge {
            interface-mode trunk;
            bridge-domain-type bvlan;
            isid-list all;
        }
    }
}
irb {
    unit 0 {
        family inet {
            address 10.0.0.1/8;
        }
        family iso;
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 127.0.0.1/8 {
                primary;
            }
        }
    }
}
pip0 {
    unit 0 {
        family bridge {
            interface-mode trunk;
            bridge-domain-type svlan;
            isid-list all-service-groups;
        }
    }
    unit 1 {
        family bridge {
            interface-mode trunk;
            bridge-domain-type svlan;
            isid-list all-service-groups;
        }
    }
}

```

```

    }
}

```

```

user@PE1# show routing-options
router-id 127.0.0.1;
autonomous-system 65221;

```

```

user@PE1# show protocols
rsvp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
mpls {
    label-switched-path PE1toPE2 {
        from 127.0.0.1;
        to 127.0.0.2;
    }
    label-switched-path PE1toPE3 {
        from 127.0.0.1;
        to 127.0.0.3;
    }
    interface all;
    interface fxp0.0 {
        disable;
    }
}
bgp {
    group ibgp {
        type internal;
        local-address 127.0.0.1;
        family evpn {
            signaling;
        }
        neighbor 127.0.0.2;
        neighbor 127.0.0.3;
    }
}
ospf {

```

```

traffic-engineering;
area 0.0.0.0 {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
}

```

```

user@PE1# show routing-instances
pbbn1 {
    instance-type virtual-switch;
    interface cbp0.0;
    route-distinguisher 127.0.0.1:100;
    vrf-target target:100:100;
    protocols {
        evpn {
            pbb-evpn-core;
            extended-isid-list [ 1000 2000 ];
        }
    }
    bridge-domains {
        bda {
            vlan-id 100;
            isid-list 1000;
            vlan-id-scope-local;
        }
        bdb {
            vlan-id 200;
            isid-list 2000;
            vlan-id-scope-local;
        }
    }
}
pbn1 {
    instance-type virtual-switch;
    interface pip0.0;
    bridge-domains {
        bda {
            domain-type bridge;
            vlan-id 10;

```

```

        interface xe-1/2/2.0;
        interface xe-1/0/0.0;
    }
    bdb {
        domain-type bridge;
        vlan-id 20;
        interface xe-1/2/2.1;
        interface xe-1/0/0.1;
    }
    bdc {
        domain-type bridge;
        vlan-id 11;
        interface xe-1/2/2.2;
    }
    bdd {
        domain-type bridge;
        vlan-id 21;
        interface xe-1/2/2.3;
    }
}
pbb-options {
    peer-instance pbbn1;
}
service-groups {
    sga {
        service-type elan;
        pbb-service-options {
            isid 1000 vlan-id-list [ 10 11 ];
            source-bmac 00:50:50:50:50:50;
        }
    }
    sgb {
        service-type elan;
        pbb-service-options {
            isid 2000 vlan-id-list [ 20 21 ];
        }
    }
}

```



```
}  
}
```

```
user@PE1# show bridge-domains  
bd {  
    vlan-id none;  
    interface ae6.0;  
    routing-interface irb.0;  
}
```

If you are done configuring the device, enter `commit` from configuration mode.

Verification

IN THIS SECTION

- [Verifying BGP Peering Status | 1470](#)
- [Verifying MPLS LSPs | 1471](#)
- [Verifying the EVPN Routing Instance | 1472](#)
- [Verifying Routing Table Entries of the EVPN Routing Instance | 1473](#)
- [Verifying the EVPN Database | 1475](#)
- [Verifying the MAC Table Entries | 1476](#)
- [Verifying the inet.3 Routing Table Entries | 1477](#)

Confirm that the configuration is working properly.

Verifying BGP Peering Status

Purpose

Verify that the BGP session is established between the PE devices.

Action

From operational mode, run the `show bgp summary` command.

```
user@PE1> show bgp summary
Groups: 1 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State   Pending
bgp.evpn.0
                8          8          0          0          0          0
Peer           AS        InPkt    OutPkt    OutQ    Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
127.0.0.2      65221         9         7         0         0        2:09 Establ
  bgp.evpn.0: 4/4/4/0
  pbbn1.evpn.0: 4/4/4/0
  __default_evpn__.evpn.0: 0/0/0/0
127.0.0.3      65221         7         7         0         0        1:25 Establ
  bgp.evpn.0: 4/4/4/0
  pbbn1.evpn.0: 4/4/4/0
  __default_evpn__.evpn.0: 0/0/0/0
```

Meaning

A BGP session is established between the PE devices.

Verifying MPLS LSPs

Purpose

Verify the MPLS LSP status on Device PE1.

Action

From operational mode, run the `show mpls lsp` command.

```
user@PE1> show mpls lsp
Ingress LSP: 2 sessions
To          From        State Rt P    ActivePath    LSPname
127.0.0.2   127.0.0.1   Up     0 *           PE1toPE2
127.0.0.3   127.0.0.1   Up     0 *           PE1toPE3
Total 2 displayed, Up 2, Down 0
```

```

Egress LSP: 2 sessions
To           From           State   Rt Style Labelin Labelout LSPname
127.0.0.1    127.0.0.3   Up      0 1 FF      3      - PE3toPE1
127.0.0.1    127.0.0.2   Up      0 1 FF      3      - PE2toPE1
Total 2 displayed, Up 2, Down 0

Transit LSP: 0 sessions

Total 0 displayed, Up 0, Down 0

```

Verifying the EVPN Routing Instance

Purpose

Verify the EVPN routing instance information.

Action

From operational mode, run the `show evpn instance extensive` command.

```

user@PE1> show evpn instance extensive
Instance: __default_evpn__
  Route Distinguisher: 127.0.0.1:0
  Number of bridge domains: 0
  Number of neighbors: 0

Instance: pbbn1
  Route Distinguisher: 127.0.0.1:100
  Per-instance MAC route label: 16
  Per-instance multicast route label: 17
PBB EVPN Core enabled
  Control word enabled
  MAC database status
                                Local  Remote
  MAC advertisements:           2      4
  MAC+IP advertisements:        0      0
  Default gateway MAC advertisements: 0      0
  Number of local interfaces: 1 (1 up)
    Interface name  ESI                                Mode           Status    AC-Role
    cbp0.0          00:00:00:00:00:00:00:00:00:00 single-homed  Up        Root
  Number of IRB interfaces: 0 (0 up)

```

```

Number of bridge domains: 2
  VLAN  Domain ID  Intfs / up  IRB intf  Mode          MAC sync  IM route label  SG
sync  IM core nexthop
      1000          0   0                Extended    Enabled   17
Disabled
      2000          0   0                Extended    Enabled   17
Disabled
Number of Bundle bridge domains: 0
Number of neighbors: 2
  Address          MAC      MAC+IP      AD      IM      ES Leaf-label
  127.0.0.2        2        0          0        2        0
  127.0.0.3        2        0          0        2        0
Number of ethernet segments: 0

```

Meaning

The output displays the `pbbn1` routing instance information, such as the integration of PBB with EVPN, the single-homed EVPN mode of operation, and the IP address of Devices PE2 and PE3 as the neighbors.

Verifying Routing Table Entries of the EVPN Routing Instance

Purpose

Verify the routing table entries of the EVPN routing instance.

Action

From operational mode, run the `show route table pbbn1.evpn.0` command.

```

user@PE1> show route table pbbn1.evpn.0
pbbn1.evpn.0: 12 destinations, 12 routes (12 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

2:127.0.0.1:100::1000::00:50:50:50:50/304 MAC/IP
      *[EVPN/170] 00:04:20
      Indirect
2:127.0.0.1:100::2000::00:1d:b5:a2:47:b0/304 MAC/IP
      *[EVPN/170] 00:04:20
      Indirect
2:127.0.0.2:100::1000::00:51:51:51:51/304 MAC/IP

```

```

*[BGP/170] 00:02:50, localpref 100, from 127.0.0.2
  AS path: I, validation-state: unverified
  > to 10.0.0.2 via irb.0, label-switched-path PE1toPE2
2:127.0.0.2:100::2000::00:23:9c:5e:a7:b0/304 MAC/IP
*[BGP/170] 00:02:50, localpref 100, from 127.0.0.2
  AS path: I, validation-state: unverified
  > to 10.0.0.2 via irb.0, label-switched-path PE1toPE2
2:127.0.0.3:100::1000::00:52:52:52:52:52/304 MAC/IP
*[BGP/170] 00:02:05, localpref 100, from 127.0.0.3
  AS path: I, validation-state: unverified
  > to 172.17.0.1 via ge-2/1/1.0, label-switched-path PE1toPE3
2:127.0.0.3:100::2000::5c:5e:ab:0d:3a:b8/304 MAC/IP
*[BGP/170] 00:02:05, localpref 100, from 127.0.0.3
  AS path: I, validation-state: unverified
  > to 172.17.0.1 via ge-2/1/1.0, label-switched-path PE1toPE3
3:127.0.0.1:100::1000::127.0.0.1/248 IM
*[EVPN/170] 00:04:20
  Indirect
3:127.0.0.1:100::2000::127.0.0.1/248 IM
*[EVPN/170] 00:04:20
  Indirect
3:127.0.0.2:100::1000::127.0.0.2/248 IM
*[BGP/170] 00:02:50, localpref 100, from 127.0.0.2
  AS path: I, validation-state: unverified
  > to 10.0.0.2 via irb.0, label-switched-path PE1toPE2
3:127.0.0.2:100::2000::127.0.0.2/248 IM
*[BGP/170] 00:02:50, localpref 100, from 127.0.0.2
  AS path: I, validation-state: unverified
  > to 10.0.0.2 via irb.0, label-switched-path PE1toPE2
3:127.0.0.3:100::1000::127.0.0.3/248 IM
*[BGP/170] 00:02:05, localpref 100, from 127.0.0.3
  AS path: I, validation-state: unverified
  > to 172.17.0.1 via ge-2/1/1.0, label-switched-path PE1toPE3
3:127.0.0.3:100::2000::127.0.0.3/248 IM
*[BGP/170] 00:02:05, localpref 100, from 127.0.0.3
  AS path: I, validation-state: unverified
  > to 172.17.0.1 via ge-2/1/1.0, label-switched-path PE1toPE3

```

Meaning

The output displays the use of IRB interfaces for routing the LSPs between the PE devices.

Verifying the EVPN Database

Purpose

Verify the EVPN database information on the PE devices.

Action

From operational mode, run the `show evpn database` command.

```
user@PE1> show evpn database
```

```
Instance: pbbn1
```

VLAN	DomainId	MAC address	Active source	Timestamp	IP address
	2000	00:1d:b5:a2:47:b0	Local	Apr 14 13:48:51	
	2000	00:23:9c:5e:a7:b0	127.0.0.2	Apr 14 13:53:04	
	2000	5c:5e:ab:0d:3a:b8	127.0.0.3	Apr 14 13:53:38	
	1000	00:50:50:50:50:50	Local	Apr 14 13:48:51	
	1000	00:51:51:51:51:51	127.0.0.2	Apr 14 13:53:04	
	1000	00:52:52:52:52:52	127.0.0.3	Apr 14 13:53:38	

```
user@PE2> show evpn database
```

```
Instance: pbbn1
```

VLAN	DomainId	MAC address	Active source	Timestamp	IP address
	2000	00:1d:b5:a2:47:b0	127.0.0.1	Apr 14 13:53:04	
	2000	00:23:9c:5e:a7:b0	Local	Apr 14 13:48:46	
	2000	5c:5e:ab:0d:3a:b8	127.0.0.3	Apr 14 13:53:37	
	1000	00:50:50:50:50:50	127.0.0.1	Apr 14 13:53:04	
	1000	00:51:51:51:51:51	Local	Apr 14 13:48:46	
	1000	00:52:52:52:52:52	127.0.0.3	Apr 14 13:53:37	

```
user@PE3> show evpn database
```

```
Instance: pbbn1
```

VLAN	DomainId	MAC address	Active source	Timestamp	IP address
	1000	00:50:50:50:50:50	127.0.0.1	Apr 14 13:53:34	
	1000	00:51:51:51:51:51	127.0.0.2	Apr 14 13:53:27	
	1000	00:52:52:52:52:52	Local	Apr 14 13:52:04	
	2000	00:1d:b5:a2:47:b0	127.0.0.1	Apr 14 13:53:34	

```

2000      00:23:9c:5e:a7:b0  127.0.0.2      Apr 14 13:53:27
2000      5c:5e:ab:0d:3a:b8  Local

```

Verifying the MAC Table Entries

Purpose

Verify the bridge MAC table entries.

Action

From operational mode, run the `show bridge mac-table` command.

```

user@PE1> show bridge mac-table
MAC flags      (S -static MAC, D -dynamic MAC, L -locally learned, C -Control MAC
               0 -OVSDB MAC, SE -Statistics enabled, NM -Non configured MAC, R -Remote PE MAC, P -Pinned
               MAC)

Routing instance : default-switch
Bridging domain : bd, VLAN : none

```

MAC address	MAC flags	Logical interface	NH Index	MAC property
00:23:9c:5e:a7:f0	D	ae6.0		

```

MAC flags      (S -static MAC, D -dynamic MAC, L -locally learned, C -Control MAC
               0 -OVSDB MAC, SE -Statistics enabled, NM -Non configured MAC, R -Remote PE MAC, P -Pinned
               MAC)

Routing instance : pbbn1
Bridging domain : bda, VLAN : 100

```

MAC address	MAC flags	Logical interface	NH Index	MAC property
00:51:51:51:51:51	DC		1048576	
00:52:52:52:52:52	DC		1048581	
01:1e:83:00:03:e8	DC		1048578	

```

MAC flags      (S -static MAC, D -dynamic MAC, L -locally learned, C -Control MAC
               0 -OVSDB MAC, SE -Statistics enabled, NM -Non configured MAC, R -Remote PE MAC, P -Pinned
               MAC)

Routing instance : pbbn1

```

Bridging domain : bdb, VLAN : 200

MAC address	MAC flags	Logical interface	NH Index	MAC property
00:23:9c:5e:a7:b0	DC		1048576	
01:1e:83:00:07:d0	DC		1048577	
5c:5e:ab:0d:3a:b8	DC		1048581	

MAC flags (S -static MAC, D -dynamic MAC,
SE -Statistics enabled, NM -Non configured MAC, P -Pinned MAC)

Routing instance : pbn1

Bridging domain : bda, ISID : 1000, VLAN : 10

MAC address	MAC flags	Logical interface	Remote BEB address
00:00:00:00:0a:00	D	xe-1/0/0.0	
00:00:00:00:0a:01	D	xe-1/0/0.0	
00:00:00:00:0a:02	D	xe-1/0/0.0	
00:00:00:00:0a:03	D	xe-1/0/0.0	
00:00:00:00:0a:04	D	xe-1/0/0.0	
00:00:00:00:0a:05	D	xe-1/0/0.0	
00:00:00:00:0a:06	D	xe-1/0/0.0	
00:00:00:00:0a:07	D	xe-1/0/0.0	
00:00:00:00:0a:08	D	xe-1/0/0.0	
00:00:00:00:0a:09	D	xe-1/0/0.0	

Meaning

The output displays the MAC addresses associated with the ae6 aggregated Ethernet bundle.

Verifying the inet.3 Routing Table Entries

Purpose

Verify the inet.3 routing table entries on Device PE1.

Action

From operational mode, run the show route table inet.3 command.

```
user@PE1> show route table inet.3
inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
```


+ = Active Route, - = Last Active, * = Both

```
127.0.0.2/8    *[RSVP/7/1] 00:11:15, metric 1
                > to 10.0.0.2 via irb.0, label-switched-path PE1toPE2
127.0.0.3/8    *[RSVP/7/1] 00:09:48, metric 1
                > to 172.17.0.1 via ge-2/1/1.0, label-switched-path PE1toPE3
```

Meaning

The LSPs to Device PE2 and PE3 are routed using the IRB interface.

RELATED DOCUMENTATION

[Provider Backbone Bridging \(PBB\) and EVPN Integration Overview | 1414](#)

[Example: Configuring PBB with Multihomed EVPN | 1478](#)

[pbb-evpn-core | 1640](#)

Example: Configuring PBB with Multihomed EVPN

IN THIS SECTION

- [Requirements | 1478](#)
- [Overview and Topology | 1479](#)
- [Configuration | 1480](#)
- [Verification | 1512](#)

This example shows how to integrate provider backbone bridging (PBB) with Ethernet VPN (EVPN). With this integration, the control plane operations in the core are simplified, providing faster convergence and scalability enhancements than regular EVPN. The PBB-EVPN applications include Data Center Interconnect (DCI) and carrier Ethernet E-LAN services.

Requirements

This example uses the following hardware and software components:

- Four provider edge (PE) devices connected to two common multihomed customer sites, and each PE device is connected to a host.
- Two multihomed customer edge (CE) devices.
- Junos OS Release 17.2R1 or later running on all the PE routers.

Before you begin:

- Configure the device interfaces.
- Configure an IGP, such as OSPF, on all the PE devices.
- Establish an internal BGP session between the PE devices.
- Enable RSVP on the PE devices.
- Configure MPLS and label-switched paths (LSPs) between the PE devices.

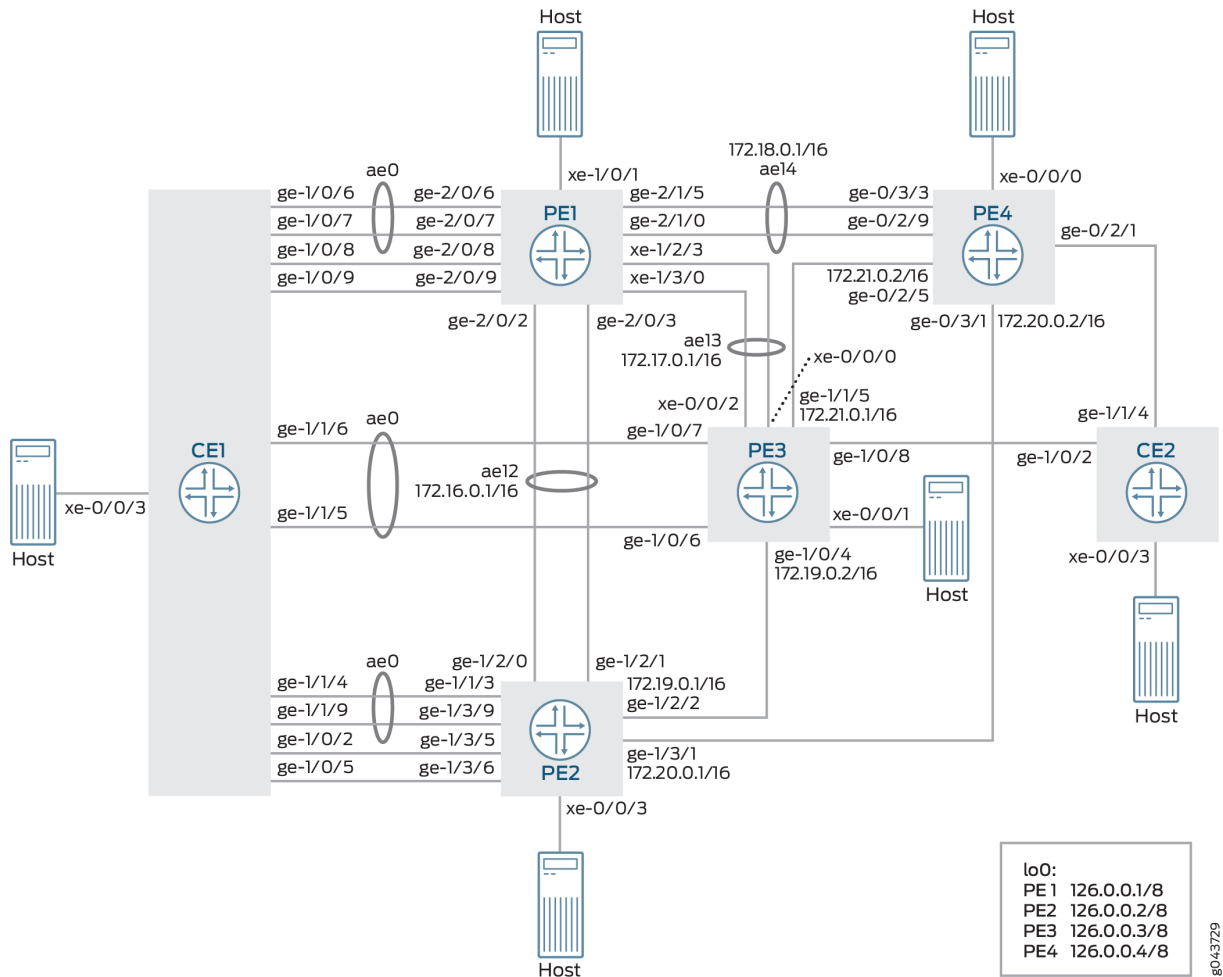
Overview and Topology

Starting in Junos OS Release 17.2R1, PBB is integrated with Ethernet VPN (EVPN) to enable significant reduction in the control plane learning across the core, allowing a huge number of Layer 2 services, such as data center connectivity, to transit the network in a simplified manner.

In a PBB-EVPN network, the backbone core bridge (BCB) device in the PBB core is replaced with MPLS, while retaining the service scaling properties of the PBB backbone edge bridge (BEB). The B-component (provider routing instance) is signaled using EVPN BGP signaling and encapsulated inside MPLS using provider edge (PE) and provider (P) devices. Thus, PBB-EVPN combines the vast scaling property of PBB

with the simplicity of a traditional basic MPLS core network, resulting in significant reduction in the amount of network-wide state information, as opposed to regular PBB.

Figure 159: PBB with Active/Standby EVPN Multihoming



In [Figure 159 on page 1480](#), PBB is integrated with EVPN, where the CE devices are multihomed in the active/standby mode. Device CE1 is multihomed to Devices PE1, PE2, and PE3, and Device CE2 is multihomed to Device PE3 and PE4.

Configuration

IN THIS SECTION

- CLI Quick Configuration | 1481

- [Configuring Device CE1 | 1493](#)
- [Configuring Device PE1 | 1498](#)

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the [edit] hierarchy level, and then enter `commit` from configuration mode.

CE1

```
set chassis aggregated-devices ethernet device-count 50
set chassis network-services enhanced-ip
set interfaces xe-0/0/3 flexible-vlan-tagging
set interfaces xe-0/0/3 encapsulation flexible-ethernet-services
set interfaces xe-0/0/3 unit 0 encapsulation vlan-bridge
set interfaces xe-0/0/3 unit 0 vlan-id 10
set interfaces xe-0/0/3 unit 1 encapsulation vlan-bridge
set interfaces xe-0/0/3 unit 1 vlan-id 20
set interfaces xe-0/0/3 unit 2 encapsulation vlan-bridge
set interfaces xe-0/0/3 unit 2 vlan-id 30
set interfaces ge-1/0/2 flexible-vlan-tagging
set interfaces ge-1/0/2 encapsulation flexible-ethernet-services
set interfaces ge-1/0/2 unit 1 encapsulation vlan-bridge
set interfaces ge-1/0/2 unit 1 vlan-id 20
set interfaces ge-1/0/5 flexible-vlan-tagging
set interfaces ge-1/0/5 encapsulation flexible-ethernet-services
set interfaces ge-1/0/5 unit 2 encapsulation vlan-bridge
set interfaces ge-1/0/5 unit 2 vlan-id 30
set interfaces ge-1/0/6 gigether-options 802.3ad ae0
set interfaces ge-1/0/7 gigether-options 802.3ad ae0
set interfaces ge-1/0/8 flexible-vlan-tagging
set interfaces ge-1/0/8 encapsulation flexible-ethernet-services
set interfaces ge-1/0/8 unit 1 encapsulation vlan-bridge
set interfaces ge-1/0/8 unit 1 vlan-id 20
set interfaces ge-1/0/9 flexible-vlan-tagging
set interfaces ge-1/0/9 encapsulation flexible-ethernet-services
set interfaces ge-1/0/9 unit 2 encapsulation vlan-bridge
set interfaces ge-1/0/9 unit 2 vlan-id 30
```

```

set interfaces ge-1/1/4 gigether-options 802.3ad ae0
set interfaces ge-1/1/5 gigether-options 802.3ad ae0
set interfaces ge-1/1/6 gigether-options 802.3ad ae0
set interfaces ge-1/1/9 gigether-options 802.3ad ae0
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 aggregated-ether-options minimum-links 1
set interfaces ae0 unit 0 encapsulation vlan-bridge
set interfaces ae0 unit 0 vlan-id 10
set bridge-domains bd10 domain-type bridge
set bridge-domains bd10 vlan-id 10
set bridge-domains bd10 interface ae0.0
set bridge-domains bd10 interface xe-0/0/3.0
set bridge-domains bd20 domain-type bridge
set bridge-domains bd20 vlan-id 20
set bridge-domains bd20 interface xe-0/0/3.1
set bridge-domains bd20 interface ge-1/0/8.1
set bridge-domains bd20 interface ge-1/0/2.1
set bridge-domains bd30 domain-type bridge
set bridge-domains bd30 vlan-id 30
set bridge-domains bd30 interface xe-0/0/3.2
set bridge-domains bd30 interface ge-1/0/9.2
set bridge-domains bd30 interface ge-1/0/5.2

```

CE2

```

set chassis aggregated-devices ethernet device-count 50
set chassis network-services enhanced-ip
set interfaces xe-0/0/3 flexible-vlan-tagging
set interfaces xe-0/0/3 encapsulation flexible-ethernet-services
set interfaces xe-0/0/3 unit 0 encapsulation vlan-bridge
set interfaces xe-0/0/3 unit 0 vlan-id 10
set interfaces xe-0/0/3 unit 0 family bridge filter output f_log
set interfaces xe-0/0/3 unit 1 encapsulation vlan-bridge
set interfaces xe-0/0/3 unit 1 vlan-id 20
set interfaces xe-0/0/3 unit 2 encapsulation vlan-bridge
set interfaces xe-0/0/3 unit 2 vlan-id 30
set interfaces ge-1/0/2 flexible-vlan-tagging
set interfaces ge-1/0/2 encapsulation flexible-ethernet-services
set interfaces ge-1/0/2 unit 0 encapsulation vlan-bridge
set interfaces ge-1/0/2 unit 0 vlan-id 10
set interfaces ge-1/0/2 unit 1 encapsulation vlan-bridge

```

```

set interfaces ge-1/0/2 unit 1 vlan-id 20
set interfaces ge-1/0/2 unit 2 encapsulation vlan-bridge
set interfaces ge-1/0/2 unit 2 vlan-id 30
set interfaces ge-1/1/4 flexible-vlan-tagging
set interfaces ge-1/1/4 encapsulation flexible-ethernet-services
set interfaces ge-1/1/4 unit 0 encapsulation vlan-bridge
set interfaces ge-1/1/4 unit 0 vlan-id 10
set interfaces ge-1/1/4 unit 1 encapsulation vlan-bridge
set interfaces ge-1/1/4 unit 1 vlan-id 20
set interfaces ge-1/1/4 unit 2 encapsulation vlan-bridge
set interfaces ge-1/1/4 unit 2 vlan-id 30
set bridge-domains bd10 domain-type bridge
set bridge-domains bd10 vlan-id 10
set bridge-domains bd10 interface ge-1/1/4.0
set bridge-domains bd10 interface ge-1/0/2.0
set bridge-domains bd10 interface xe-0/0/3.0
set bridge-domains bd20 domain-type bridge
set bridge-domains bd20 vlan-id 20
set bridge-domains bd20 interface ge-1/1/4.1
set bridge-domains bd20 interface ge-1/0/2.1
set bridge-domains bd20 interface xe-0/0/3.1
set bridge-domains bd30 domain-type bridge
set bridge-domains bd30 vlan-id 30
set bridge-domains bd30 interface xe-0/0/3.2
set bridge-domains bd30 interface ge-1/1/4.2
set bridge-domains bd30 interface ge-1/0/2.2

```

PE1

```

set chassis aggregated-devices ethernet device-count 50
set chassis network-services enhanced-ip
set interfaces xe-1/0/1 flexible-vlan-tagging
set interfaces xe-1/0/1 encapsulation flexible-ethernet-services
set interfaces xe-1/0/1 unit 0 encapsulation vlan-bridge
set interfaces xe-1/0/1 unit 0 vlan-id 10
set interfaces xe-1/2/3 gigether-options 802.3ad ae13
set interfaces xe-1/3/0 gigether-options 802.3ad ae13
set interfaces ge-2/0/2 gigether-options 802.3ad ae12
set interfaces ge-2/0/3 gigether-options 802.3ad ae12
set interfaces ge-2/0/6 gigether-options 802.3ad ae0
set interfaces ge-2/0/7 gigether-options 802.3ad ae0
set interfaces ge-2/0/8 flexible-vlan-tagging

```

```

set interfaces ge-2/0/8 encapsulation flexible-ethernet-services
set interfaces ge-2/0/8 esi 00:22:22:22:22:22:22:22:22:22
set interfaces ge-2/0/8 esi single-active
set interfaces ge-2/0/8 esi source-bmac 00:22:22:22:22:22
set interfaces ge-2/0/8 unit 0 encapsulation vlan-bridge
set interfaces ge-2/0/8 unit 0 vlan-id 20
set interfaces ge-2/0/9 flexible-vlan-tagging
set interfaces ge-2/0/9 encapsulation flexible-ethernet-services
set interfaces ge-2/0/9 esi 00:33:33:33:33:33:33:33:33:33
set interfaces ge-2/0/9 esi single-active
set interfaces ge-2/0/9 esi source-bmac 00:33:33:33:33:33
set interfaces ge-2/0/9 unit 0 encapsulation vlan-bridge
set interfaces ge-2/0/9 unit 0 vlan-id 30
set interfaces ge-2/1/0 gigether-options 802.3ad ae14
set interfaces ge-2/1/5 gigether-options 802.3ad ae14
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 esi 00:11:11:11:11:11:11:11:11:11
set interfaces ae0 esi single-active
set interfaces ae0 esi source-bmac 00:11:11:11:11:11
set interfaces ae0 unit 0 encapsulation vlan-bridge
set interfaces ae0 unit 0 vlan-id 10
set interfaces ae12 flexible-vlan-tagging
set interfaces ae12 encapsulation flexible-ethernet-services
set interfaces ae12 aggregated-ether-options minimum-links 1
set interfaces ae12 unit 0 vlan-id 1200
set interfaces ae12 unit 0 family inet address 172.16.0.1/16
set interfaces ae12 unit 0 family iso
set interfaces ae12 unit 0 family mpls
set interfaces ae13 flexible-vlan-tagging
set interfaces ae13 encapsulation flexible-ethernet-services
set interfaces ae13 aggregated-ether-options minimum-links 1
set interfaces ae13 unit 0 vlan-tags outer 1300
set interfaces ae13 unit 0 vlan-tags inner 13
set interfaces ae13 unit 0 family inet address 172.17.0.1/16
set interfaces ae13 unit 0 family iso
set interfaces ae13 unit 0 family mpls
set interfaces ae14 aggregated-ether-options minimum-links 1
set interfaces ae14 unit 0 family inet address 172.18.0.1/16
set interfaces ae14 unit 0 family iso
set interfaces ae14 unit 0 family mpls
set interfaces cbp0 unit 0 family bridge interface-mode trunk
set interfaces cbp0 unit 0 family bridge bridge-domain-type bvlan

```

```

set interfaces cbp0 unit 0 family bridge isid-list all
set interfaces cbp0 unit 1 family bridge interface-mode trunk
set interfaces cbp0 unit 1 family bridge bridge-domain-type bvlan
set interfaces cbp0 unit 1 family bridge isid-list all
set interfaces lo0 unit 0 family inet address 127.0.0.1/8 primary
set interfaces lo0 unit 0 family iso
set interfaces pip0 unit 0 family bridge interface-mode trunk
set interfaces pip0 unit 0 family bridge bridge-domain-type svlan
set interfaces pip0 unit 0 family bridge isid-list all-service-groups
set interfaces pip0 unit 1 family bridge interface-mode trunk
set interfaces pip0 unit 1 family bridge bridge-domain-type svlan
set interfaces pip0 unit 1 family bridge isid-list all-service-groups
set routing-options router-id 127.0.0.1
set routing-options autonomous-system 65221
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path pe1tope2 from 127.0.0.1
set protocols mpls label-switched-path pe1tope2 to 127.0.0.2
set protocols mpls label-switched-path pe1tope2 primary direct_to_pe2
set protocols mpls label-switched-path pe1tope3 from 127.0.0.1
set protocols mpls label-switched-path pe1tope3 to 127.0.0.3
set protocols mpls label-switched-path pe1tope3 primary direct_to_pe3
set protocols mpls label-switched-path pe1tope4 from 127.0.0.1
set protocols mpls label-switched-path pe1tope4 to 127.0.0.4
set protocols mpls label-switched-path pe1tope4 primary direct_to_pe4
set protocols mpls path direct_to_pe2 100.12.1.2 strict
set protocols mpls path direct_to_pe3 100.13.1.3 strict
set protocols mpls path direct_to_pe4 100.14.1.4 strict
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 127.0.0.1
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 127.0.0.2
set protocols bgp group ibgp neighbor 127.0.0.3
set protocols bgp group ibgp neighbor 127.0.0.4
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set routing-instances pbbn1 instance-type virtual-switch
set routing-instances pbbn1 interface cbp0.0
set routing-instances pbbn1 route-distinguisher 127.0.0.1:1
set routing-instances pbbn1 vrf-target target:100:1

```



```

set routing-instances pbbn1 protocols evpn pbb-evpn-core
set routing-instances pbbn1 protocols evpn extended-isid-list 1000
set routing-instances pbbn1 bridge-domains bda vlan-id 100
set routing-instances pbbn1 bridge-domains bda isid-list 1000
set routing-instances pbbn1 bridge-domains bda vlan-id-scope-local
set routing-instances pbn1 instance-type virtual-switch
set routing-instances pbn1 interface pip0.0
set routing-instances pbn1 bridge-domains bda domain-type bridge
set routing-instances pbn1 bridge-domains bda vlan-id 10
set routing-instances pbn1 bridge-domains bda interface ae0.0
set routing-instances pbn1 bridge-domains bda interface xe-1/0/1.0
set routing-instances pbn1 bridge-domains bda interface ge-2/0/9.0
set routing-instances pbn1 bridge-domains bda interface ge-2/0/8.0
set routing-instances pbn1 pbb-options peer-instance pbbn1
set routing-instances pbn1 service-groups sga service-type elan
set routing-instances pbn1 service-groups sga pbb-service-options isid 1000 vlan-id-list 10

```

PE2

```

set chassis aggregated-devices ethernet device-count 50
set chassis network-services enhanced-ip
set interfaces xe-0/0/3 flexible-vlan-tagging
set interfaces xe-0/0/3 encapsulation flexible-ethernet-services
set interfaces xe-0/0/3 unit 0 encapsulation vlan-bridge
set interfaces xe-0/0/3 unit 0 vlan-id 10
set interfaces ge-1/2/0 gigether-options 802.3ad ae12
set interfaces ge-1/2/1 gigether-options 802.3ad ae12
set interfaces ge-1/2/2 unit 0 family inet address 172.19.0.1/16
set interfaces ge-1/2/2 unit 0 family iso
set interfaces ge-1/2/2 unit 0 family mpls
set interfaces ge-1/3/1 unit 0 family inet address 172.20.0.1/16
set interfaces ge-1/3/1 unit 0 family iso
set interfaces ge-1/3/1 unit 0 family mpls
set interfaces ge-1/3/3 gigether-options 802.3ad ae0
set interfaces ge-1/3/5 flexible-vlan-tagging
set interfaces ge-1/3/5 encapsulation flexible-ethernet-services
set interfaces ge-1/3/5 esi 00:22:22:22:22:22:22:22:22:22
set interfaces ge-1/3/5 esi single-active
set interfaces ge-1/3/5 esi source-bmac 00:22:22:22:22:23
set interfaces ge-1/3/5 unit 0 encapsulation vlan-bridge
set interfaces ge-1/3/5 unit 0 vlan-id 20
set interfaces ge-1/3/6 flexible-vlan-tagging

```

```

set interfaces ge-1/3/6 encapsulation flexible-ethernet-services
set interfaces ge-1/3/6 esi 00:33:33:33:33:33:33:33:33
set interfaces ge-1/3/6 esi single-active
set interfaces ge-1/3/6 esi source-bmac 00:33:33:33:33:34
set interfaces ge-1/3/6 unit 0 encapsulation vlan-bridge
set interfaces ge-1/3/6 unit 0 vlan-id 30
set interfaces ge-1/3/9 gigether-options 802.3ad ae0
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 esi 00:11:11:11:11:11:11:11:11
set interfaces ae0 esi single-active
set interfaces ae0 esi source-bmac 00:11:11:11:11:12
set interfaces ae0 unit 0 encapsulation vlan-bridge
set interfaces ae0 unit 0 vlan-id 10
set interfaces ae0 unit 0 family bridge filter output f_log
set interfaces ae12 flexible-vlan-tagging
set interfaces ae12 encapsulation flexible-ethernet-services
set interfaces ae12 aggregated-ether-options minimum-links 1
set interfaces ae12 unit 0 vlan-id 1200
set interfaces ae12 unit 0 family inet address 100.12.1.2/24
set interfaces ae12 unit 0 family iso
set interfaces ae12 unit 0 family mpls
set interfaces cbp0 unit 0 family bridge interface-mode trunk
set interfaces cbp0 unit 0 family bridge bridge-domain-type bvlan
set interfaces cbp0 unit 0 family bridge isid-list all
set interfaces cbp0 unit 1 family bridge interface-mode trunk
set interfaces cbp0 unit 1 family bridge bridge-domain-type bvlan
set interfaces cbp0 unit 1 family bridge isid-list all
set interfaces lo0 unit 0 family inet address 127.0.0.2/8 primary
set interfaces lo0 unit 0 family iso
set interfaces pip0 unit 0 family bridge interface-mode trunk
set interfaces pip0 unit 0 family bridge bridge-domain-type svlan
set interfaces pip0 unit 0 family bridge isid-list all-service-groups
set interfaces pip0 unit 1 family bridge interface-mode trunk
set interfaces pip0 unit 1 family bridge bridge-domain-type svlan
set interfaces pip0 unit 1 family bridge isid-list all-service-groups
set routing-options router-id 127.0.0.2
set routing-options autonomous-system 65221
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path pe2tope1 from 127.0.0.2
set protocols mpls label-switched-path pe2tope1 to 127.0.0.1
set protocols mpls label-switched-path pe2tope1 primary direct_to_pe1

```

```

set protocols mpls label-switched-path pe2tope3 from 127.0.0.2
set protocols mpls label-switched-path pe2tope3 to 127.0.0.3
set protocols mpls label-switched-path pe2tope3 primary direct_to_pe3
set protocols mpls label-switched-path pe2tope4 from 127.0.0.2
set protocols mpls label-switched-path pe2tope4 to 127.0.0.4
set protocols mpls label-switched-path pe2tope4 primary direct_to_pe4
set protocols mpls path direct_to_pe1 172.16.0.1 strict
set protocols mpls path direct_to_pe3 100.23.1.3 strict
set protocols mpls path direct_to_pe4 172.20.0.2 strict
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 127.0.0.2
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 127.0.0.1
set protocols bgp group ibgp neighbor 127.0.0.3
set protocols bgp group ibgp neighbor 127.0.0.4
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface all
set routing-instances pbbn1 instance-type virtual-switch
set routing-instances pbbn1 interface cbp0.0
set routing-instances pbbn1 route-distinguisher 127.0.0.2:1
set routing-instances pbbn1 vrf-target target:100:1
set routing-instances pbbn1 protocols evpn pbb-evpn-core
set routing-instances pbbn1 protocols evpn extended-isid-list 1000
set routing-instances pbbn1 bridge-domains bda vlan-id 100
set routing-instances pbbn1 bridge-domains bda isid-list 1000
set routing-instances pbbn1 bridge-domains bda vlan-id-scope-local
set routing-instances pbn1 instance-type virtual-switch
set routing-instances pbn1 interface pip0.0
set routing-instances pbn1 bridge-domains bda domain-type bridge
set routing-instances pbn1 bridge-domains bda vlan-id 10
set routing-instances pbn1 bridge-domains bda interface ae0.0
set routing-instances pbn1 bridge-domains bda interface xe-0/0/3.0
set routing-instances pbn1 bridge-domains bda interface ge-1/3/6.0
set routing-instances pbn1 bridge-domains bda interface ge-1/3/5.0
set routing-instances pbn1 bridge-domains bda bridge-options mac-statistics
set routing-instances pbn1 pbb-options peer-instance pbbn1
set routing-instances pbn1 service-groups sga service-type elan
set routing-instances pbn1 service-groups sga pbb-service-options isid 1000 vlan-id-list 10

```

PE3

```

set chassis aggregated-devices ethernet device-count 50
set chassis network-services enhanced-ip
set interfaces xe-0/0/0 gigether-options 802.3ad ae13
set interfaces xe-0/0/1 flexible-vlan-tagging
set interfaces xe-0/0/1 encapsulation flexible-ethernet-services
set interfaces xe-0/0/1 unit 0 encapsulation vlan-bridge
set interfaces xe-0/0/1 unit 0 vlan-id 10
set interfaces xe-0/0/2 gigether-options 802.3ad ae13
set interfaces ge-1/0/4 unit 0 family inet address 100.23.1.3/24
set interfaces ge-1/0/4 unit 0 family iso
set interfaces ge-1/0/4 unit 0 family mpls
set interfaces ge-1/0/6 gigether-options 802.3ad ae0
set interfaces ge-1/0/7 gigether-options 802.3ad ae0
set interfaces ge-1/0/8 flexible-vlan-tagging
set interfaces ge-1/0/8 encapsulation flexible-ethernet-services
set interfaces ge-1/0/8 esi 00:44:44:44:44:44:44:44:44:44
set interfaces ge-1/0/8 esi single-active
set interfaces ge-1/0/8 esi source-bmac 00:44:44:44:44:44
set interfaces ge-1/0/8 unit 0 encapsulation vlan-bridge
set interfaces ge-1/0/8 unit 0 vlan-id 10
set interfaces ge-1/1/5 unit 0 family inet address 172.21.0.1/16
set interfaces ge-1/1/5 unit 0 family iso
set interfaces ge-1/1/5 unit 0 family mpls
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 esi 00:11:11:11:11:11:11:11:11:11
set interfaces ae0 esi single-active
set interfaces ae0 esi source-bmac 00:11:11:11:11:13
set interfaces ae0 unit 0 encapsulation vlan-bridge
set interfaces ae0 unit 0 vlan-id 10
set interfaces ae13 flexible-vlan-tagging
set interfaces ae13 encapsulation flexible-ethernet-services
set interfaces ae13 aggregated-ether-options minimum-links 1
set interfaces ae13 unit 0 vlan-tags outer 1300
set interfaces ae13 unit 0 vlan-tags inner 13
set interfaces ae13 unit 0 family inet address 100.13.1.3/24
set interfaces ae13 unit 0 family iso
set interfaces ae13 unit 0 family mpls
set interfaces cbp0 unit 0 family bridge interface-mode trunk
set interfaces cbp0 unit 0 family bridge bridge-domain-type bvlan

```

```

set interfaces cbp0 unit 0 family bridge isid-list all
set interfaces cbp0 unit 1 family bridge interface-mode trunk
set interfaces cbp0 unit 1 family bridge bridge-domain-type bvlan
set interfaces cbp0 unit 1 family bridge isid-list all
set interfaces lo0 unit 0 family inet address 127.0.0.3/8 primary
set interfaces lo0 unit 0 family iso
set interfaces pip0 unit 0 family bridge interface-mode trunk
set interfaces pip0 unit 0 family bridge bridge-domain-type svlan
set interfaces pip0 unit 0 family bridge isid-list all-service-groups
set interfaces pip0 unit 1 family bridge interface-mode trunk
set interfaces pip0 unit 1 family bridge bridge-domain-type svlan
set interfaces pip0 unit 1 family bridge isid-list all-service-groups
set routing-options router-id 127.0.0.3
set routing-options autonomous-system 65221
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path pe3tope1 from 127.0.0.3
set protocols mpls label-switched-path pe3tope1 to 127.0.0.1
set protocols mpls label-switched-path pe3tope1 primary direct_to_pe1
set protocols mpls label-switched-path pe3tope2 from 127.0.0.3
set protocols mpls label-switched-path pe3tope2 to 127.0.0.2
set protocols mpls label-switched-path pe3tope2 primary direct_to_pe2
set protocols mpls label-switched-path pe3tope4 from 127.0.0.3
set protocols mpls label-switched-path pe3tope4 to 127.0.0.4
set protocols mpls label-switched-path pe3tope4 primary direct_to_pe4
set protocols mpls path direct_to_pe1 172.17.0.1 strict
set protocols mpls path direct_to_pe2 172.19.0.1 strict
set protocols mpls path direct_to_pe4 172.21.0.2 strict
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 127.0.0.3
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 127.0.0.1
set protocols bgp group ibgp neighbor 127.0.0.2
set protocols bgp group ibgp neighbor 127.0.0.4
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set routing-instances pbbn1 instance-type virtual-switch
set routing-instances pbbn1 interface cbp0.0
set routing-instances pbbn1 route-distinguisher 127.0.0.3:1
set routing-instances pbbn1 vrf-target target:100:1

```

```

set routing-instances pbbn1 protocols evpn traceoptions file pbbevpn.log
set routing-instances pbbn1 protocols evpn traceoptions file size 500m
set routing-instances pbbn1 protocols evpn traceoptions flag all
set routing-instances pbbn1 protocols evpn control-word
set routing-instances pbbn1 protocols evpn pbb-evpn-core
set routing-instances pbbn1 protocols evpn extended-isid-list 1000
set routing-instances pbbn1 bridge-domains bda vlan-id 100
set routing-instances pbbn1 bridge-domains bda isid-list 1000
set routing-instances pbbn1 bridge-domains bda vlan-id-scope-local
set routing-instances pbn1 instance-type virtual-switch
set routing-instances pbn1 interface pip0.0
set routing-instances pbn1 bridge-domains bda domain-type bridge
set routing-instances pbn1 bridge-domains bda vlan-id 10
set routing-instances pbn1 bridge-domains bda interface ae0.0
set routing-instances pbn1 bridge-domains bda interface xe-0/0/1.0
set routing-instances pbn1 bridge-domains bda interface ge-1/0/8.0
set routing-instances pbn1 bridge-domains bda bridge-options mac-statistics
set routing-instances pbn1 pbb-options peer-instance pbbn1
set routing-instances pbn1 service-groups sga service-type elan
set routing-instances pbn1 service-groups sga pbb-service-options isid 1000 vlan-id-list 10

```

PE4

```

set chassis aggregated-devices ethernet device-count 50
set chassis network-services enhanced-ip
set interfaces xe-0/0/0 flexible-vlan-tagging
set interfaces xe-0/0/0 encapsulation flexible-ethernet-services
set interfaces xe-0/0/0 unit 0 encapsulation vlan-bridge
set interfaces xe-0/0/0 unit 0 vlan-id 10
set interfaces ge-0/2/1 flexible-vlan-tagging
set interfaces ge-0/2/1 encapsulation flexible-ethernet-services
set interfaces ge-0/2/1 esi 00:44:44:44:44:44:44:44:44
set interfaces ge-0/2/1 esi single-active
set interfaces ge-0/2/1 esi source-bmac 00:44:44:44:44:45
set interfaces ge-0/2/1 unit 0 encapsulation vlan-bridge
set interfaces ge-0/2/1 unit 0 vlan-id 10
set interfaces ge-0/2/5 unit 0 family inet address 172.21.0.2/16
set interfaces ge-0/2/5 unit 0 family iso
set interfaces ge-0/2/5 unit 0 family mpls
set interfaces ge-0/2/9 gigether-options 802.3ad ae14
set interfaces ge-0/3/1 unit 0 family inet address 172.20.0.2/16
set interfaces ge-0/3/1 unit 0 family iso

```

```

set interfaces ge-0/3/1 unit 0 family mpls
set interfaces ge-0/3/3 gigether-options 802.3ad ae14
set interfaces ae14 aggregated-ether-options minimum-links 1
set interfaces ae14 unit 0 family inet address 100.14.1.4/24
set interfaces ae14 unit 0 family iso
set interfaces ae14 unit 0 family mpls
set interfaces cbp0 unit 0 family bridge interface-mode trunk
set interfaces cbp0 unit 0 family bridge bridge-domain-type bvlan
set interfaces cbp0 unit 0 family bridge isid-list all
set interfaces cbp0 unit 1 family bridge interface-mode trunk
set interfaces cbp0 unit 1 family bridge bridge-domain-type bvlan
set interfaces cbp0 unit 1 family bridge isid-list all
set interfaces lo0 unit 0 family inet address 127.0.0.4/8 primary
set interfaces pip0 unit 0 family bridge interface-mode trunk
set interfaces pip0 unit 0 family bridge bridge-domain-type svlan
set interfaces pip0 unit 0 family bridge isid-list all-service-groups
set interfaces pip0 unit 1 family bridge interface-mode trunk
set interfaces pip0 unit 1 family bridge bridge-domain-type svlan
set interfaces pip0 unit 1 family bridge isid-list all-service-groups
set routing-options router-id 127.0.0.4
set routing-options autonomous-system 65221
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path pe4tope1 from 127.0.0.4
set protocols mpls label-switched-path pe4tope1 to 127.0.0.1
set protocols mpls label-switched-path pe4tope1 primary direct_to_pe1
set protocols mpls label-switched-path pe4tope2 from 127.0.0.4
set protocols mpls label-switched-path pe4tope2 to 127.0.0.2
set protocols mpls label-switched-path pe4tope2 primary direct_to_pe2
set protocols mpls label-switched-path pe4tope3 from 127.0.0.4
set protocols mpls label-switched-path pe4tope3 to 127.0.0.3
set protocols mpls label-switched-path pe4tope3 primary direct_to_pe3
set protocols mpls path direct_to_pe1 172.18.0.1 strict
set protocols mpls path direct_to_pe2 172.20.0.1 strict
set protocols mpls path direct_to_pe3 172.21.0.1 strict
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 127.0.0.4
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 127.0.0.1
set protocols bgp group ibgp neighbor 127.0.0.2
set protocols bgp group ibgp neighbor 127.0.0.3

```

```

set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set routing-instances pbbn1 instance-type virtual-switch
set routing-instances pbbn1 interface cbp0.0
set routing-instances pbbn1 route-distinguisher 127.0.0.4:1
set routing-instances pbbn1 vrf-target target:100:1
set routing-instances pbbn1 protocols evpn pbb-evpn-core
set routing-instances pbbn1 protocols evpn extended-isid-list 1000
set routing-instances pbbn1 bridge-domains bda vlan-id 100
set routing-instances pbbn1 bridge-domains bda isid-list 1000
set routing-instances pbbn1 bridge-domains bda vlan-id-scope-local
set routing-instances pbn1 instance-type virtual-switch
set routing-instances pbn1 interface pip0.0
set routing-instances pbn1 bridge-domains bda domain-type bridge
set routing-instances pbn1 bridge-domains bda vlan-id 10
set routing-instances pbn1 bridge-domains bda interface xe-0/0/0.0
set routing-instances pbn1 bridge-domains bda interface ge-0/2/1.0
set routing-instances pbn1 pbb-options peer-instance pbbn1
set routing-instances pbn1 service-groups sga service-type elan
set routing-instances pbn1 service-groups sga pbb-service-options isid 1000 vlan-id-list 10

```

Configuring Device CE1

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the [CLI User Guide](#).

To configure Device CE1:

1. Set the number of aggregated Ethernet interfaces on Device CE1.

```

[edit chassis]
user@CE1# set aggregated-devices ethernet device-count 50

```


2. Set Device CE1's network services to enhanced Internet Protocol and use enhanced mode capabilities.

```
[edit chassis]
user@CE1# set chassis network-services enhanced-ip
```

3. Configure Device CE1's interfaces.

```
[edit interfaces]
user@CE1# set interfaces xe-0/0/3 flexible-vlan-tagging
user@CE1# set interfaces xe-0/0/3 encapsulation flexible-ethernet-services
user@CE1# set interfaces xe-0/0/3 unit 0 encapsulation vlan-bridge
user@CE1# set interfaces xe-0/0/3 unit 0 vlan-id 10
user@CE1# set interfaces xe-0/0/3 unit 1 encapsulation vlan-bridge
user@CE1# set interfaces xe-0/0/3 unit 1 vlan-id 20
user@CE1# set interfaces xe-0/0/3 unit 2 encapsulation vlan-bridge
user@CE1# set interfaces xe-0/0/3 unit 2 vlan-id 30
user@CE1# set interfaces ge-1/0/2 flexible-vlan-tagging
user@CE1# set interfaces ge-1/0/2 encapsulation flexible-ethernet-services
user@CE1# set interfaces ge-1/0/2 unit 1 encapsulation vlan-bridge
user@CE1# set interfaces ge-1/0/2 unit 1 vlan-id 20
user@CE1# set interfaces ge-1/0/5 flexible-vlan-tagging
user@CE1# set interfaces ge-1/0/5 encapsulation flexible-ethernet-services
user@CE1# set interfaces ge-1/0/5 unit 2 encapsulation vlan-bridge
user@CE1# set interfaces ge-1/0/5 unit 2 vlan-id 30
user@CE1# set interfaces ge-1/0/6 gigether-options 802.3ad ae0
user@CE1# set interfaces ge-1/0/7 gigether-options 802.3ad ae0
user@CE1# set interfaces ge-1/0/8 flexible-vlan-tagging
user@CE1# set interfaces ge-1/0/8 encapsulation flexible-ethernet-services
user@CE1# set interfaces ge-1/0/8 unit 1 encapsulation vlan-bridge
user@CE1# set interfaces ge-1/0/8 unit 1 vlan-id 20
user@CE1# set interfaces ge-1/0/9 flexible-vlan-tagging
user@CE1# set interfaces ge-1/0/9 encapsulation flexible-ethernet-services
user@CE1# set interfaces ge-1/0/9 unit 2 encapsulation vlan-bridge
user@CE1# set interfaces ge-1/0/9 unit 2 vlan-id 30
user@CE1# set interfaces ge-1/1/4 gigether-options 802.3ad ae0
user@CE1# set interfaces ge-1/1/5 gigether-options 802.3ad ae0
user@CE1# set interfaces ge-1/1/6 gigether-options 802.3ad ae0
user@CE1# set interfaces ge-1/1/9 gigether-options 802.3ad ae0
```

4. Configure the aggregated Ethernet bundle on Device CE1.

```
[edit interfaces]
user@CE1# set interfaces ae0 flexible-vlan-tagging
user@CE1# set interfaces ae0 encapsulation flexible-ethernet-services
user@CE1# set interfaces ae0 aggregated-ether-options minimum-links 1
user@CE1# set interfaces ae0 unit 0 encapsulation vlan-bridge
user@CE1# set interfaces ae0 unit 0 vlan-id 10
```

5. Configure the bridge domains on Device CE1.

```
[edit bridge-domains]
user@CE1# set bd10 domain-type bridge
user@CE1# set bd10 vlan-id 10
user@CE1# set bd10 interface ae0.0
user@CE1# set bd10 interface xe-0/0/3.0
user@CE1# set bd20 domain-type bridge
user@CE1# set bd20 vlan-id 20
user@CE1# set bd20 interface xe-0/0/3.1
user@CE1# set bd20 interface ge-1/0/8.1
user@CE1# set bd20 interface ge-1/0/2.1
user@CE1# set bd30 domain-type bridge
user@CE1# set bd30 vlan-id 30
user@CE1# set bd30 interface xe-0/0/3.2
user@CE1# set bd30 interface ge-1/0/9.2
user@CE1# set bd30 interface ge-1/0/5.2
```

Results

From configuration mode, confirm your configuration by entering the `show chassis`, `show interfaces`, and `show bridge-domains` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@CE1# show chassis
aggregated-devices {
  ethernet {
    device-count 50;
  }
}
```

```

}
network-services enhanced-ip;

```

```

user@CE1# show interfaces
xe-0/0/3 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 0 {
        encapsulation vlan-bridge;
        vlan-id 10;
    }
    unit 1 {
        encapsulation vlan-bridge;
        vlan-id 20;
    }
    unit 2 {
        encapsulation vlan-bridge;
        vlan-id 30;
    }
}
ge-1/0/2 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 1 {
        encapsulation vlan-bridge;
        vlan-id 20;
    }
}
ge-1/0/5 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 2 {
        encapsulation vlan-bridge;
        vlan-id 30;
    }
}
ge-1/0/6 {
    gigaether-options {
        802.3ad ae0;
    }
}

```

```
ge-1/0/7 {
    gigether-options {
        802.3ad ae0;
    }
}
ge-1/0/8 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 1 {
        encapsulation vlan-bridge;
        vlan-id 20;
    }
}
ge-1/0/9 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 2 {
        encapsulation vlan-bridge;
        vlan-id 30;
    }
}
ge-1/1/4 {
    gigether-options {
        802.3ad ae0;
    }
}
ge-1/1/5 {
    gigether-options {
        802.3ad ae0;
    }
}
ge-1/1/6 {
    gigether-options {
        802.3ad ae0;
    }
}
ge-1/1/9 {
    gigether-options {
        802.3ad ae0;
    }
}
ae0 {
    flexible-vlan-tagging;
```

```

encapsulation flexible-ethernet-services;
aggregated-ether-options {
    minimum-links 1;
}
unit 0 {
    encapsulation vlan-bridge;
    vlan-id 10;
}
}

```

```

user@CE1# show bridge-domains

```

```

bd10 {
    domain-type bridge;
    vlan-id 10;
    interface ae0.0;
    interface xe-0/0/3.0;
}
bd20 {
    domain-type bridge;
    vlan-id 20;
    interface xe-0/0/3.1;
    interface ge-1/0/8.1;
    interface ge-1/0/2.1;
}
bd30 {
    domain-type bridge;
    vlan-id 30;
    interface xe-0/0/3.2;
    interface ge-1/0/9.2;
    interface ge-1/0/5.2;
}

```

If you are done configuring the device, enter `commit` from configuration mode.

Configuring Device PE1

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the [CLI User Guide](#).

To configure Device PE1:

1. Set the number of aggregated Ethernet interfaces on Device PE1.

```
[edit chassis]
user@PE1# set aggregated-devices ethernet device-count 50
```

2. Set Device PE1's network services to enhanced Internet Protocol and use enhanced mode capabilities.

```
[edit chassis]
user@PE1# set network-services enhanced-ip
```

3. Configure the CE-facing interfaces of Device PE1.

```
[edit interfaces]
user@PE1# set ge-2/0/6 gigether-options 802.3ad ae0
user@PE1# set ge-2/0/7 gigether-options 802.3ad ae0
user@PE1# set ge-2/0/8 flexible-vlan-tagging
user@PE1# set ge-2/0/8 encapsulation flexible-ethernet-services
user@PE1# set ge-2/0/8 unit 0 encapsulation vlan-bridge
user@PE1# set ge-2/0/8 unit 0 vlan-id 20
user@PE1# set ge-2/0/9 flexible-vlan-tagging
user@PE1# set ge-2/0/9 encapsulation flexible-ethernet-services
user@PE1# set ge-2/0/9 unit 0 encapsulation vlan-bridge
user@PE1# set ge-2/0/9 unit 0 vlan-id 30
```

4. Configure the aggregate Ethernet bundle on Device PE1 that connects to Device CE1.

```
[edit interfaces]
user@PE1# set ae0 flexible-vlan-tagging
user@PE1# set ae0 encapsulation flexible-ethernet-services
user@PE1# set ae0 unit 0 encapsulation vlan-bridge
user@PE1# set ae0 unit 0 vlan-id 10
```

5. Configure the EVPN multihoming parameters for the CE-facing interfaces and aggregate Ethernet bundle that connect to the multihomed customer site.

```
[edit interfaces]
user@PE1# set ge-2/0/8 esi 00:22:22:22:22:22:22:22:22
user@PE1# set ge-2/0/8 esi single-active
user@PE1# set ge-2/0/8 esi source-bmac 00:22:22:22:22:22
user@PE1# set ge-2/0/9 esi 00:33:33:33:33:33:33:33:33
user@PE1# set ge-2/0/9 esi single-active
user@PE1# set ge-2/0/9 esi source-bmac 00:33:33:33:33:33
user@PE1# set ae0 esi 00:11:11:11:11:11:11:11:11
user@PE1# set ae0 esi single-active
user@PE1# set ae0 esi source-bmac 00:11:11:11:11:11
```

NOTE: In this example, the EVPN multihoming is operating in the active/standby mode. To configure active/active EVPN multihoming, include the active-active statement at the [edit interfaces *interface-name* esi] hierarchy level instead of the single-active statement.

6. Configure the interface of Device PE1 that connects to Devices PE2, PE3, and PE4.

```
[edit interfaces]
user@PE1# set ge-2/0/2 gether-options 802.3ad ae12
user@PE1# set ge-2/0/3 gether-options 802.3ad ae12
user@PE1# set xe-1/2/3 gether-options 802.3ad ae13
user@PE1# set xe-1/3/0 gether-options 802.3ad ae13
user@PE1# set ge-2/1/0 gether-options 802.3ad ae14
user@PE1# set ge-2/1/5 gether-options 802.3ad ae14
```

7. Configure the aggregate Ethernet bundle on Device PE1 that connect to Devices PE2, PE3, and PE4.

```
[edit interfaces]
user@PE1# set ae12 flexible-vlan-tagging
user@PE1# set ae12 encapsulation flexible-ethernet-services
user@PE1# set ae12 aggregated-ether-options minimum-links 1
user@PE1# set ae12 unit 0 vlan-id 1200
user@PE1# set ae12 unit 0 family inet address 172.16.0.1/16
user@PE1# set ae12 unit 0 family iso
```

```

user@PE1# set ae12 unit 0 family mpls
user@PE1# set ae13 flexible-vlan-tagging
user@PE1# set ae13 encapsulation flexible-ethernet-services
user@PE1# set ae13 aggregated-ether-options minimum-links 1
user@PE1# set ae13 unit 0 vlan-tags outer 1300
user@PE1# set ae13 unit 0 vlan-tags inner 13
user@PE1# set ae13 unit 0 family inet address 172.17.0.1/16
user@PE1# set ae13 unit 0 family iso
user@PE1# set ae13 unit 0 family mpls
user@PE1# set ae14 aggregated-ether-options minimum-links 1
user@PE1# set ae14 unit 0 family inet address 172.18.0.1/16
user@PE1# set ae14 unit 0 family iso
user@PE1# set ae14 unit 0 family mpls

```

8. Configure the interface of Device PE1 that connects to the host.

```

[edit interfaces]
user@PE1# set xe-1/0/1 flexible-vlan-tagging
user@PE1# set xe-1/0/1 encapsulation flexible-ethernet-services
user@PE1# set xe-1/0/1 unit 0 encapsulation vlan-bridge
user@PE1# set xe-1/0/1 unit 0 vlan-id 10

```

9. Configure the loopback interface of Device PE1.

```

[edit interfaces]
user@PE1# set lo0 unit 0 family inet address 127.0.0.1/8 primary
user@PE1# set lo0 unit 0 family iso

```

10. Configure the customer backbone port (CBP) interfaces on Device PE1.

```

[edit interfaces]
user@PE1# set cbp0 unit 0 family bridge interface-mode trunk
user@PE1# set cbp0 unit 0 family bridge bridge-domain-type bvlan
user@PE1# set cbp0 unit 0 family bridge isid-list all
user@PE1# set cbp0 unit 1 family bridge interface-mode trunk
user@PE1# set cbp0 unit 1 family bridge bridge-domain-type bvlan
user@PE1# set cbp0 unit 1 family bridge isid-list all

```


11. Configure the Provider Instance Port (PIP) on Device PE1.

```
[edit interfaces]
user@PE1# set pip0 unit 0 family bridge interface-mode trunk
user@PE1# set pip0 unit 0 family bridge bridge-domain-type svlan
user@PE1# set pip0 unit 0 family bridge isid-list all-service-groups
user@PE1# set pip0 unit 1 family bridge interface-mode trunk
user@PE1# set pip0 unit 1 family bridge bridge-domain-type svlan
user@PE1# set pip0 unit 1 family bridge isid-list all-service-groups
```

12. Configure the router ID and autonomous system number for Device PE1.

```
[edit routing-options]
user@PE1# set router-id 127.0.0.1
user@PE1# set autonomous-system 65221
```

13. Configure RSVP on all the interfaces of Device PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set rsvp interface all
user@PE1# set rsvp interface fxp0.0 disable
```

14. Configure MPLS on all the interfaces of Device PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set mpls interface all
user@PE1# set mpls interface fxp0.0 disable
```

15. Configure LSPs from Device PE1 to all other PE devices.

```
[edit protocols]
user@PE1# set mpls label-switched-path pe1tope2 from 127.0.0.1
user@PE1# set mpls label-switched-path pe1tope2 to 127.0.0.2
user@PE1# set mpls label-switched-path pe1tope2 primary direct_to_pe2
user@PE1# set mpls label-switched-path pe1tope3 from 127.0.0.1
user@PE1# set mpls label-switched-path pe1tope3 to 127.0.0.3
user@PE1# set mpls label-switched-path pe1tope3 primary direct_to_pe3
user@PE1# set mpls label-switched-path pe1tope4 from 127.0.0.1
```

```

user@PE1# set mpls label-switched-path pe1tope4 to 127.0.0.4
user@PE1# set mpls label-switched-path pe1tope4 primary direct_to_pe4

```

16. Configure MPLS paths from Device PE1 to all other PE devices.

```

[edit protocols]
user@PE1# set mpls path direct_to_pe2 100.12.1.2 strict
user@PE1# set mpls path direct_to_pe3 100.13.1.3 strict
user@PE1# set mpls path direct_to_pe4 100.14.1.4 strict

```

17. Configure an internal BGP session under family EVPN from Device PE1 to all other PE devices.

```

[edit protocols]
user@PE1# set bgp group ibgp type internal
user@PE1# set bgp group ibgp local-address 127.0.0.1
user@PE1# set bgp group ibgp family evpn signaling
user@PE1# set bgp group ibgp neighbor 127.0.0.2
user@PE1# set bgp group ibgp neighbor 127.0.0.3
user@PE1# set bgp group ibgp neighbor 127.0.0.4

```

18. Configure OSPF on all the interfaces of Device of PE1, excluding the management interface.

```

[edit protocols]
user@PE1# set ospf traffic-engineering
user@PE1# set ospf area 0.0.0.0 interface all
user@PE1# set ospf area 0.0.0.0 interface fxp0.0 disable

```

19. Configure a customer routing instance (I-component) on Device PE1 with type virtual switch. Assign the CBP interface, route-distinguisher, and virtual routing and forwarding (VRF) target values to the PBBN routing instance.

```

[edit routing-instances]
user@PE1# set pbbn1 instance-type virtual-switch
user@PE1# set pbbn1 interface cbp0.0
user@PE1# set pbbn1 route-distinguisher 127.0.0.1:1
user@PE1# set pbbn1 vrf-target target:100:1

```

20. Configure PBB-EVPN integration from the customer routing instance. Assign the extended I-SID list and bridge domains to the routing instance.

```
[edit routing-instances]
user@PE1# set pbbn1 protocols evpn pbb-evpn-core
user@PE1# set pbbn1 protocols evpn extended-isid-list 1000
user@PE1# set pbbn1 bridge-domains bda vlan-id 100
user@PE1# set pbbn1 bridge-domains bda isid-list 1000
user@PE1# set pbbn1 bridge-domains bda vlan-id-scope-local
```

21. Configure a provider routing instance on Device PE1 with type virtual switch. Assign the PBP interface and bridge domains to the routing instance.

```
[edit routing-instances]
user@PE1# set pbn1 instance-type virtual-switch
user@PE1# set pbn1 interface pip0.0
user@PE1# set pbn1 bridge-domains bda domain-type bridge
user@PE1# set pbn1 bridge-domains bda vlan-id 10
user@PE1# set pbn1 bridge-domains bda interface ae0.0
user@PE1# set pbn1 bridge-domains bda interface xe-1/0/1.0
user@PE1# set pbn1 bridge-domains bda interface ge-2/0/9.0
user@PE1# set pbn1 bridge-domains bda interface ge-2/0/8.0
```

22. Configure the peer PBBN routing instance in the customer routing instance.

```
[edit routing-instances]
user@PE1# set pbn1 pbb-options peer-instance pbbn1
```

23. Configure the service groups to be supported in the customer routing instance.

```
[edit routing-instances]
user@PE1# set pbn1 service-groups sga service-type elan
user@PE1# set pbn1 service-groups sga pbb-service-options isid 1000 vlan-id-list 10
```

- 24.

Results

From configuration mode, confirm your configuration by entering the `show chassis`, `show interfaces`, `routing-options`, `show protocols` and `show routing-instances` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show chassis
aggregated-devices {
  ethernet {
    device-count 50;
  }
}
network-services enhanced-ip;
```

```
user@PE1# show interfaces
xe-1/0/1 {
  flexible-vlan-tagging;
  encapsulation flexible-ethernet-services;
  unit 0 {
    encapsulation vlan-bridge;
    vlan-id 10;
  }
}
xe-1/2/3 {
  gigether-options {
    802.3ad ae13;
  }
}
xe-1/3/0 {
  gigether-options {
    802.3ad ae13;
  }
}
ge-2/0/2 {
  gigether-options {
    802.3ad ae12;
  }
}
ge-2/0/3 {
  gigether-options {
    802.3ad ae12;
```

```

    }
}
ge-2/0/6 {
    gigether-options {
        802.3ad ae0;
    }
}
ge-2/0/7 {
    gigether-options {
        802.3ad ae0;
    }
}
ge-2/0/8 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    esi {
        00:22:22:22:22:22:22:22:22;
        single-active;
        source-bmac 00:22:22:22:22:22;
    }
    unit 0 {
        encapsulation vlan-bridge;
        vlan-id 20;
    }
}
ge-2/0/9 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    esi {
        00:33:33:33:33:33:33:33:33;
        single-active;
        source-bmac 00:33:33:33:33:33;
    }
    unit 0 {
        encapsulation vlan-bridge;
        vlan-id 30;
    }
}
ge-2/1/0 {
    gigether-options {
        802.3ad ae14;
    }
}

```

```

ge-2/1/5 {
    gigeother-options {
        802.3ad ae14;
    }
}
ae0 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    esi {
        00:11:11:11:11:11:11:11:11:11;
        single-active;
        source-bmac 00:11:11:11:11:11;
    }
    unit 0 {
        encapsulation vlan-bridge;
        vlan-id 10;
    }
}
ae12 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    aggregated-ether-options {
        minimum-links 1;
    }
    unit 0 {
        vlan-id 1200;
        family inet {
            address 172.16.0.1/16;
        }
        family iso;
        family mpls;
    }
}
ae13 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    aggregated-ether-options {
        minimum-links 1;
    }
    unit 0 {
        vlan-tags outer 1300 inner 13;
        family inet {
            address 172.17.0.1/16;
        }
    }
}

```

```

    }
    family iso;
    family mpls;
}
}
ae14 {
    aggregated-ether-options {
        minimum-links 1;
    }
    unit 0 {
        family inet {
            address 172.18.0.1/16;
        }
        family iso;
        family mpls;
    }
}
cbp0 {
    unit 0 {
        family bridge {
            interface-mode trunk;
            bridge-domain-type bvlan;
            isid-list all;
        }
    }
    unit 1 {
        family bridge {
            interface-mode trunk;
            bridge-domain-type bvlan;
            isid-list all;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 127.0.0.1/8 {
                primary;
            }
        }
        family iso;
    }
}
}

```

```

pip0 {
    unit 0 {
        family bridge {
            interface-mode trunk;
            bridge-domain-type svlan;
            isid-list all-service-groups;
        }
    }
    unit 1 {
        family bridge {
            interface-mode trunk;
            bridge-domain-type svlan;
            isid-list all-service-groups;
        }
    }
}

```

```

user@PE1# show routing-options
router-id 127.0.0.1;
autonomous-system 65221;

```

```

user@PE1# show protocols
rsvp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
mpls {
    label-switched-path pe1tope2 {
        from 127.0.0.1;
        to 127.0.0.2;
        primary direct_to_pe2;
    }
    label-switched-path pe1tope3 {
        from 127.0.0.1;
        to 127.0.0.3;
        primary direct_to_pe3;
    }
    label-switched-path pe1tope4 {

```



```

        from 127.0.0.1;
        to 127.0.0.4;
        primary direct_to_pe4;
    }
    path direct_to_pe2 {
        100.12.1.2 strict;
    }
    path direct_to_pe3 {
        100.13.1.3 strict;
    }
    path direct_to_pe4 {
        100.14.1.4 strict;
    }
    interface all;
    interface fxp0.0 {
        disable;
    }
}
bgp {
    group ibgp {
        type internal;
        local-address 127.0.0.1;
        family evpn {
            signaling;
        }
        neighbor 127.0.0.2;
        neighbor 127.0.0.3;
        neighbor 127.0.0.4;
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
}

```

```

    }
}

```

```
user@PE1# show routing-instances
```

```

pbbn1 {
    instance-type virtual-switch;
    interface cbp0.0;
    route-distinguisher 127.0.0.1:1;
    vrf-target target:100:1;
    protocols {
        evpn {
            pbb-evpn-core;
            extended-isid-list 1000;
        }
    }
    bridge-domains {
        bda {
            vlan-id 100;
            isid-list 1000;
            vlan-id-scope-local;
        }
    }
}

pbn1 {
    instance-type virtual-switch;
    interface pip0.0;
    bridge-domains {
        bda {
            domain-type bridge;
            vlan-id 10;
            interface ae0.0;
            interface xe-1/0/1.0;
            interface ge-2/0/9.0;
            interface ge-2/0/8.0;
        }
    }
    pbb-options {
        peer-instance pbbn1;
    }
    service-groups {
        sga {

```

```

        service-type elan;
        pbb-service-options {
            isid 1000 vlan-id-list 10;
        }
    }
}

```

If you are done configuring the device, enter `commit` from configuration mode.

Verification

IN THIS SECTION

- [Verifying BGP Peering Status | 1512](#)
- [Verify MAC Table Entries | 1513](#)
- [Verifying the EVPN Database | 1514](#)
- [Verifying EVPN Routing Instances | 1514](#)

Confirm that the configuration is working properly.

Verifying BGP Peering Status

Purpose

Verify that the BGP session is established between the PE devices.

Action

From operational mode, run the `show bgp summary` command.

```

user@PE1> show bgp summary
Groups: 1 Peers: 3 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State   Pending
bgp.evpn.0
              13         13         0           0         0         0
Peer          AS      InPkt   OutPkt   OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...

```

```

127.0.0.2      65221      10      8      0      0      42 Establ
  bgp.evpn.0: 6/6/6/0
  pbbn1.evpn.0: 3/3/3/0
  __default_evpn__.evpn.0: 3/3/3/0
127.0.0.3      65221      8      8      0      0      40 Establ
  bgp.evpn.0: 4/4/4/0
  pbbn1.evpn.0: 3/3/3/0
  __default_evpn__.evpn.0: 1/1/1/0
127.0.0.4      65221      9      11     0      0      1:04 Establ
  bgp.evpn.0: 3/3/3/0
  pbbn1.evpn.0: 3/3/3/0
  __default_evpn__.evpn.0: 0/0/0/0

```

Meaning

A BGP session is established between the PE devices.

Verify MAC Table Entries

Purpose

Verify the number of rbeb interfaces learned in the MAC table on all PEs.

Action

From operational mode, run the `show bgp summary` command.

```

user@PE1> show bridge mac-table count instance pbn1 bridge-domain bda
10 MAC address learned in routing instance pbn1 bridge domain bda

```

MAC address count per interface within routing instance:

Logical interface	MAC count
ae0.0:10	0
ge-2/0/8.0:10	10
ge-2/0/9.0:10	0
rbeb.32772	0
rbeb.32771	0
xe-1/0/0.0:10	0

MAC address count per learn VLAN within routing instance:

Learn VLAN ID	MAC count
10	10

Meaning

For VLAN ID 10, 10 MAC addresses have been learned in the MAC table of Device PE1.

Verifying the EVPN Database

Purpose

Verify the EVPN database information on Device PE4.

Action

From operational mode, run the `show evpn database` command.

```
user@PE1> show evpn database
Instance: pbbn1
```

VLAN	DomainId	MAC address	Active source	Timestamp	IP address
1000		00:11:11:11:11:11	127.0.0.1	Jan 26 12:09:29	
1000		00:11:11:11:11:12	127.0.0.2	Jan 26 12:09:38	
1000		00:1d:b5:a2:47:b0	127.0.0.1	Jan 26 12:09:10	
1000		00:22:22:22:22:22	127.0.0.1	Jan 26 12:09:29	
1000		00:23:9c:5e:a7:b0	Local	Jan 26 12:08:02	
1000		00:23:9c:f0:f9:b0	127.0.0.3	Jan 26 12:09:30	
1000		00:33:33:33:33:33	127.0.0.1	Jan 26 12:09:29	
1000		00:44:44:44:44:44	127.0.0.3	Jan 26 12:09:34	
1000		00:44:44:44:44:45	Local	Jan 26 12:09:28	
1000		80:71:1f:c1:ed:b0	127.0.0.2	Jan 26 12:09:31	

Verifying EVPN Routing Instances

Purpose

Verify the EVPN routing instance information on all the PE devices.

Action

From operational mode, run the `show evpn routing-instance` command.

```

user@PE1> show evpn routing-instance
Instance: pbbn1
  Route Distinguisher: 127.0.0.1:1
  Per-instance MAC route label: 300080
  Per-instance multicast route label: 300096
PBB EVPN Core enabled
  Control word enabled
  MAC database status
    Local Remote
  MAC advertisements:          4      6
  MAC+IP advertisements:      0      0
  Default gateway MAC advertisements: 0      0
  Number of local interfaces: 4 (4 up)
    Interface name  ESI                               Mode      Status    AC-Role
  ae0.0            00:11:11:11:11:11:11:11:11:11:11 single-active Up        Root
  cbp0.0           00:00:00:00:00:00:00:00:00:00 single-homed Up        Root
  ge-2/0/8.0       00:22:22:22:22:22:22:22:22:22 single-active Up        Root
  ge-2/0/9.0       00:33:33:33:33:33:33:33:33:33 single-active Up        Root
  Number of IRB interfaces: 0 (0 up)
  Number of bridge domains: 1
    VLAN Domain ID  Intfs / up  IRB intf  Mode      MAC sync  IM route label  SG
  sync  IM core nexthop
    1000          0    0          Extended  Enabled    300096
  Disabled
  Number of Bundle bridge domains: 0
  Number of neighbors: 3
    Address      MAC    MAC+IP    AD    IM    ES Leaf-label
  127.0.0.2      2      0         0     1     0
  127.0.0.3      2      0         0     1     0
  127.0.0.4      2      0         0     1     0
  Number of ethernet segments: 3
  ESI: 00:11:11:11:11:11:11:11:11:11
    Status: Resolved by IFL ae0.0
    Local interface: ae0.0, Status: Up/Blocking
    Designated forwarder: 127.0.0.2
    Backup forwarder: 127.0.0.1
    Backup forwarder: 127.0.0.3
    Last designated forwarder update: Jan 26 12:43:42
    Minimum ISID: 1000

```

```

ESI: 00:22:22:22:22:22:22:22:22
  Status: Resolved by IFL ge-2/0/8.0
  Local interface: ge-2/0/8.0, Status: Up/Forwarding
  Designated forwarder: 127.0.0.1
  Backup forwarder: 127.0.0.2
  Last designated forwarder update: Jan 26 12:43:42
  Minimum ISID: 1000
ESI: 00:33:33:33:33:33:33:33:33
  Status: Resolved by IFL ge-2/0/9.0
  Local interface: ge-2/0/9.0, Status: Up/Forwarding
  Designated forwarder: 127.0.0.1
  Backup forwarder: 127.0.0.2
  Last designated forwarder update: Jan 26 12:43:42
  Minimum ISID: 1000

```

```

user@PE2> show evpn routing-instance
Instance: pbbn1
  Route Distinguisher: 127.0.0.2:1
  Per-instance MAC route label: 338721
  Per-instance multicast route label: 338737
  PBB EVPN Core enabled
  Control word enabled
  MAC database status
    Local Remote
  MAC advertisements:          2      8
  MAC+IP advertisements:      0      0
  Default gateway MAC advertisements: 0      0
  Number of local interfaces: 4 (4 up)
    Interface name  ESI                               Mode           Status    AC-Role
  ae0.0            00:11:11:11:11:11:11:11:11 single-active   Up         Root
  cbp0.0           00:00:00:00:00:00:00:00:00 single-homed    Up         Root
  ge-1/3/5.0       00:22:22:22:22:22:22:22:22 single-active   Up         Root
  ge-1/3/6.0       00:33:33:33:33:33:33:33:33 single-active   Up         Root
  Number of IRB interfaces: 0 (0 up)
  Number of bridge domains: 1
    VLAN  Domain ID  Intfs / up  IRB intf  Mode           MAC sync  IM route label  SG
  sync  IM core nexthop
    1000      0      0           Extended   Enabled    338737
  Disabled
  Number of Bundle bridge domains: 0
  Number of neighbors: 3
    Address          MAC    MAC+IP    AD    IM    ES Leaf-label

```

127.0.0.1	4	0	0	1	0
127.0.0.3	2	0	0	1	0
127.0.0.4	2	0	0	1	0

Number of ethernet segments: 3

ESI: 00:11:11:11:11:11:11:11:11

Status: Resolved by IFL ae0.0

Local interface: ae0.0, Status: **Up/Forwarding**

Designated forwarder: 127.0.0.2

Backup forwarder: 127.0.0.1

Backup forwarder: 127.0.0.3

Last designated forwarder update: Jan 26 12:09:37

Minimum ISID: 1000

ESI: 00:22:22:22:22:22:22:22:22

Status: Resolved by IFL ge-1/3/5.0

Local interface: ge-1/3/5.0, Status: **Up/Blocking**

Designated forwarder: 127.0.0.1

Backup forwarder: 127.0.0.2

Last designated forwarder update: Jan 26 12:09:33

Minimum ISID: 1000

ESI: 00:33:33:33:33:33:33:33:33

Status: Resolved by IFL ge-1/3/6.0

Local interface: ge-1/3/6.0, Status: **Up/Blocking**

Designated forwarder: 127.0.0.1

Backup forwarder: 127.0.0.2

Last designated forwarder update: Jan 26 12:09:33

Minimum ISID: 1000

user@PE3> show evpn routing-instance

Instance: pbbn1

Route Distinguisher: 127.0.0.3:1

Per-instance MAC route label: 338833

Per-instance multicast route label: 338849

PBB EVPN Core enabled

Control word enabled

MAC database status

	Local	Remote
MAC advertisements:	2	8
MAC+IP advertisements:	0	0
Default gateway MAC advertisements:	0	0

Number of local interfaces: 3 (3 up)

Interface name	ESI	Mode	Status	AC-Role
ae0.0	00:11:11:11:11:11:11:11:11	single-active	Up	Root


```

    cbp0.0          00:00:00:00:00:00:00:00:00 single-homed Up      Root
    ge-1/0/8.0      00:44:44:44:44:44:44:44 single-active Up      Root
Number of IRB interfaces: 0 (0 up)
Number of bridge domains: 1
  VLAN Domain ID  Intfs / up  IRB intf  Mode          MAC sync  IM route label  SG
sync  IM core nexthop
      1000        0    0              Extended    Enabled    338849
Disabled
Number of Bundle bridge domains: 0
Number of neighbors: 3
  Address          MAC      MAC+IP      AD      IM      ES Leaf-label
  127.0.0.1        4        0          0        1        0
  127.0.0.2        2        0          0        1        0
  127.0.0.4        2        0          0        1        0
Number of ethernet segments: 2
ESI: 00:11:11:11:11:11:11:11:11
  Status: Resolved by IFL ae0.0
  Local interface: ae0.0, Status: Up/Blocking
  Designated forwarder: 127.0.0.2
  Backup forwarder: 127.0.0.1
  Backup forwarder: 127.0.0.3
  Last designated forwarder update: Jan 26 12:09:46
  Minimum ISID: 1000
ESI: 00:44:44:44:44:44:44:44:44
  Status: Resolved by IFL ge-1/0/8.0
  Local interface: ge-1/0/8.0, Status: Up/Forwarding
  Designated forwarder: 127.0.0.3
  Backup forwarder: 127.0.0.4
  Last designated forwarder update: Jan 26 12:09:31
  Minimum ISID: 1000

```

```

user@PE4> show evpn routing-instance
Instance: pbbn1
Route Distinguisher: 127.0.0.4:1
Per-instance MAC route label: 301520
Per-instance multicast route label: 301536
PBB EVPN Core enabled
Control word enabled
MAC database status
MAC advertisements:          Local Remote
MAC+IP advertisements:      0        0

```

```

Default gateway MAC advertisements:      0      0
Number of local interfaces: 2 (2 up)
Interface name  ESI                      Mode      Status    AC-Role
cbp0.0         00:00:00:00:00:00:00:00:00:00 single-homed Up        Root
ge-0/2/1.0     00:44:44:44:44:44:44:44:44:44 single-active Up        Root
Number of IRB interfaces: 0 (0 up)
Number of bridge domains: 1
VLAN  Domain ID  Intfs / up  IRB intf  Mode      MAC sync  IM route label  SG
sync  IM core nexthop
      1000      0    0              Extended  Enabled    301536
Disabled
Number of Bundle bridge domains: 0
Number of neighbors: 3
Address      MAC    MAC+IP      AD      IM      ES Leaf-label
127.0.0.1    4      0          0      1      0
127.0.0.2    2      0          0      1      0
127.0.0.3    2      0          0      1      0
Number of ethernet segments: 1
ESI: 00:44:44:44:44:44:44:44:44
Status: Resolved by IFL ge-0/2/1.0
Local interface: ge-0/2/1.0, Status: Up/Blocking
Designated forwarder: 127.0.0.3
Backup forwarder: 127.0.0.4
Last designated forwarder update: Jan 26 12:43:52
Minimum ISID: 1000

```

Meaning

The command output displays PBB-EVPN integration, where EVPN is in the active/standby multihoming mode. If EVPN multihoming was configured in the active/active mode, the status of all the ESIs would be Up/Forwarding. In the active/standby multihoming mode, one ESI is in the Up/Forwarding state and all other ESIs remain in the Up/Blocking state.

RELATED DOCUMENTATION

[Provider Backbone Bridging \(PBB\) and EVPN Integration Overview | 1414](#)

[Example: Configuring PBB with Single-Homed EVPN | 1450](#)

[pbb-evpn-core | 1640](#)

Configuring MAC Pinning for PBB-EVPNs

IN THIS CHAPTER

- [PBB-EVPN MAC Pinning Overview | 1520](#)
- [Configuring PBB-EVPN MAC Pinning | 1521](#)

PBB-EVPN MAC Pinning Overview

Starting in Junos OS Release 17.2, the MAC pinning feature is enabled on provider backbone bridging (PBB) and Ethernet VPN (EVPN) integration, including customer edge (CE) interfaces and EVPN over PBB core in both all-active or single-active mode.

The MAC pinning feature is used to avoid loops in a network and is also used for MAC security restriction by avoiding MAC move on duplicate MAC detection. When MAC pinning is enabled, the dynamically learned MAC addresses are not allowed to move to any other interface in a bridge domain until it is aged out and traffic received with the same source MAC address on other bridge interfaces are discarded. This feature is an advantage over blocking of the complete interface on duplicate MAC detection or loop, as MAC pinning works at the MAC label. This feature is local to a provider edge (PE) device and does not require any interoperability.

PBB has I-component and B-Component, where I-component (customer routing instance) is responsible for mapping the CE port traffic to the instance source ID (I-SID), and the B-component learns and forwards traffic on the backbone port. Traffic received from the MPLS core or from the PBB port is classified and based on the I-SID and PBB MAC, and gets mapped to the correct I-component. Remote customer MAC addresses are learned over remote backbone edge port (BEB) interface in the I-component bridge domain. This interface is created dynamically on PBB neighbor detection. MAC addresses learned over the remote BEB interface in I-component are pinned when MAC pinning is enabled for PBB-EVPN.

To configure MAC pinning for PBB-EVPN, include the `mac-pinning` statement at the `[edit routing-instances pbbn protocols evpn]`, where `pbbn` is the PBB routing instance over backbone port (B-component). With this configuration, the dynamically learned MAC addresses in the PBB I-component bridge domain over CE interfaces, as well as PBB-MPLS core interfaces are pinned.

When configuring the PBB-EVPN MAC pinning feature, take the following into consideration:

- PBB-EVPN MAC pinning is supported on MX Series routers with MPC and MIC interfaces only.
- PBB-EVPN MAC pinning is supported on Ethernet Layer 2 bridge interfaces only.
- When there is a MAC move between the I-component and an access interface, the MAC address is learned locally over the PBB-EVPN MPLS core over a remote BEB interface in the I-component bridge domain. The MAC moves between the CE or core interfaces for this MAC is not allowed.
- In MAC pinning for PBB with EVPN active-active and single-active multihoming, MAC pinning must be enabled or disabled on all the multihomed PE devices in the broadcast domain. This is because MAC pinning at a multihomed PE device is local to the PE, and it is possible that a MAC address that is pinned towards a multihomed CE device and PE device is also pinned toward a single-homed customer site or toward any other Ethernet segment identifier (ESI) at another multihomed PE device.
- A next hop bridge domain is created in PBB-EVPN I-component bridge domain toward the B-component when there is an unresolved source MAC notification when the first remote MAC address is received. As a result, the first MAC address learned over PBB back bone core interface can be delayed on pinning, and may result moving to other single-homed or ESI interface if the same MAC traffic is received.
- Static MAC addresses are given preference over dynamic pin MACs.
- MAC pinning is enabled for all neighbors of a PBB routing instance and cannot be enabled for a specific neighbor.
- PBB-EVPN MAC pin discard notification is not generated for a remote BEB interface when traffic is discarded due to MAC pinning until a MAC is learned locally over the remote BEB interface.

RELATED DOCUMENTATION

Understanding MAC Pinning

[Configuring PBB-EVPN MAC Pinning | 1521](#)

mac-pinning

[pbb-evpn-core | 1640](#)

Configuring PBB-EVPN MAC Pinning

Starting in Junos OS Release 17.2, the MAC pinning feature is enabled on provider backbone bridging (PBB) and Ethernet VPN (EVPN) integration, including customer edge (CE) interfaces and EVPN over PBB core in both all-active or single-active mode.

When MAC pinning is enabled, the dynamically learned MAC addresses are not allowed to move to any other interface in a bridge domain until it is aged out and traffic received with the same source MAC address on other bridge interfaces are discarded. This feature is an advantage over blocking of the complete interface on duplicate MAC detection or loop detection, as MAC pinning works at the MAC label. This feature is local to a provider edge (PE) device and does not require any interoperability.

Before you begin:

- Configure the device interfaces, including the customer backbone port (CBP) interface, the provider instance port (PIP) interfaces, and the loopback interface. Assign the bridge family to the interfaces.
- Assign the router ID and autonomous system ID to the device.
- Configure an internal BGP group with EVPN signaling.
- Enable the following protocols on the device:
 - MPLS
 - LDP
 - OSPF

To enable MAC pinning on PBB-EVPN:

1. Configure the B-component routing instance.

Assign the virtual switch instance type and the CBP interface to it. Configure other routing instance attributes like route distinguisher and virtual routing and forwarding (VRF) target to the routing instance.

NOTE: Configure B-component routing instances for other CBP interface units on the device, and assign different VLAN IDs and I-SID lists for the different interface units.

```
[edit routing-instances]
user@R1# set pbbn instance-type virtual-switch
user@R1# set pbbn interface cbp-interface
user@R1# set pbbn route-distinguisher route-distinguisher-value
user@R1# set pbbn vrf-target vrf-target
```

2. Enable PBB- EVPN integration for the B-component routing instance.

```
[edit routing-instances]
user@R1# set pbbn protocols evpn pbb-evpn-core
```

3. Enable MAC pinning for the B-component routing instance.

```
[edit routing-instances]
user@R1# set pbbn protocols evpn mac-pinning
```

4. Assign instance source IDs (I-SID) list to the B-component routing instance.

```
[edit routing-instances]
user@R1# set pbbn protocols evpn extended-isid-list extended-isid-list
```

5. Configure a bridge domain for the B-component routing instance and assign a VLAN and and I-SID list to the bridge domain.

```
[edit routing-instances]
user@R1# set pbbn bridge-domains bridge-domain vlan-id vlan-id
user@R1# set pbbn bridge-domains bridge-domain isid-list isid-list
```

6. Configure the I-component routing instance.
Assign the virtual switch instance type and the PIP interface to it.

NOTE: Configure I-component routing instances for other PIP interface units on the device, and assign different bridge domains, VLAN IDs and I-SID lists for the different interface units.

```
[edit routing-instances]
user@R1# set pbn instance-type virtual-switch
user@R1# set pbn interface pip-interface
```

7. Configure bridge domain for the I-component routing instance and assign interfaces and VLANs to the bridge domain.

```
[edit routing-instances]
user@R1# set pbn bridge-domains bridge-domain domain-type bridge
user@R1# set pbn bridge-domains bridge-domain vlan-id vlan-id
user@R1# set pbn bridge-domains bridge-domain interface interface-name
```

8. Enable MAC pinning for the interface in the I-component routing instance.

```
[edit routing-instances]
user@R1# set pbn bridge-domains bridge-domain bridge-options interface interface-name mac-
pinning
```

9. Configure peering between the B-component and the I-component routing instances.

```
[edit routing-instances]
user@R1# set pbn bridge-domains bridge-domain pbb-options peer-instance pbbn
```

10. Configure PBB service group and assign I-SID and VLAN ID list.

```
[edit routing-instances]
user@R1# set pbn service-groups service-group pbb-service-options isid isid vlan-id-list
valn-id-list
```

RELATED DOCUMENTATION

Understanding MAC Pinning

[PBB-EVPN MAC Pinning Overview | 1520](#)

mac-pinning

[pbb-evpn-core | 1640](#)

8

PART

EVPN Standards

EVPN Standards | 1526

EVPN Standards

IN THIS CHAPTER

- [Supported EVPN Standards | 1526](#)

Supported EVPN Standards

RFCs and Internet drafts that define standards for EVPNs:

- RFC 4364, *BGP/MPLS IP Virtual Private Networks (VPNs)*
- RFC 4761, *Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling*
- RFC 7209, *Requirements for Ethernet VPN (EVPN)*
- RFC 7432, *BGP MPLS-Based Ethernet VPN*

The following features are not supported:

- Automatic derivation of Ethernet segment (ES) values. Only static ES configurations are supported.
- Host proxy ARP.
- RFC 7623, *Provider Backbone Bridging Combined with Ethernet VPN (PBB-EVPN)*
- RFC 8214, *Virtual Private Wire Service Support in Ethernet VPN*
- RFC 8317, *Ethernet-Tree (E-Tree) Support in Ethernet VPN (EVPN) and Provider Backbone Bridging EVPN (PBB-EVPN)*
- RFC 8365, *A Network Virtualization Overlay Solution Using Ethernet VPN (EVPN)*
- Internet draft draft-ietf-bess-evpn-igmp-ml-d-proxy, *IGMP and MLD Proxy for EVPN*
- Internet draft draft-ietf-bess-evpn-inter-subnet-forwarding, *Integrated Routing and Bridging in EVPN*

- Internet draft draft-ietf-bess-evpn-na-flags, *Propagation of IPv6 Neighbor Advertisement Flags in EVPN*
- Internet draft draft-ietf-bess-evpn-oam-req-frmwk, *EVPN Operations, Administration and Maintenance Requirements and Framework*
- Internet draft draft-ietf-bess-evpn-optimized-ir, *Optimized Ingress Replication solution for EVPN*
- Internet draft draft-ietf-bess-evpn-pref-df, *Preference-based EVPN DF Election*
- Internet draft draft-ietf-bess-evpn-prefix-advertisement, *IP Prefix Advertisement in EVPN*
- Internet draft draft-ietf-bess-evpn-proxy-arp-nd, *Operational Aspects of Proxy-ARP/ND in EVPN Networks*
- Internet draft draft-ietf-bess-evpn-virtual-eth-segment, *EVPN Virtual Ethernet Segment*
- Internet draft draft-ietf-bess-evpn-vpls-seamless-integ, *(PBB-)EVPN Seamless Integration with (PBB-)VPLS*
- Internet draft draft-ietf-bess-evpn-vpws-fxc, *EVPN VPWS Flexible Cross-Connect Service*
- Internet draft draft-ietf-bess-evpn-yang, *Yang Data Model for EVPN*
- Internet draft draft-ietf-isis-segment-routing-extensions-13, *IS-IS Extensions for Segment Routing*
- Internet draft draft-ietf-spring-segment-routing-13, *Segment Routing Architecture*
- Internet draft draft-ietf-spring-segment-routing-mpls-11, *Segment Routing with MPLS data plane*
- Internet draft draft-wsv-bess-extended-evpn-optimized-ir, *Extended Procedures for EVPN Optimized Ingress Replication*

RELATED DOCUMENTATION

[EVPN Overview](#) | 874

[Accessing Standards Documents on the Internet](#)

9

PART

VXLAN-Only Features

[Flexible VXLAN Tunnels](#) | 1529

[Static VXLAN](#) | 1536

Flexible VXLAN Tunnels

IN THIS CHAPTER

- [Understanding Programmable Flexible VXLAN Tunnels | 1529](#)

Understanding Programmable Flexible VXLAN Tunnels

IN THIS SECTION

- [Benefits of Programmable Flexible Tunnels | 1530](#)
- [Programmable Flexible Tunnels, Flexible Routes, and Tunnel Profiles | 1531](#)
- [Routes and Routing Tables | 1532](#)
- [Flexible Tunnel Traffic Flows | 1533](#)
- [Preparing the Gateway Devices | 1533](#)
- [Understanding Flexible Tunnel Behavior | 1534](#)
- [Flexible Tunnel Limitations | 1534](#)

Starting in Junos OS Release 19.1R1, we support flexible tunnels in the following data center environment:

- Implements one or more controllers.
- Uses Virtual Extensible LAN (VXLAN) as the overlay encapsulation protocol.

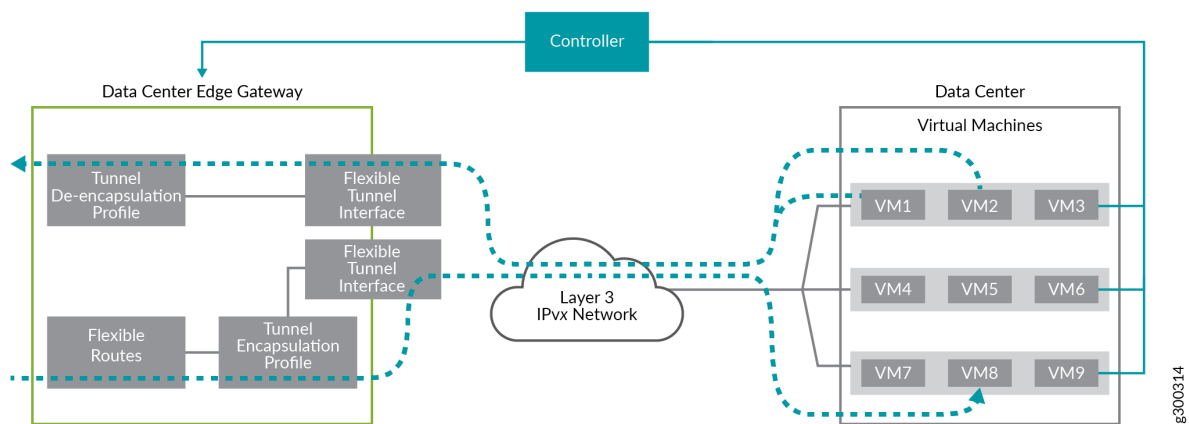
NOTE: The flexible tunnel feature supports IPv4 and IPv6 VXLAN encapsulation and de-encapsulation.

- Controllers in your data center environment and the JET APIs enable you to program a large number of flexible tunnels into the gateway devices. This new method enables edge gateways to communicate with a large number of hosts in a data center.
- The tunnel encapsulation and de-encapsulation profiles for a particular flexible tunnel are typically separate and do not need to be symmetrical. A static route for a particular data center host is mapped to an encapsulation profile, and traffic destined for that host is encapsulated in a distinct tunnel. However, de-encapsulation is not associated with a particular static route. As a result, traffic from one or more data center hosts that match attributes in a de-encapsulation profile can be aggregated into a single tunnel. This mechanism results in fewer de-encapsulation profiles and more efficient use of the existing profiles and flexible tunnels.

Programmable Flexible Tunnels, Flexible Routes, and Tunnel Profiles

Figure 161 on page 1531 shows a sample topology in which we support flexible tunnels. In this topology, a data center edge gateway interfaces with virtual machines (VMs) in a data center in which VXLAN is used. When tunneling Layer 3 traffic over the intervening IPv4 or IPv6 network, the gateway device and the hypervisors or other specialized devices that manage the VMs also function as virtual tunnel endpoints (VTEPs). The VTEPs encapsulate the packets with a VXLAN header, remove, or de-encapsulate, the header from the packets, then forward the packets.

Figure 161: Flexible Tunnel Components and Traffic Flows



To support a flexible tunnel, the following components are required:

- A flexible tunnel interface, which is a Layer 3 logical interface that supports both IPv4 and IPv6 families, on the gateway device. To configure a flexible tunnel interface, you must use the `tunnel` and `family` stanzas at the `[edit interfaces unit logical-unit-number]` hierarchy level. For more information, see [Flexible Tunnel Interfaces Overview](#).
- A flexible route, which is comprised of the following:
 - A static route that specifies, among other attributes, a tunnel encapsulation profile.
 - A tunnel encapsulation profile that specifies, among other attributes, a flexible tunnel interface.
- A tunnel de-encapsulation profile, which is typically separate from the encapsulation profile and specifies a subset of the attributes required for the encapsulation profile.

You can program a static route and map an encapsulation profile to the route using the `rib.service.proto` API file and fully define the encapsulation profile using the `flexible_tunnel_profile.proto` API file.

You can fully define de-encapsulation profiles using the `flexible_tunnel_profile.proto` API file and use the `flexible_tunnel_service.proto` API file to perform the bulk addition, modification, and deletion of

these profiles. The **flexible_tunnel_service.proto** API file also includes parameters that enable you to view a specific de-encapsulation profile.

For complete information about the JET API files, see the [JET API Guide](#).

To display information about tunnel encapsulation and de-encapsulation profiles on the Juniper Networks gateway devices, you can use the *show route detail* and *show route extensive* commands. To display information about de-encapsulation profiles, you can use the ["show flexible-tunnels profiles" on page 2033](#) command.

Routes and Routing Tables

The routes associated with tunnel encapsulation and de-encapsulation profiles have different purposes, and are therefore, stored in different routing tables on the gateway device. [Table 50 on page 1532](#) provides a summary of the tunnel profiles, the routes associated with the profiles, and routing tables.

Table 50: Summary of Tunnel Profiles, Routes, and Routing Tables

Tunnel Profiles	Routes	Route Purpose	Routing Tables	Notes
Tunnel encapsulation profile	Flexible route (static route and a mapped tunnel encapsulation profile)	Used to forward traffic to hosts in the data center.	Routing information base (RIB) or routing table determined by the API configuration.	–
Tunnel de-encapsulation profile	Automatically generated internal route	Used to associate the de-encapsulation profile to a route so that it can be downloaded to an internal routing table.	Internal routing table named __flexible_tunnel_profiles__.inet.0	Upon receipt of an encapsulated packet from a data center host, this routing table provides a means of implementing a lookup of de-encapsulation profiles and matching a particular packet with a de-encapsulation profile.

If performing detailed troubleshooting, it is helpful to know the mapping between a de-encapsulation profile and an internal route. The ["show flexible-tunnels profiles" on page 2033](#) command displays the internal route in the Route prefix field.

Flexible Tunnel Traffic Flows

The flexible tunnel feature supports the following general IPv4 and IPv6 traffic flows:

- When the gateway device receives a packet with a destination address that matches a flexible route in the routing table, the device functions as a source VTEP and takes the following action:
 - Encapsulates the packet according to parameters specified in the encapsulation profile. Encapsulation parameters include but are not limited to encapsulation type, source prefix, destination address, flexible tunnel interface, and VXLAN network identifier (VNI).
 - Forwards the packet toward the destination VM through the flexible tunnel interface specified in the encapsulation profile.
 - Implements the output features—for example, statistics, sampling, mirroring, and filters—configured on the flexible tunnel interface.
- When the gateway device receives an encapsulated packet with content that matches a de-encapsulation profile, the device functions as a destination VTEP and takes the following actions:
 - De-encapsulates the packet.
 - Implements the input features configured on the flexible tunnel interface.
 - Forwards the packet toward its destination through the flexible tunnel interface specified in the de-encapsulation profile.

Preparing the Gateway Devices

Before programming flexible tunnels using the JET APIs, you must enable gRPC Remote Procedure Calls (gRPC) on the gateway devices using the following command:

```
set system services extension-service request-response grpc ssl
```

After enabling gRPC, the gateway device is ready to receive route and tunnel service requests over a secure connection from JET applications.

For complete information about preparing the gateway devices to interact with JET APIs, see the [JET API Guide](#).

Understanding Flexible Tunnel Behavior

Keep the following in mind about the flexible tunnel feature:

- If a static route is mapped to a flexible tunnel interface that is not yet configured, the route remains inactive until the interface is configured.
- If one or more flexible routes are mapped to a flexible tunnel interface that was deleted, the flexible routes become inactive and are removed from the Packet Forwarding Engine.
- If a flexible tunnel interface goes down, is not yet configured, or is offline, traffic to be encapsulated or de-encapsulated is discarded gracefully.
- Each static route must be mapped to one tunnel encapsulation profile. In other words, there should be a one-to-one correspondence between a static route and an encapsulation profile. We do not support the mapping of two or more static routes to the same encapsulation profile.
- If adding or changing a tunnel de-encapsulation profile using the **flexible_tunnel_service.proto** API file and a conflict with another de-encapsulation profile arises, the operation fails, and an error message is provided. A conflict between two de-encapsulation profiles occurs when one or more values of the following parameters are the same:
 - UDP destination port
 - Source prefix
 - Source prefix length
 - VNI

Flexible Tunnel Limitations

The flexible tunnel feature has the following limitations:

- We do not support source lookup features, including but not limited to reverse-path-forwarding (RPF) and MAC address validation, on routing tables in which flexible routes are stored.
- Each flexible route has a tunnel encapsulation profile as a next hop instead of an IP address. As a result, we do not support next-hop features, including but not limited to load balancing, across encapsulation profiles.
- Flexible tunnels do not have control plane functionality. That is, protocols and other control plane features cannot run over these tunnels. Instead, the forwarding plane handles the flexible tunnels.

Release History Table

Release	Description
19.1R1	Starting in Junos OS Release 19.1R1, we support flexible tunnels in the following data center environment:

Static VXLAN

IN THIS CHAPTER

- [Static VXLAN | 1536](#)

Static VXLAN

IN THIS SECTION

- [Understanding Static VXLAN | 1536](#)
- [Configuring Static VXLAN | 1539](#)

Juniper Networks supports the static Virtual Extensible LAN (VXLAN) feature in a small multichassis link aggregation group (MC-LAG) network. For more information, see these topics:

Understanding Static VXLAN

IN THIS SECTION

- [Benefits of Static VXLAN | 1537](#)
- [How Static VXLAN Works | 1537](#)
- [Supported Static VXLAN Use Case | 1538](#)

Static Virtual Extensible LAN (VXLAN), also known as unicast VXLAN, enables you to statically configure source and destination virtual tunnel endpoints (VTEPs) for a particular traffic flow. A source VTEP

encapsulates and a destination VTEP de-encapsulates Layer 2 packets with a VXLAN header, thereby tunneling the packets through an underlying Layer 3 IP network.

Starting in Junos OS Release 14.1X53-D40, Juniper Networks supports static VXLAN in small MC-LAG networks.

This topic includes the following information:

Benefits of Static VXLAN

- Instead of using an Ethernet VPN (EVPN) control plane to learn the MAC addresses of hosts, static VXLAN uses a flooding-and-learning technique in the VXLAN data plane. Therefore, using static VXLAN reduces complexity in the control plane.
- For a small MC-LAG network, EVPN might be overly complex to configure and maintain. Static VXLAN provides the benefits of VXLAN and is relatively easy to design and configure.

How Static VXLAN Works

To enable static VXLAN on a Juniper Networks device that functions as a VTEP, you must configure:

- A list that includes one or more remote VTEPs with which the local VTEP can form a VXLAN tunnel.
- Ingress node replication.
- The VTEP's loopback interface (lo0):
 - Configure an anycast IP address as the primary interface.
 - Specify this interface as the source interface for a VXLAN tunnel.

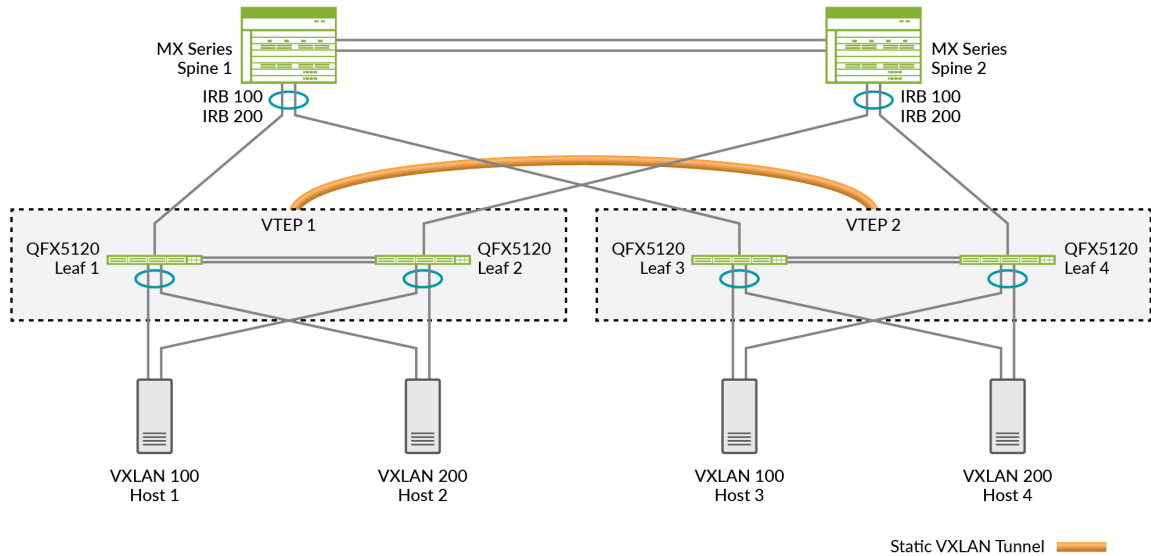
When a VTEP receives a broadcast, unknown unicast, or multicast (BUM) packet, the VTEP uses ingress node replication to replicate and flood the packet to the statically defined remote VTEPs on your list. The remote VTEPs in turn flood the packet to the hosts in each VXLAN of which the VTEPs are aware.

The VTEPs learn the MAC addresses of remote hosts from the VTEPs on the remote VTEP list and the MAC addresses of local hosts from the local access interfaces. Upon learning a MAC address of a host, the MAC address is then added to the Ethernet switching table.

Supported Static VXLAN Use Case

We support static VXLAN in a small network in which MC-LAGs are implemented. In the MC-LAG network shown in [Figure 162 on page 1538](#):

Figure 162: Sample MC-LAG Network with Static VXLAN



- The bottom layer includes hosts (Host 1 through Host 4), each of which is multihomed to leaf devices and is a member of a VXLAN.
- The middle layer includes leaf devices (Leaf 1 through Leaf 4), each of which functions as a Layer 2 VXLAN gateway. In addition, Leaf 1 and Leaf 2 form an MC-LAG and together, both leaf devices function as VTEP 1. Leaf 3 and Leaf 4 form another MC-LAG and function as VTEP 2.
- The top layer includes spine devices (Spine 1 and Spine 2), which form an MC-LAG. These devices function as IP routers.

Note the following about the configuration of Leaf 1 through Leaf 4 on which static VXLAN is configured:

- Instead of EVPN, static VXLAN is enabled.
- Each leaf device is an MC-LAG member and functions along with its MC-LAG peer as a VTEP. That is, two physical leaf devices make up each virtual VTEP. This situation impacts the configuration in the following ways:
 - On each leaf device that forms a VTEP, you must configure the same loopback address.

- On each leaf device that forms a VTEP, you must configure the same remote VTEP list.

For a sample configuration, see ["Configuring Static VXLAN" on page 1539](#).

- Instead of an IP multicast feature, ingress node replication is enabled.

Configuring Static VXLAN

You can implement Virtual Extensible LAN (VXLAN) in a small multichassis link aggregation group (MC-LAG) network using the static VXLAN feature. In the MC-LAG network, Juniper Networks devices function as source and destination virtual tunnel endpoints (VTEPs). In this environment, static VXLAN serves two purposes:

- To learn the MAC addresses of hosts in a VXLAN. To accomplish this task, static VXLAN uses the ingress node replication feature to flood broadcast, unknown unicast, and multicast (BUM) packets throughout a VXLAN. The VTEPs learn the MAC addresses of remote hosts from the VTEPs on the remote VTEP list and the MAC addresses of local hosts from the local access interfaces. Upon learning of a host's MAC address, the MAC address is then added to the Ethernet switching table.
- To encapsulate Layer 2 packets with a VXLAN header and later de-encapsulate the packets, thereby enabling them to be tunneled through an underlying Layer 3 IP network. For this task to be accomplished, on each VTEP, you configure a list of statically defined remote VTEPs with which the local VTEP can form a VXLAN tunnel.

Before You Begin

- Verify that Ethernet VPN (EVPN) is not enabled on the Juniper Networks devices that function as leaf devices.
- Verify that IP multicast is not enabled on the leaf devices.

To ensure that you understand how to configure the loopback address and remote VTEP list on each Juniper Networks device that functions as a VTEP, see [Figure 163 on page 1540](#) and [Table 51 on page 1540](#).

In the sample MC-LAG network shown in [Figure 163 on page 1540](#), note that each VTEP (VTEP 1 and VTEP 2) comprises two physical leaf devices. That is, the two physical leaf devices function as a single virtual entity. As a result, for both leaf devices that compose a VTEP, you must configure the following:

- The same loopback anycast IP address.

- The same remote VTEP list.

Figure 163: Sample MC-LAG Network with Static VXLAN

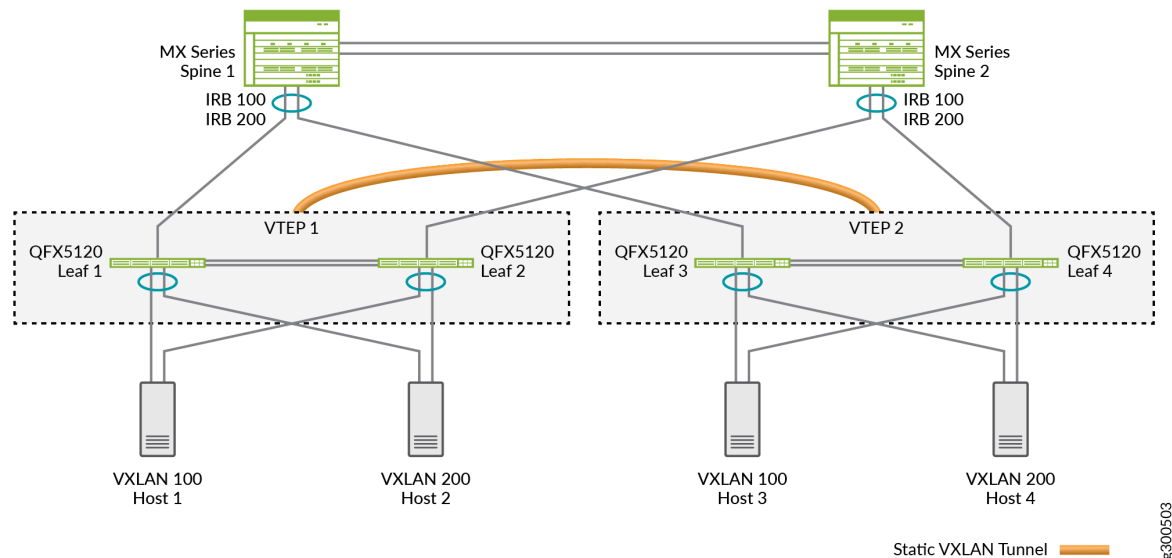


Table 51 on page 1540 shows the sample loopback address and remote VTEP list configured on each leaf device in the network.

Table 51: Sample Static VXLAN Configurations

Static VXLAN Parameters	VTEP 1: Leaf 1 Configuration	VTEP 1: Leaf 2 Configuration	VTEP 2: Leaf 3 Configuration	VTEP 2: Leaf 4 Configuration
Loopback anycast IP address	192.88.99.110/32	192.88.99.110/32	192.88.99.120/32	192.88.99.120/32
Remote VTEP list	192.88.99.120	192.88.99.120	192.88.99.110	192.88.99.110

NOTE: This procedure focuses on configuring the static VXLAN feature. It does not show how to configure peripheral but related entities such as interfaces, VLANs, and so on. However, the sample configuration that follows the procedure includes a more comprehensive configuration, including the related entities.

To enable static VXLAN on a leaf device:

1. Configure the loopback interface (lo0).
 - a. Specify an anycast IP address as the primary address for the loopback interface.

```
[edit]user@switch# set interfaces
lo0 unit logical-unit-number family inet address anycast-ip-address primary
```

- b. Specify the loopback interface as the source interface for VXLAN tunnels.

```
[edit]user@switch# set switch-options
vtep-source-interface lo0.logical-unit-number
```

2. Create a list of statically defined remote VTEPs.

```
[edit]user@switch# set switch-options
remote-vtep-list remote-vtep-loopback address(es)
```

3. For each VXLAN, enable ingress node replication.

```
[edit]user@switch# set vlans vxlan-name vxlan ingress-node-replication
```

Sample Static VXLAN Configuration

These show configuration output snippets display a static VXLAN sample configuration for leafs 1 and 2 (VTEP 1) and include the parameters outlined in [Table 51 on page 1540](#).

```
interfaces {
...  ae0 {
...    unit 0 {
        family ethernet-switching {
            interface-mode trunk;
            vlan {
                members 100 200;
            }
        }
    }
}
...  lo0 {
    unit 0 {
        family inet {
```



```

        address 192.88.99.110/32 {
            primary;
        }
    }
}
}...switch-options {
...    vtep-source-interface lo0.0;
        remote-vtep-list 192.88.99.120;
}...vlans {
    vlan100 {
        vlan-id 100;
        vxlan {
            vni 1100;
            ingress-node-replication;
        }
    }
    vlan200 {
        vlan-id 200;
        vxlan {
            vni 2200;
            ingress-node-replication;
        }
    }
}
}
```

Release History Table

Release	Description
14.1X53-D40	Starting in Junos OS Release 14.1X53-D40, Juniper Networks supports static VXLAN in small MC-LAG networks.

10

PART

Configuration Statements and Operational Commands

[EVPN Configuration Statements](#) | 1544

[VXLAN Configuration Statements](#) | 1673

[EVPN Operational Commands](#) | 1720

[VXLAN Operational Commands](#) | 2033

EVPN Configuration Statements

IN THIS CHAPTER

- [advertise-ip-prefix](#) | 1546
- [assisted-replication](#) | 1547
- [auto-derive](#) | 1549
- [bgp-peer](#) | 1551
- [designated-forwarder-election-hold-time \(evpn\)](#) | 1552
- [df-election-granularity](#) | 1554
- [duplicate-mac-detection](#) | 1555
- [esi](#) | 1558
- [esi-resolution-per-bridge-domain](#) | 1561
- [evpn](#) | 1562
- [evpn-aliasing-optimize](#) | 1567
- [evpn-ssm-reports-only](#) | 1568
- [exclusive-mac](#) | 1570
- [export \(Routing Options\)](#) | 1572
- [extended-isid-list](#) | 1574
- [extended-vlan-list](#) | 1575
- [flow-label-receive-static](#) | 1577
- [flow-label-transmit-static](#) | 1579
- [forwarding-instance identifier](#) | 1580
- [global-mac-ip-limit](#) | 1582
- [global-mac-ip-table-aging-time](#) | 1583
- [global-mac-move](#) | 1585
- [global-no-control-mac-aging](#) | 1587
- [import](#) | 1588
- [instance-type](#) | 1589
- [instance-type mac-vrf](#) | 1593

- [interconnect](#) | 1595
- [interconnected-vni-list](#) | 1597
- [interface \(EVPN Routing Instances\)](#) | 1599
- [interface-mac-ip-limit](#) | 1600
- [interface-mac-limit \(VPLS\)](#) | 1602
- [interface-state](#) | 1604
- [ip-prefix-routes](#) | 1606
- [ip-prefix-support](#) | 1610
- [label-allocation](#) | 1612
- [leaf](#) | 1613
- [loop-detect \(EVPN\)](#) | 1615
- [mac-ip-table-size](#) | 1619
- [mac-pinning \(EVPN Routing Instances\)](#) | 1620
- [mclag](#) | 1622
- [multicast-mode \(EVPN\)](#) | 1623
- [multicast-replication](#) | 1625
- [multicast-snooping-options](#) | 1628
- [no-arp-suppression](#) | 1630
- [no-default-gateway-ext-comm](#) | 1632
- [nsr-phantom-holdtime](#) | 1633
- [oism](#) | 1634
- [oism \(Multicast Snooping Options\)](#) | 1636
- [overlay-ecmp](#) | 1638
- [pbb-evpn-core](#) | 1640
- [per-esi](#) | 1641
- [per-esi-vlan](#) | 1643
- [proxy-mac](#) | 1644
- [proxy-macip-advertisement](#) | 1646
- [remote-ip-host-routes](#) | 1647
- [replicator](#) | 1649
- [route-distinguisher](#) | 1651
- [service-type](#) | 1655

- [traceoptions \(Protocols EVPN\) | 1656](#)
- [translation-vni | 1659](#)
- [vlan-id \(routing instance\) | 1662](#)
- [vpn-apply-export | 1664](#)
- [vpws-service-id | 1665](#)
- [vrf-export | 1667](#)
- [vrf-import | 1668](#)
- [vrf-target | 1670](#)

advertise-ip-prefix

IN THIS SECTION

- [Syntax | 1546](#)
- [Hierarchy Level | 1546](#)
- [Description | 1547](#)
- [Required Privilege Level | 1547](#)
- [Release Information | 1547](#)

Syntax

```
advertise-ip-prefix
```

Hierarchy Level

```
[edit protocols evpn]  
[edit routing-instances routing-instance-name protocols evpn],
```

Description

Enable the advertisement of Ethernet VPN (EVPN) pure type-5 routes for the IP prefix of the integrated routing and bridging (IRB) interface associated with customer bridge domains or Virtual Extensible LANs (VXLAN). EVPN pure type-5 routing is used when a customer Layer 2 domain is not extended across data centers and the IP subnet for the domain is confined within a single data center. If a BD or VXLAN is not extended to the EVPN, then no EVPN type-5 advertisement for the IP prefix associated with that domain or VXLAN occurs.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 17.1.

RELATED DOCUMENTATION

[Understanding EVPN with VXLAN Data Plane Encapsulation | 398](#)

[Example: Configure an EVPN-VXLAN Centrally-Routed Bridging \(CRB\) Using MX Routers as Spines | 580](#)

[EVPN Type 5 Route with MPLS encapsulation for EVPN-MPLS | 22](#)

[EVPN Type 5 Route with VXLAN encapsulation for EVPN-VXLAN | 21](#)

assisted-replication

IN THIS SECTION

- [Syntax | 1548](#)
- [Hierarchy Level | 1548](#)
- [Description | 1548](#)
- [Options | 1549](#)

- Required Privilege Level | 1549
- Release Information | 1549

Syntax

```
assisted-replication {
    leaf {
        replicator-activation-delay seconds;
    }
    replicator {
        inet ip;
        vxlan-encapsulation-source-ip (assisted-replicator-ip | ingress-replication-ip | retain-
incoming-source-ip);
    }
}
```

Hierarchy Level

```
[edit logical-systems name routing-instances name protocols evpn],
[edit routing-instances name protocols evpn],
[edit protocols evpn]
```

Description

Enable assisted replication (AR) in an EVPN network.

AR helps to optimize broadcast, unknown unicast, and multicast (BUM) traffic in EVPN networks. To enable AR, you configure devices in the EVPN network to operate as AR replicator and AR leaf devices. Use the statements in this hierarchy to define and set up AR replicators and AR leaf devices.

AR replicator devices help perform BUM traffic replication and forwarding tasks for AR leaf devices. When an AR leaf device receives BUM traffic, instead of always using ingress replication to the EVPN core, it forwards the traffic on an AR overlay tunnel to an AR replicator device that can better handle the replication load. The AR replicator device then replicates and forwards the traffic to other ingress replication overlay tunnels. You configure a loopback interface IP address on each AR replicator device for the AR overlay tunnel.

AR devices can interoperate with devices in the EVPN network that don't support or are not configured into an AR device role.

Options

leaf Configure a device as an AR leaf device.

replicator Configure a device as an AR replicator device.

These statements and their options are explained in more detail separately.

Required Privilege Level

routing

Release Information

Statement introduced in Junos OS Release 18.4R2 and 19.4R1.

RELATED DOCUMENTATION

| [Assisted Replication Multicast Optimization in EVPN Networks | 752](#)

auto-derive

IN THIS SECTION

- [Syntax | 1550](#)
- [Hierarchy Level | 1550](#)
- [Description | 1550](#)
- [Options | 1551](#)
- [Required Privilege Level | 1551](#)
- [Release Information | 1551](#)

Syntax

```
auto-derive {
    lacp;
}
```

Hierarchy Level

```
[edit interfaces aeX esi]
[edit interfaces aeX unit logical-unit-number esi]
```

Description

Configure aggregated Ethernet interfaces and aggregated Ethernet logical interfaces to automatically derive Ethernet segment identifiers (ESIs) from the Link Aggregation Control Protocol (LACP) configuration. We support this feature in the following environments:

- On Juniper Networks devices that support the automatic ESI feature and are multihomed in active-active mode in an EVPN-VXLAN overlay network.
- On Juniper Networks devices that support the automatic ESI feature and are multihomed in active-standby or active-active mode in an EVPN-MPLS overlay network.

In general, you can implement the automatic ESI feature on aggregated Ethernet and aggregated Ethernet logical interfaces as follows:

- You can configure automatic ESI on an aggregated Ethernet interface on which LACP is enabled. In this case, an ESI is generated, and that particular ESI is assigned to all logical interfaces on the aggregated Ethernet interface.
- You can configure automatic ESI on one or more logical interfaces on an aggregated Ethernet interface on which LACP is enabled. In this case, an ESI is generated for each logical interface on which the feature is enabled and assigned to that particular logical interface.
- On an aggregated Ethernet interface on which LACP is enabled, you can manually configure an ESI using the `esi identifier` configuration statement at the `[edit interfaces aeX]` hierarchy level. On one or more logical interfaces on that particular aggregated Ethernet interface, you can configure automatic ESI. In this case, an ESI is generated for each logical interface on which the feature is enabled and assigned to that particular logical interface.

For more guidelines on configuring automatic ESI and information about how the ESI is derived, see ["Understanding Automatically Generated ESIs in EVPN Networks" on page 352](#).

Options

lACP Specify that the ESI is automatically derived from the LACP configuration on the aggregated Ethernet interface.

Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

Release Information

Statement introduced on MX Series routers, QFX Series switches, and Junos on White Box in Junos OS Release 18.4R1.

bgp-peer

IN THIS SECTION

- [Syntax | 1551](#)
- [Hierarchy Level | 1552](#)
- [Description | 1552](#)
- [Options | 1552](#)
- [Required Privilege Level | 1552](#)
- [Release Information | 1552](#)

Syntax

```
bgp-peer ip-address;
```

Hierarchy Level

```
[edit routing-instances name protocols evpn mclag]
```

Description

Configure an aggregation device in a Junos Fusion Enterprise or a multichassis link aggregation group (MC-LAG) topology to interwork with an Ethernet VPN-MPLS (EVPN-MPLS) device.

Options

ip-address IP address of the BGP peer. Typically, a BGP peer is identified by the IP address of the device's loopback interface.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 17.4R1.

RELATED DOCUMENTATION

| [Understanding EVPN-MPLS Interworking with Junos Fusion Enterprise and MC-LAG](#) | 1359

designated-forwarder-election-hold-time (evpn)

IN THIS SECTION

- [Syntax](#) | 1553
- [Hierarchy Level](#) | 1553

- [Description | 1553](#)
- [Required Privilege Level | 1553](#)
- [Release Information | 1553](#)

Syntax

```
designated-forwarder-election-hold-time seconds {  
    encapsulation-type  
    extended-vni-list  
    extended-vni-all  
    no-default-gateway-ext-comm  
}
```

Hierarchy Level

```
[edit routing-instances protocols evpn]  
[edit protocols evpn]
```

Description

A designated forwarder (DF) is required when customer edge devices (CEs) are multihomed to more than one provider edge (PE) device. Without a designated forwarder, multihomed hosts would receive duplicate packets. Designated forwarders are chosen for an Ethernet segment identifier (ESI) based on type 4 route advertisements.

Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 14.1X53-D15 for QFX Series switches.

RELATED DOCUMENTATION

| [EVPN Multihoming Overview](#) | 96

df-election-granularity

IN THIS SECTION

- [Syntax](#) | 1554
- [Hierarchy Level](#) | 1554
- [Description](#) | 1554
- [Options](#) | 1555
- [Required Privilege Level](#) | 1555
- [Release Information](#) | 1555

Syntax

```
df-election-granularity {  
    per-esi {  
        lacp-oos-on-ndf;  
    }  
    per-esi-vlan;  
}
```

Hierarchy Level

```
[edit interfaces name esi]
```

Description

Configure an aggregated Ethernet interface to be the designated forwarder (DF) in an EVPN single-active configuration. When the aggregated Ethernet interface is the DF, it is emulating the behavior of an MC-LAG configuration in active-standby mode without having to configure an ICCP or ICL interface.

This behavior is different than in a standard EVPN configuration, where the logical interfaces configured on an aggregated Ethernet interface can have different designated forwarder election roles.

In an EVPN single-active configuration, the provider edge (PE) device that is the non-DF will send LACP out-of-sync packets to the customer edge device (CE). As a result, LACP goes down on the CE device, and the CE device does not use the links connected to the non-DF for sending traffic. If the connection between a CE device and a DF PE device fails, the PE device is re-elected as a DF. If the connection between a CE device and a non-DF PE device fails, the current DF PE device is not changed.

Options

per-esi lacp-oos-on-ndf Configure an aggregated Ethernet interface to be the designated forwarder (DF). Doing this ensures that the DF role of the ESI represents the role of the aggregated Ethernet interface.

per-esi-vlan Configure an aggregated Ethernet interface to have different DF election roles. This is the default behavior.

The remaining statements are explained separately. Search for a statement in [CLI Explorer](#) or click a linked statement in the Syntax section for details.

Required Privilege Level

interface

Release Information

Statement introduced in Junos OS Release 20.4R1.

duplicate-mac-detection

IN THIS SECTION

- [Syntax | 1556](#)
- [Hierarchy Level | 1556](#)
- [Description | 1556](#)
- [Options | 1556](#)

- Required Privilege Level | 1557
- Release Information | 1557

Syntax

```
duplicate-mac-detection {
    auto-recovery-time minutes;
    detection-threshold detection-threshold;
    detection-window seconds;
}
```

Hierarchy Level

```
[edit logical-systems logical-systems-name protocols evpn],
[edit logical-systems logical-systems-name routing-instances routing-instance-name protocols evpn],
[edit protocols evpn]
[edit routing-instances routing-instance-name protocols evpn],
```

Description

Configure duplicate MAC address detection settings. You can configure a threshold for MAC address mobility events and the window for detecting the number of MAC address mobility events. In addition, you can configure the optional recovery time that the Juniper Networks device waits before the duplicate MAC address is unsuppressed.

Options

auto-recovery-time

The length of time a device suppresses a duplicate MAC address. At the end of this interval, MAC address updates will resume. When an auto recovery time is not specified, duplicate MAC address updates will continue to be suppressed. To manually clear the suppression of duplicate MAC address, use the `clear evpn duplicate-mac-suppression` command.

- **Range:** 1-360 minutes

detection-threshold	<p>Number of MAC mobility events detected for a MAC address before it is identified as a duplicate MAC address. Once the threshold is reached, updates for this MAC address are suppressed.</p> <ul style="list-style-type: none"> • Default: 5 • Range: 2-20
detection-window	<p>The time interval used in detecting a duplicate MAC address. When the number of MAC mobility events for a MAC address exceeds the detection-threshold within the detection window, the MAC address is identified as a duplicate MAC address.</p> <ul style="list-style-type: none"> • Default: 180 seconds • Range: 5-600 seconds

NOTE: To ensure that the mobility advertisements have sufficient time to age out, set an auto-recovery-time greater than the detection window.

Required Privilege Level

routing

Release Information

Statement introduced in Junos OS Release 17.4R1.

RELATED DOCUMENTATION

[Overview of MAC Mobility | 16](#)

[Changing Duplicate MAC Address Detection Settings | 19](#)

esi

IN THIS SECTION

- [Syntax | 1558](#)
- [Hierarchy Level | 1558](#)
- [Description | 1558](#)
- [Options | 1560](#)
- [Required Privilege Level | 1560](#)
- [Release Information | 1560](#)

Syntax

```
esi identifier {  
    all-active | single-active;  
    auto-derive {  
        lacp;  
    }  
    source-bmac <mac-address>;  
}
```

Hierarchy Level

```
[edit interfaces interface-name]  
[edit interfaces interface-name unit logical-unit-number]  
[edit protocols evpn interconnect]  
[edit routing-instances routing-instance-name protocols evpn interconnect]
```

Description

Configure an Ethernet Segment Identifier (ESI) in either EVPN multihoming active/standby or active/active mode by using one of the following methods:

- Manually configure an ESI on a physical, aggregated Ethernet, or logical interface using the `esi identifier` configuration statement.
- Configure an aggregated Ethernet interface or aggregated Ethernet logical interface to automatically derive an ESI from the Link Aggregation Control Protocol (LACP) configuration. Use the `auto-derive lacp` configuration statement.

If you configure automatic ESI on an aggregated Ethernet interface, the device generates an ESI. The device assigns that particular ESI to all logical interfaces on the aggregated Ethernet interface. You can also configure automatic ESI on one or more logical interfaces on an aggregated Ethernet interface. In that case, the device generates an ESI for each logical interface on which you enabled the feature.

NOTE: You can't commit a configuration in which you try to configure the following on the same physical or logical aggregated Ethernet interface:

- Manually configure an ESI on the aggregated Ethernet interface.
- Enable the automatic ESI feature on the aggregated Ethernet interface.

Starting in Junos OS Releases 15.1F6 and 17.1R1 for MX Series routers and Junos OS Release 17.3R1 for EX9200 switches, you can specify an ESI on a logical interface. In earlier releases, you can configure an ESI only on a physical or aggregated Ethernet interface. For example:

```
set interfaces ae0 esi 00:11:22:33:44:55:66:77:88:99
```

When configuring an ESI, remember that the ESI is a factor in the designated forwarder (DF) election process. For example, let's look at a case in which you configure EVPN multihoming active/standby on aggregated Ethernet interface ae0. In this case, ae0 also has logical interfaces on unit 1 and unit 2. The DF election process might put ae0 into the down state due to the ESI and other determining factors. The logical interfaces configured on ae0 would also be down, and unable to provide services to their respective customer sites (VLANs).

If you specify an ESI on a logical interface, the DF election process now occurs at the individual logical interface level. You can better utilize logical interfaces. For example, you configure logical interfaces ae0.1 and ae0.2 on aggregated Ethernet interface ae0. You also enable EVPN multihoming active-standby on both logical interfaces. Then the DF election brings ae0.1 down based on the ESI on ae0.1 and other DF determining factors. Despite logical interface ae0.1 being down, logical interface ae0.2 and other ae0 logical interfaces can remain up. The DF election process doesn't disrupt customer services to the respective customer sites (VLANs).

NOTE: A physical interface on which you configure an ESI won't become a DF if any one of its logical units is down. As a result, if you don't want logical interfaces in one unit to forward traffic, then delete the logical unit rather than disabling it. We recommend that you do not explicitly disable any logical units associated with a physical interface on which you configured an ESI.

Options

esi identifier Ten octet value. Reserved values: ESI value 0 and all fixed filters. You can't use the reserved values for configuring a multihomed Ethernet segment.

NOTE:

- You can't configure two interfaces (physical, logical, or aggregated Ethernet) with the same ESI value.
- The left most octet must be configured as 00. The other 9 octets are fully configurable.

all-active Configure the EVPN active-active multihoming mode of operation.

single-active Configure the EVPN active-standby multihoming mode of operation.

The remaining statements are explained separately. Search for a statement in [CLI Explorer](#) or click a linked statement in the Syntax section for details.

Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 14.1.

source-bmac option introduced in Junos OS Release 18.1R1 on MX Series routers with MPCs and MICs.

auto-derive stanza introduced in Junos OS Release 18.4R1 on MX Series routers and QFX Series switches.

Support at the following hierarchy levels introduced in Junos OS Release 20.3R1 on QFX Series switches: [edit protocols evpn interconnect] and [edit routing-instances *routing-instance-name* protocols evpn interconnect].

RELATED DOCUMENTATION

[evpn | 1562](#)

[Example: Configuring an ESI on a Logical Interface With EVPN Multihoming | 336](#)

[Understanding Automatically Generated ESIs in EVPN Networks | 352](#)

esi-resolution-per-bridge-domain

IN THIS SECTION

- [Syntax | 1561](#)
- [Hierarchy Level | 1561](#)
- [Description | 1562](#)
- [Required Privilege Level | 1562](#)
- [Release Information | 1562](#)

Syntax

```
esi-resolution-per-bridge-domain;
```

Hierarchy Level

```
[edit protocols evpn]
```

Description

Enables a device that does not support per bridge domain routing to interoperate with devices in the EVPN MPLS network that advertises MPLS labels on a per bridge domain by building next hop routing tables based on a per bridge domain basis.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Command introduced in Junos OS Release 21.1.

evpn

IN THIS SECTION

- [Syntax | 1562](#)
- [Hierarchy Level | 1565](#)
- [Description | 1565](#)
- [Options | 1565](#)
- [Required Privilege Level | 1566](#)
- [Release Information | 1566](#)

Syntax

```
evpn {  
  assisted-replication {  
    leaf {  
      replicator-activation-delay seconds;  
    }  
    replicator {
```

```

        inet ip;
        vxlan-encapsulation-source-ip (assisted-replicator-ip | ingress-replication-ip |
retain-incoming-source-ip);
    }
}
default-gateway {
    (advertise | do-not-advertise| no-gateway-community);
}
designated-forwarder-election-hold-time seconds;
designated-forwarder-preference-least;
duplicate-mac-detection {
    auto-recovery-time minutes;
    detection-threshold detection-threshold;
    detection-window seconds;
}
encapsulation encapsulation-type;
es-import-oldstyle;
es-label-oldstyle;
esi-resolution-per-bridge-domain;
evpn-etree;
extended-isid-list (all | single-isid | [isid-list] | isid-range);
extended-vlan-list (vlan-id | [list of vlans]);
extended-vni-list (all | [list of VNIs]);
interconnect {
    esi identifier {
        all-active;
        df-election-type {
            mod;
            preference {
                value value (0..65535);
            }
        }
    }
}
interconnected-vni-list (all | [vni-list]);
route-distinguisher (number:id | ip-address:id);
vrf-export (policy-name | [policy-names]);
vrf-import (policy-name | [policy-names]);
vrf-target {
    community;
    import community-name;
    export community-name;
}
}

```

```

interface interface-name {
    ignore-encapsulation-mismatch;
    interface-mac-limit limit {
        packet-action drop;
    }
    mac-pinning ;
    no-mac-learning;
    static-mac mac-address;
}
interface-mac-limit limit {
    packet-action drop;
}
leave-sync-route-oldstyle;
label-allocation per-instance;
mac-list name {
    mac-addresses;
}
mac-mobility {
    no-sequence-numbers;
}
mac-statistics;
mac-table-size limit {
    packet-action drop;
}
multicast-mode client | ingress-replication;
no-core-isolation;
no-default-gateway-ext-comm;
no-mac-learning;
oism {
    originate-smet-on-revenue-vlan-too;
    pim-evpn-gateway {
        external-irb name;
    }
    supplemental-bridge-domain-irb supplemental-bridge-domain-irb;
}
p2mp-bud-support;
pbb-evpn-core;
pmsi-tunnel-endpoint pmsi-tunnel-endpoint;
remote-ip-host-routes {
    import [ import ... ];
    no-advertise-community;
}
smet-etag-carry-vid;

```

```

traceoptions {
    file filename <files number> <size size> <world-readable | no-world-readable>;
    flag flag <flag-modifier>;
}
vni-options vni vxlan-network-identifier {
    designated-forwarder-election-hold-time seconds;
    vrf-target {
        community;
        auto;
        import community-name;
        export community-name;
    }
}
}

```

Hierarchy Level

```

[edit logical-systems logical-system-name routing-instances routing-instance-name protocols],
[edit protocols],
[edit routing-instances routing-instance-name protocols]

```

Description

Enable an Ethernet VPN (EVPN) on the routing instance.

Options

default-gateway	Specify the behavior for IRB interfaces defined as default gateways in an EVPN fabric: <ul style="list-style-type: none"> • advertise (Default behavior) • do-not-advertise: Don't advertise the default gateway MAC address. • no-gateway-community: Advertise virtual gateway addresses and IRB MAC addresses to EVPN peer devices so that Ethernet-only PE devices learn those MAC addresses.
designated-forwarder-election-hold-time <i>seconds</i>	Time in seconds to wait before electing a designated forwarder (DF). Peer provider edge (PE) devices elect a designated forwarder (DF) when customer edge

devices (CEs) are multihomed to more than one PE device. Without a designated forwarder, multihomed hosts would receive duplicate packets. The designated forwarder choice for an Ethernet segment identifier (ESI) is based on EVPN Type 4 route advertisements.

- **Range:** 1 through 1800 seconds

designated-forwarder-preference-least	Use least preference value for DF election. By default, the preference-based DF election is based on the highest preference value configured.
es-import-oldstyle	Enable non-compliant ES import route-target computation.
es-label-oldstyle	Enable devices to use the lower-order bits for the ESI label field. This non-compliant ESI 24-bit ESI label can be found in previous Junos OS versions.
evpn-etree	Configure an Ethernet VPN E-TREE service.
leave-sync-route-oldstyle	Enable device to interoperate with devices that do not support the maximum response time attributes on EVPN Type 8 routes.
mac-mobility no-sequence-numbers	Disable MAC mobility. This feature is enabled by default. See "Overview of MAC Mobility" on page 16 .
no-core-isolation	Disable the core isolation feature (enabled by default on supporting platforms), which is desirable in certain EVPN fabric use cases. See "Understanding When to Disable EVPN-VXLAN Core Isolation" on page 329 for details.
pmsi-tunnel-endpoint <i>pmsi-tunnel-endpoint</i>	Configure the PMSI tunnel endpoint on the ingress router to use the loopback address (router ID) that is the MPLS network PMSI tunnel endpoint for EVPN Type 3 routes.
smet-etag-carry-vid	Enable a device to interoperate with other devices in the EVPN network that use VNIs (VLAN ID) as the Ethernet Tag ID.

The remaining statements are explained separately.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 13.2.

`designated-forwarder-election-hold-time` *seconds* statement introduced in Junos OS Release 14.1.

`extended-vlan-list` *vlan-id* | [*vlan-id set*] statement introduced in Junos OS Release 14.1.

NOTE: QFX10000 switches don't support the `extended-vlan-list` statement.

`designated-forwarder-preference-least` and `service-type` statements introduced in Junos OS Release 17.3 for MX Series routers with MPC interfaces and MIC interfaces.

`no-core-isolation` statement added in Junos OS Releases 17.3R3, 17.4R2, 18.1R3, 18.2R2, and 18.3R1.

Support for logical systems on MX Series routers added in Junos OS Release 17.4R1.

`p2mp-bud-support` statement introduced in Junos OS Release 18.2R1.

`assisted-replication` statement introduced in Junos OS Release 18.4R2 and 19.4R1.

`pmsi-tunnel-endpoint` statement introduced in Junos OS Release 20.3R1.

`oism` statement hierarchy introduced in Junos OS Release 21.2R1.

RELATED DOCUMENTATION

[Implementing EVPN-VXLAN for Data Centers](#) | 432

[Configuring EVPN Routing Instances](#) | 10

[Tracing EVPN Traffic and Operations](#) | 42

[Provider Backbone Bridging \(PBB\) and EVPN Integration Overview](#) | 1414

[Optimized Inter-Subnet Multicast in EVPN Networks](#) | 773

evpn-aliasing-optimize

IN THIS SECTION

- [Syntax](#) | 1568
- [Hierarchy Level](#) | 1568
- [Description](#) | 1568
- [Required Privilege Level](#) | 1568
- [Release Information](#) | 1568

Syntax

```
evpn-aliasing-optimize;
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-options forwarding-table ],  
[edit routing-options forwarding-table]
```

Description

Enables EVPN aliasing optimization. EVPN aliasing optimization allows the device to quickly recover when a link to a multihomed PE device fails. It reduces convergence time, and improves device performance. By default, `evpn-aliasing-optimize` is enabled on the PTX10008 Router.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Command introduced in Junos Evolve OS Release 21.1.

evpn-ssm-reports-only

IN THIS SECTION

- [Syntax | 1569](#)
- [Hierarchy Level | 1569](#)
- [Description | 1569](#)
- [Required Privilege Level | 1569](#)
- [Release Information | 1570](#)

Syntax

```
evpn-ssm-reports-only;
```

Hierarchy Level

```
[edit bridge-domains bridge-domain-name protocols igmp-snooping],
[edit bridge-domains bridge-domain-name protocols mld-snooping],
[edit protocols igmp-snooping vlan vlan-name],
[edit protocols mld-snooping vlan vlan-name],
[edit routing-instances routing-instance-name bridge-domains bridge-domain-name protocols igmp-
snooping],
[edit routing-instances routing-instance-name bridge-domains bridge-domain-name protocols mld-
snooping],
[edit routing-instances routing-instance-name protocols igmp-snooping],
[edit routing-instances routing-instance-name protocols mld-snooping]
```

Description

Accept and process only (S,G) reports for source-specific multicast (SSM) groups when:

- You configure IGMP snooping for IGMPv3 with IPv4 multicast traffic in an EVPN network.
- You configure Multicast Listener Discovery (MLD) snooping for MLDv2 with IPv6 multicast traffic in an EVPN network.

Devices use IGMP and MLD to discover and indicate interest in multicast groups. IGMP or MLD snooping restricts the forwarding of multicast traffic to only those interfaces in a bridge domain (VLAN) that have interested listeners.

By default, when IGMP snooping or MLD snooping is enabled, the device accepts and processes IGMPv2, IGMPv3, MLDv1 or MLDv2 (*,G) (any-source multicast [ASM]) reports. IGMPv3 and MLDv2 also support (S,G) source-specific multicast (SSM) reports and queries, but accepting and processing both (*,G) and (S,G) reports at the same time is not supported. When you enable the `evpn-ssm-reports-only` option for IGMP or MLD snooping, the device honors only IGMPv3 or MLDv2 (S,G) reports, and drops any IGMPv2, IGMPv3, MLDv1 or MLDv2 (*,G) reports.

Required Privilege Level

routing

Release Information

Statement introduced in Junos OS Release 18.4R1.

Statement added in Junos OS Releases 19.4R2 and 20.3R1 on ACX Series routers with EVPN multicast support.

RELATED DOCUMENTATION

[Overview of Multicast Forwarding with IGMP or MLD Snooping in an EVPN-MPLS Environment | 1013](#)

[Overview of Multicast Forwarding with IGMP Snooping or MLD Snooping in an EVPN-VXLAN Environment | 675](#)

[Configure Multicast Forwarding with MLD Snooping in an EVPN-MPLS Environment | 1038](#)

exclusive-mac

IN THIS SECTION

- [Syntax | 1570](#)
- [Hierarchy Level | 1571](#)
- [Description | 1571](#)
- [Options | 1571](#)
- [Required Privilege Level | 1571](#)
- [Release Information | 1572](#)

Syntax

```
exclusive-mac virtual-mac-mac-address/mask;
```

Hierarchy Level

```
[edit protocols l2-learning global-mac-move]
```

Description

Exclude MAC addresses from the MAC move limit algorithm.

The global MAC move feature is used to track MAC addresses when they appear on a different physical interface or within a different unit of the same physical interface. When you configure the `exclusive-mac virtual-mac-mac-address/mask` parameter at the `[edit protocols l2-learning global-mac-move]` hierarchy level, specified MAC addresses are excluded and will not be tracked. In addition, excluded MAC addresses cannot be pinned when `mac-pinning` is set on an interface.

This feature can be useful in OVSDB-managed topologies with VRRP servers deployed in a redundancy configuration (primary/member), and when MAC move limit is configured. Both servers could negotiate primary role, and the same MAC address could be learned under the global MAC move feature while negotiation is occurring. In such cases, excluding the MAC address of the VRRP servers by using the `exclusive-mac` statement prevents this “false” move from being tracked.

The following example excludes VRRP V2 virtual router MAC addresses, as defined in RFC 3768:

```
[edit]
set protocols l2-learning global-mac-move exclusive-mac 00:00:5e:00:01:00/40
```

The following example excludes VRRP V3 virtual router MAC addresses, as defined in RFC 5798:

```
[edit]
set protocols l2-learning global-mac-move exclusive-mac 00:00:5e:00:02:00/40
```

Options

<i>virtual-mac-mac-address/mask</i>	Specify a MAC address and a mask. If the mask is 48, only the exact MAC address is excluded. If the mask is 40, all the MAC addresses that have the same first 5 bytes are excluded.
-------------------------------------	--

Required Privilege Level

view-level—To view this statement in the configuration.

control-level—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 14.1X53-D45.

RELATED DOCUMENTATION

| [Configuring MAC Move Parameters](#)

export (Routing Options)

IN THIS SECTION

- [Syntax | 1572](#)
- [Hierarchy Level | 1572](#)
- [Description | 1573](#)
- [Options | 1573](#)
- [Required Privilege Level | 1573](#)
- [Release Information | 1573](#)

Syntax

```
export [ policy-name ];
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name routing-
options forwarding-table],
[edit logical-systems logical-system-name routing-options forwarding-table],
```

```
[edit routing-instances routing-instance-name routing-options forwarding-table],
[edit routing-options forwarding-table]
```

Description

Apply one or more policies to routes being exported from the routing table into the forwarding table.

In the `export` statement, list the name of the routing policy to be evaluated when routes are being exported from the routing table into the forwarding table. Only active routes are exported from the routing table.

You can reference the same routing policy one or more times in the same or a different `export` statement.

You can apply export policies to routes being exported from the routing table into the forwarding table for the following features:

- Per-packet load balancing
- Class of service (CoS)

Options

policy-name—Name of one or more policies.

Required Privilege Level

`routing`—To view this statement in the configuration.

`routing-control`—To add this statement to the configuration.

Release Information

Statement introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

| [Example: Load Balancing BGP Traffic](#)

extended-isid-list

IN THIS SECTION

- [Syntax | 1574](#)
- [Hierarchy Level | 1574](#)
- [Description | 1574](#)
- [Options | 1574](#)
- [Required Privilege Level | 1575](#)
- [Release Information | 1575](#)

Syntax

```
extended-isid-list (single-isid | isid-list | isid-range | all);
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols]  
[edit routing-instances routing-instance-name protocol evpn]
```

Description

Specify the service identifiers (ISIDs) that are being extended over the Ethernet VPN (EVPN) core.

NOTE: When you deactivate the `extended-isid-list` statement, with or without configuring the automatic derivation of the BGP route target (auto-RT) for advertised prefixes, both Type 2 and Type 3 routes are retained.

Options

single-isid Specify a single ISID.

isid-list	Specify a list of multiple ISIDs.
isid-range	Specify a range of ISIDs.
all	Specify that all ISIDs are being extended over the EVPN core.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 16.1.

Support for logical systems on MX Series routers added in Junos OS Release 17.4R1.

RELATED DOCUMENTATION

| [evpn](#) | [1562](#)

extended-vlan-list

IN THIS SECTION

- [Syntax](#) | [1576](#)
- [Hierarchy Level](#) | [1576](#)
- [Description](#) | [1576](#)
- [Options](#) | [1576](#)
- [Required Privilege Level](#) | [1576](#)
- [Release Information](#) | [1576](#)

Syntax

```
extended-vlan-list vlan-id | [vlan-id set];
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols]
[edit routing-instances routing-instance-name instance-type virtual-switch protocols evpn]
```

Description

Specify the VLAN or range of VLANs that are extended over the WAN, wherein all the single VLAN bridge domains corresponding to these VLANs are stretched.

NOTE: The `extended-vni-list` statement is an exclusive command. You cannot configure the `extended-vni-list` statement with either the `extended-vlan-list` or `extended-vni-all` statements.

Options

- | | |
|--------------------|--|
| <i>vlan-id</i> | VLAN ID to be EVPN extended. |
| <i>vlan-id set</i> | List of VLAN IDs to be EVPN extended. |
| | <ul style="list-style-type: none"> • Range: 1 through 4094 VLANs |

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 14.1.

Support for logical systems on MX Series routers added in Junos OS Release 17.4R1.

RELATED DOCUMENTATION

| [evpn](#) | [1562](#)

flow-label-receive-static

IN THIS SECTION

- [Syntax](#) | [1577](#)
- [Hierarchy Level](#) | [1577](#)
- [Description](#) | [1577](#)
- [Required Privilege Level](#) | [1578](#)
- [Release Information](#) | [1578](#)

Syntax

```
flow-label-receive-static;
```

Hierarchy Level

```
[edit protocols l2circuit neighbor neighbor-id interface interface-name]  
[edit routing-instances instance-name protocols evpn],  
[edit routing-instances instance-name protocols evpn interface interface-name]
```

Description

Configure the router to pop the flow label on pseudowire packets received from a remote provider edge (PE) router. The ingress PE router inserts the flow label in the pseudowire packet regardless of the information exchanged in the signaling plane. If the egress PE router cannot handle the pseudowire packet with the flow label, it drops the packet.

Flow-aware transport of pseudowires (FAT) flow labels enable load balancing of MPLS packets across equal-cost multipath (ECMP) paths or link aggregation groups (LAGs) without the need for deep packet inspection of the payload. You can use this statement to enable devices to pop FAT flow labels for:

- Forwarding equivalence class (FEC) 128 pseudowires in:

- Virtual private LAN service (VPLS) networks.
- Virtual private wire service (VPWS) networks.

In these cases, configure this statement in the `[edit protocols l2circuit neighbor neighbor-id interface interface-name]` hierarchy.

- Pseudowires in Ethernet VPN (EVPN)-MPLS networks with VPWS (EVPN-VPWS):

- Globally in an EVPN-VPWS routing instance.

In this case, configure this statement in the `[edit routing-instances instance-name protocols evpn]` hierarchy.

- For specific interfaces in an EVPN-VPWS routing instance.

In this case, configure this statement in the `[edit routing-instances instance-name protocols evpn interface interface-name]` hierarchy.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 14.1.

Support in a EVPN-VPWS routing instance added in Junos OS Release 21.1R1.

RELATED DOCUMENTATION

Configuring the FAT Flow Label for FEC 128 VPWS Pseudowires for Load-Balancing MPLS Traffic

[FAT Flow Labels in EVPN-VPWS Routing Instances](#) | 1096

flow-label-transmit-static

IN THIS SECTION

- [Syntax | 1579](#)
- [Hierarchy Level | 1579](#)
- [Description | 1579](#)
- [Required Privilege Level | 1580](#)
- [Release Information | 1580](#)

Syntax

```
flow-label-transmit-static;
```

Hierarchy Level

```
[edit protocols l2circuit neighbor neighbor-id interface interface-name],  
[edit routing-instances instance-name protocols evpn],  
[edit routing-instances instance-name protocols evpn interface interface-name]
```

Description

Configure the router to push the flow label on pseudowire packets it sends to a remote provider edge (PE) router. The ingress PE router inserts the flow label in the pseudowire packet regardless of the information exchanged in the signaling plane. If the egress PE router can't handle a pseudowire packet that contains the flow label, it drops the packet.

Flow-aware transport of pseudowires (FAT) flow labels enable load balancing of MPLS packets across equal-cost multipath (ECMP) paths or link aggregation groups (LAGs) without the need for deep packet inspection of the payload. You can use this statement to enable devices to push FAT flow labels for:

- Forwarding equivalence class (FEC) 128 pseudowires in:
 - Virtual private LAN service (VPLS) networks.
 - Virtual private wire service (VPWS) networks.

In these cases, configure this statement in the [edit protocols l2circuit neighbor *neighbor-id* interface *interface-name*] hierarchy.

- Pseudowires in Ethernet VPN (EVPN)-MPLS networks with VPWS (EVPN-VPWS):

- Globally in an EVPN-VPWS routing instance.

In this case, configure this statement in the [edit routing-instances *instance-name* protocols evpn] hierarchy.

- For specific interfaces in an EVPN-VPWS routing instance.

In this case, configure this statement in the [edit routing-instances *instance-name* protocols evpn interface *interface-name*] hierarchy.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 14.1.

Support in a EVPN-VPWS routing instance added in Junos OS Release 21.1R1.

RELATED DOCUMENTATION

Configuring the FAT Flow Label for FEC 128 VPWS Pseudowires for Load-Balancing MPLS Traffic

[FAT Flow Labels in EVPN-VPWS Routing Instances](#) | 1096

forwarding-instance identifier

IN THIS SECTION

- [Syntax](#) | 1581
- [Hierarchy Level](#) | 1581
- [Description](#) | 1581

- Options | 1581
- Required Privilege Level | 1581
- Release Information | 1582

Syntax

```
forwarding-instance identifier identifier (1-99);
```

Hierarchy Level

```
[edit routing-instances mac-vrf routing instance name],
[edit tenants tenant name routing-instances mac-vrf routing instance
name]
```

Description

QFX10000 line of switches only—Configure the forwarding-instance parameters for a MAC-VRF routing instance.

If you configure the forwarding instance with an identifier, then the switch maps the specified MAC-VRF routing instance to the forwarding instance specified by the *identifier* variable.

If you do not configure an identifier, then the switch maps the specified MAC-VRF routing instance to the system default forwarding instance, default-switch.

Options

(Optional) <i>identifier</i> <i>forwarding instance identifier</i> (1-99)	Forwarding instance identifier. Specify the forwarding instance identifier as an integer value in the range of 1-99.
--	--

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 20.4R1.

RELATED DOCUMENTATION

| [MAC-VRF Routing Instance Type Overview](#) | [459](#)

global-mac-ip-limit

IN THIS SECTION

- [Syntax](#) | [1582](#)
- [Hierarchy Level](#) | [1582](#)
- [Description](#) | [1583](#)
- [Options](#) | [1583](#)
- [Required Privilege Level](#) | [1583](#)
- [Release Information](#) | [1583](#)

Syntax

```
global-mac-ip-limit {  
    number;  
}
```

Hierarchy Level

```
[edit logical-systems name protocols l2-learning],  
[edit protocols l2-learning]
```

Description

Limit the number of entries that can be added systemwide to the MAC-IP bindings database.

Options

number Specify the maximum number of entries that can added systemwide to the MAC-IP bindings database.. When the specified maximum is reached, no new entries are added to the database.

- **Range:** 20 through 1,048,575.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 17.4R2.

Statement introduced in Junos OS Release 18.1R1 for QFX Series switches.

RELATED DOCUMENTATION

[EVPN Proxy ARP and ARP Suppression, and Proxy NDP and NDP Suppression | 361](#)

[global-mac-ip-table-aging-time | 1583](#)

[interface-mac-ip-limit | 1600](#)

[mac-ip-table-size | 1619](#)

global-mac-ip-table-aging-time

IN THIS SECTION

● [Syntax | 1584](#)

● [Hierarchy Level | 1584](#)

- [Description | 1584](#)
- [Options | 1584](#)
- [Required Privilege Level | 1584](#)
- [Release Information | 1585](#)

Syntax

```
global-mac-ip-table-aging-time seconds;
```

Hierarchy Level

```
[edit logical-systems logical-system-name protocols l2-learning],  
[edit protocols l2-learning]
```

Description

Configure the timeout interval systemwide for entries in the MAC-IP address bindings database.

Options

seconds Specify the time that is elapsed before entries in the MAC-IP bindings database are timed out and deleted.

- **Range:** 60 through 1 million seconds
- **Default:** 1200 seconds (20 minutes)

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 17.4R1.

Statement introduced in Junos OS Release 18.1R1 for QFX Series switches.

RELATED DOCUMENTATION

[EVPN Proxy ARP and ARP Suppression, and Proxy NDP and NDP Suppression | 361](#)

[global-mac-ip-limit | 1582](#)

[interface-mac-ip-limit | 1600](#)

[mac-ip-table-size | 1619](#)

global-mac-move

IN THIS SECTION

- [Syntax | 1585](#)
- [Hierarchy Level | 1586](#)
- [Description | 1586](#)
- [Default | 1586](#)
- [Required Privilege Level | 1586](#)
- [Release Information | 1586](#)

Syntax

```
global-mac-move {  
    cooloff-time seconds;  
    disable-action;  
    exclusive-mac virtual-mac-mac-address/mask;  
    interface-recovery-time seconds;  
    notification-time seconds;  
    reopen-time seconds;
```

```

    statistical-approach-wait-time seconds;
    threshold-count count;
    threshold-time seconds;
    virtual-mac mac-address /mask;
}

```

Hierarchy Level

```
[edit protocols l2-learning]
```

Description

Set parameters for media access control (MAC) address move reporting.

Default

By default, MAC moves notify every second, with a threshold time of 1 second and a threshold count of 50.

Required Privilege Level

view-level—To view this statement in the configuration.

control-level—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 9.4.

Support for logical systems added in Junos OS Release 9.6.

Support for disable-action and reopen-time added in Junos OS Release 13.2.

Support for exclusive-mac added in Junos OS Release 14.1X53-D45.

Statements cooloff-time, interface-recovery-time, statistical-approach-wait-time, and virtual-mac moved from vpls-mac-move to global-mac-move hierarchy level in Junos OS Release 17.4.

RELATED DOCUMENTATION

[Configuring MAC Move Parameters](#)

MAC Moves Loop Prevention in VPLS Network Overview

Example: Configuring Loop Prevention in VPLS Network Due to MAC Moves

virtual-mac

global-no-control-mac-aging

IN THIS SECTION

- [Syntax | 1587](#)
- [Hierarchy Level | 1587](#)
- [Description | 1587](#)
- [Required Privilege Level | 1588](#)
- [Release Information | 1588](#)

Syntax

```
global-no-control-mac-aging;
```

Hierarchy Level

```
[edit logical-systems name protocols l2-learning],  
[edit protocols l2-learning]
```

Description

Disable control MAC-address aging from software.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 20.4R1.

import

IN THIS SECTION

- [Syntax | 1588](#)
- [Hierarchy Level | 1588](#)
- [Description | 1589](#)
- [Options | 1589](#)
- [Required Privilege Level | 1589](#)
- [Release Information | 1589](#)

Syntax

```
import [ policy-names ];
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name routing-
options resolution rib],
[edit logical-systems logical-system-name routing-options resolution rib],
[edit routing-instances routing-instance-name routing-options resolution rib],
[edit routing-options resolution rib]
```

Description

Specify one or more import policies to use for route resolution.

Options

policy-names—Name of one or more import policies.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

| *Example: Configuring Route Resolution on PE Routers*

instance-type

IN THIS SECTION

- [Syntax | 1590](#)
- [Hierarchy Level | 1590](#)
- [Description | 1590](#)
- [Options | 1590](#)
- [Required Privilege Level | 1592](#)
- [Release Information | 1592](#)

Syntax

```
instance-type type;
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name],  
[edit routing-instances routing-instance-name]
```

Description

Define the type of routing instance.

Options

NOTE: On ACX Series routers, you can configure only the forwarding, virtual router, and VRF routing instances.

type—Can be one of the following:

- *evpn*—(MX 3D Series routers, QFX switches and EX9200 switches)—Enable an Ethernet VPN (EVPN) on the routing instance.
hierarchy level.
- *evpn-vpws*—Enable an Ethernet VPN (EVPN) Virtual Private Wire Service (VPWS) on the routing instance.
- *forwarding*—Provide support for filter-based forwarding, where interfaces are not associated with instances. All interfaces belong to the default instance. Other instances are used for populating RPD learned routes. For this instance type, there is no one-to-one mapping between an interface and a routing instance. All interfaces belong to the default instance *inet.0*.
- *l2backhaul-vpn*—Provide support for Layer 2 wholesale VLAN packets with no existing corresponding logical interface. When using this instance, the router learns both the outer tag and inner tag of the incoming packets, when the *instance-role* statement is defined as *access*, or the outer VLAN tag only, when the *instance-role* statement is defined as *nni*.
- *l2vpn*—Enable a Layer 2 VPN on the routing instance. You must configure the *interface*, *route-distinguisher*, *vrf-import*, and *vrf-export* statements for this type of routing instance.

- `layer2-control`—(MX Series routers only) Provide support for RSTP or MSTP in customer edge interfaces of a VPLS routing instance. This instance type cannot be used if the customer edge interface is multihomed to two provider edge interfaces. If the customer edge interface is multihomed to two provider edge interfaces, use the default BPDU tunneling.
- `mac-vrf`—(MX Series and vMX routers; QFX5100, QFX5110, QFX5120, QFX5130-32CD, QFX5200, QFX10002 (including QFX10002-60C), QFX10008, and QFX10016 switches; EX9200 Series switches only) Use this routing instance type to configure multiple customer-specific EVPN instances (EVIs) of type `mac-vrf`, each of which can support a different EVPN service type. This configuration results in customer-specific virtual routing and forwarding (VRF) tables with MAC addresses on each Juniper Networks device that serves as a virtual tunnel endpoint (VTEP) in the EVPN-VXLAN network. This type of routing instance is for EVPN unicast routes only.
- `mpls-forwarding`—(MX Series routers only) Allow filtering and translation of route distinguisher (RD) values in IPv4 and IPv6 VPN address families on both routes received and routes sent for selected BGP sessions. In particular, for Inter-AS VPN Option-B networks, this option can prevent the malicious injection of VPN labels from one peer AS boundary router to another.
- `mpls-internet-multicast`—(EX Series, M Series, MX Series, and T Series routers only) Provide support for ingress replication provider tunnels to carry IP multicast data between routers through an MPLS cloud, using MBGP or next-generation MVPN.
- `no-forwarding`—This is the default routing instance. Do not create a corresponding forwarding instance. Use this routing instance type when a separation of routing table information is required. There is no corresponding forwarding table. All routes are installed into the default forwarding table. IS-IS instances are strictly nonforwarding instance types.
- `virtual-router`—Enable a virtual router routing instance. This instance type is similar to a VPN routing and forwarding instance type, but used for non-VPN-related applications. You must configure the `interface` statement for this type of routing instance. You do not need to configure the `route-distinguisher`, `vrf-import`, and `vrf-export` statements.
- `virtual-switch`—(MX Series routers, EX9200 switches, and QFX switches only) Provide support for Layer 2 bridging. Use this routing instance type to isolate a LAN segment with its Spanning Tree Protocol (STP) instance and to separate its VLAN identifier space.
- `vpls`—Enable VPLS on the routing instance. Use this routing instance type for point-to-multipoint LAN implementations between a set of sites in a VPN. You must configure the `interface`, `route-distinguisher`, `vrf-import`, and `vrf-export` statements for this type of routing instance.
- `vrf`—VPN routing and forwarding (VRF) instance. Provides support for Layer 3 VPNs, where interface routes for each instance go into the corresponding forwarding table only. Required to create a Layer 3 VPN. Create a VRF table (*instance-name.inet.0*) that contains the routes originating from and destined for a particular Layer 3 VPN. For this instance type, there is a one-to-one mapping between an interface and a routing instance. Each VRF instance corresponds with a forwarding table. Routes

on an interface go into the corresponding forwarding table. You must configure the interface, route-distinguisher, vrf-import, and vrf-export statements for this type of routing instance.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced before Junos OS Release 7.4.

virtual-switch and layer2-control options introduced in Junos OS Release 8.4.

mpls-internet-multicast option introduced in Junos OS Release 11.1 for the EX Series, M Series, MX Series, and T Series.

evpn option introduced in Junos OS Release 13.2 for MX 3D Series routers.

evpn option introduced in Junos OS Release 17.3 for the QFX Series.

forwarding option introduced in Junos OS Release 14.2 for the PTX Series.

mpls-forwarding option introduced in Junos OS Release 16.1 for the MX Series.

evpn-vpws option introduced in Junos OS Release 17.1 for MX Series routers.

mac-vrf option introduced in Junos OS Release 20.4 for MX Series and vMX routers; QFX5100, QFX5110, QFX5120, QFX5200, QFX10002 (including QFX10002-60C), QFX10008, QFX10016, and EX9200 Series switches. Also introduced in Junos OS Evolved Release 21.2R1 on QFX5130-32CD switches.

Support for logical systems on MX Series routers added in Junos OS Release 17.4R1.

evpn-vpws option introduced in cRPD Release 20.3R1.

RELATED DOCUMENTATION

[Configuring EVPN Routing Instances | 10](#)

[Configuring EVPN Routing Instances on EX9200 Switches | 13](#)

[Configuring Virtual Router Routing Instances](#)

[Example: Configuring Filter-Based Forwarding on the Source Address](#)

[Example: Configuring Filter-Based Forwarding on Logical Systems](#)

instance-type mac-vrf

IN THIS SECTION

- [Syntax | 1593](#)
- [Hierarchy Level | 1593](#)
- [Description | 1593](#)
- [Options | 1594](#)
- [Required Privilege Level | 1594](#)
- [Release Information | 1594](#)

Syntax

```
instance-type mac-vrf;
```

Hierarchy Level

```
[edit logical-systems name routing-instances],  
[edit logical-systems name tenants name routing-instances],  
[edit routing-instances],  
[edit tenants name routing-instances]
```

Description

Specify a routing instance of type MAC-VRF. A MAC-VRF routing instance can have its own forwarding instance or be associated with the default forwarding instance of the device. A MAC-VRF routing instance is a type of virtual-switch routing instance. You can use a MAC-VRF routing instance in EVPN-VXLAN environments to create a MAC-based virtual routing and forwarding instance.

You can configure MAC-VRF routing instances on the following platforms:

- MX Series routers

- EX9200 line of switches
- QFX Series switches

Options

none	Create a routing instance of type MAC-VRF.
(bridge-domain vlans)	Specify the bridge domains or the VLANs associated with the MAC-VRF routing instance. You must specify VLANs in the MAC-VRF routing instance or specify a bridge domain that you configure elsewhere in the configuration. If you do not specify either, the error message: "Warning: bridge-domains must be configured for mac-vrf instance" is shown in the configuration editor and you cannot commit the configuration.
protocol evpn	Set the protocol for use in the MAC-VRF instance. In this case, you must select EVPN as the protocol.
service-type (vlan-aware vlan-based vlan-bundle)	Select the Ethernet VPN (EVPN) service type for this MAC-VRF instance. See "Understanding VLAN-Aware Bundle and VLAN-Based Service for EVPN" on page 497 .

Required Privilege Level

- routing—To view this statement in the configuration.
- routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 20.4R1.

RELATED DOCUMENTATION

| [MAC-VRF Routing Instance Type Overview](#) | 459

interconnect

IN THIS SECTION

- [Syntax | 1595](#)
- [Hierarchy Level | 1596](#)
- [Description | 1596](#)
- [Options | 1597](#)
- [Required Privilege Level | 1597](#)
- [Release Information | 1597](#)

Syntax

```
interconnect {  
    esi identifier {  
        all-active;  
        df-election-type {  
            mod;  
            preference {  
                value value (0..65535);  
            }  
        }  
    }  
}  
interconnected-vni-list (vni-list | all);  
route-distinguisher[number:id | ip-address:id];  
vrf-export (policy-name | [policy-names]);  
vrf-import (policy-name | [policy-names]);  
vrf-target {  
    community;  
    import community-name;  
    export community-name;  
}  
}
```

Hierarchy Level

```
[edit protocols evpn]
[edit routing-instances routing-instance-name protocols evpn]
```

Description

Enables interconnection of the following:

- EVPN-VXLAN points of delivery (PODs) in a data center.
- EVPN-VXLAN data centers. This use case is also known as data center interconnect (DCI).

In these use cases, a Juniper Networks device that supports the seamless EVPN-VXLAN stitching feature can serve as either a spine or super spine device that interconnects the PODs or data centers through an EVPN-VXLAN WAN network.

When configuring the interconnection, you can set up a single routing instance of type `virtual-switch` or `evpn` on each spine or super spine device. Or, you can use the default switching instance. In the instance, you specify these elements:

- Two sets of route distinguishers and route targets. The first set is for the POD or data center network. The second set, which you specify using the `interconnect` stanza of configuration statements, is for the WAN network.
- An Ethernet segment identifier (ESI) if a spine or super spine device is multihomed in `all-active` mode using an aggregated Ethernet interface.

NOTE: When configuring the ESI, keep in mind that the seamless EVPN-VXLAN stitching feature does not support `single-active` mode, and the `auto-derive lacp` and `source-bmac` options.

- All or a list of VXLAN network identifiers (VNIs) that the spine or super spine device translates when different VNIs are configured for the same VLAN in the PODs or data centers, and WAN. For more information about setting up VNI translation, see ["translation-vni" on page 1659](#) and ["interconnected-vni-list" on page 1597](#).

After you configure these elements, the EVPN control plane stitches the EVPN routes from the POD or data center network and from the WAN network into a single MAC forwarding table.

Options

The remaining statements are explained separately. Search for a statement in [CLI Explorer](#) or click a linked statement in the Syntax section for details.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 20.3R1.

interconnected-vni-list

IN THIS SECTION

- [Syntax | 1597](#)
- [Hierarchy Level | 1598](#)
- [Description | 1598](#)
- [Options | 1598](#)
- [Required Privilege Level | 1598](#)
- [Release Information | 1598](#)

Syntax

```
interconnected-vni-list (vni-list | all);
```


Hierarchy Level

```
[edit protocols evpn interconnect]  
[edit routing-instances routing-instance-name protocols evpn interconnect]
```

Description

Establishes that a Juniper Networks switch translates all VXLAN network identifiers (VNIs) or just a specified list of VNIs. This translation occurs when different VNIs are configured for the same VLAN in EVPN-VXLAN points of delivery (PODs) or data centers, and in the EVPN-VXLAN WAN network that interconnects them.

For information about specifying the VNI to which a Juniper Networks switch translates a VNI, see ["translation-vni" on page 1659](#).

Options

vni-list Translate only the specified VNIs with values that range from 1 through 16777214. For example, `interconnected-vni-list [10-50 60 70]`.

all Translate all VNIs.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 20.3R1.

interface (EVPN Routing Instances)

IN THIS SECTION

- [Syntax | 1599](#)
- [Hierarchy Level | 1599](#)
- [Description | 1599](#)
- [Options | 1600](#)
- [Required Privilege Level | 1600](#)
- [Release Information | 1600](#)

Syntax

```
interface interface-name {  
    ignore-encapsulation-mismatch;  
    interface-mac-limit limit {  
        packet-action drop;  
    }  
    mac-pinning  
    no-mac-learning;  
    protect-interface  
    static-mac mac-address;  
    vpws-service-id  
}
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols]  
[edit routing-instances routing-instance-name protocols evpn]
```

Description

Specify each interface over which the Ethernet VPN (EVPN) traffic travels between the PE device and customer edge (CE) device. The interfaces are bound to the EVPN routing instance.

Options

interface-name—Name of the interface.

The remaining statements are explained separately. See [CLI Explorer](#).

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 13.2.

Statement (mac-pinning) introduced in Junos OS Release 16.2 on MX Series routers.

vpws-service-id statement introduced in Junos OS Release 17.1 on MX Series routers.

Support for logical systems on MX Series routers added in Junos OS Release 17.4R1.

RELATED DOCUMENTATION

[Configuring EVPN Routing Instances | 10](#)

[Configuring EVPN Routing Instances on EX9200 Switches | 13](#)

[evpn | 1562](#)

[instance-type](#)

[vpws-service-id | 1665](#)

interface-mac-ip-limit

IN THIS SECTION

- [Syntax | 1601](#)
- [Hierarchy Level | 1601](#)
- [Description | 1601](#)

- Options | 1601
- Required Privilege Level | 1602
- Release Information | 1602

Syntax

```
interface-mac-ip-limit number;
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name vlans vlan-name
switch-options],
[edit logical-systems logical-system-name vlans vlan-name switch-options],
[edit routing-instances routing-instance-name protocols evpn],
[edit routing-instances routing-instance-name bridge-domains bridge-domain-name bridge-options],
[edit routing-instances routing-instance-name vlans vlan-name switch-options],
[edit vlans vlan-name switch-options]
```

Description

Limit the number of entries that can be learned on an interface through the MAC-IP bindings database. Depending on the Juniper Networks device, this limit can be applied to EVPN instances, bridge domains configured in a virtual-switch routing instance, or VLANs configured in a virtual-switch routing instance.

To apply this limit systemwide, use the `global-mac-ip-limit` statement at the `[edit protocols l2-learning]` hierarchy level.

Options

number Specify the maximum number of MAC-IP bindings per interface. After that maximum is reached, no additional MAC-IP entries are added to the database.

- **Range:** 1 through 1024.
- **Default:** 124

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 17.4R2.

RELATED DOCUMENTATION

[EVPN Proxy ARP and ARP Suppression, and Proxy NDP and NDP Suppression](#) | 361

[mac-ip-table-size](#) | 1619

interface-mac-limit (VPLS)

IN THIS SECTION

- [Syntax](#) | 1602
- [Hierarchy Level](#) | 1603
- [Description](#) | 1603
- [Options](#) | 1603
- [Required Privilege Level](#) | 1603
- [Release Information](#) | 1604

Syntax

```
interface-mac-limit limit {  
    packet-action drop;  
}
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols vpls
site site-name interfaces interface-name],
[edit routing-instances routing-instance-name protocols evpn],
[edit routing-instances routing-instance-name protocols evpn interface interface-name],
[edit routing-instances routing-instance-name protocols vpls site site-name interfaces interface-
name]
```

Description

Specify the maximum number of media access control (MAC) addresses that can be learned by the EVPN or VPLS routing instance. You can configure the same limit for all interfaces configured for a routing instance. You can also configure a limit for a specific interface.

Starting with Junos OS Release 12.3R4, if you do not configure the parameter to limit the number of MAC addresses to be learned by a VPLS instance, the default value is not effective. Instead, if you do not include the `interface-mac-limit` option at the `[edit logical-systems logical-system-name routing-instances routing-instance-name protocols vpls site site-name interfaces interface-name]`, hierarchy level, this setting is not present in the configuration with the default value of 1024 addresses. If you upgrade a router running a Junos OS release earlier than Release 12.3R4 to Release 12.3R4 or later, you must configure the `interface-mac-limit` option with a valid value for it to be saved in the configuration.

Options

limit—Number of MAC addresses that can be learned from each interface.

- **Range:** 1 through 131,071 MAC addresses

NOTE: For M120 devices only, the range is 16 through 65,536 MAC addresses.

- **Default:** 1024 addresses

The remaining statement is explained separately. Search for a statement in [CLI Explorer](#) or click a linked statement in the Syntax section for details.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced before Junos OS Release 7.4.

Support for EVPNs introduced in Junos OS Release 13.2 on MX 3D Series routers.

Support for EVPNs introduced in Junos OS Release 14.2 on EX Series switches.

RELATED DOCUMENTATION

[Configuring EVPN Routing Instances | 10](#)

[Configuring EVPN Routing Instances on EX9200 Switches | 13](#)

Limiting the Number of MAC Addresses Learned from an Interface

[interface \(EVPN Routing Instances\) | 1599](#)

mac-table-size

interface-state

IN THIS SECTION

- [Syntax | 1604](#)
- [Hierarchy Level | 1605](#)
- [Description | 1605](#)
- [Options | 1605](#)
- [Required Privilege Level | 1605](#)
- [Release Information | 1605](#)

Syntax

```
interface-state {
  local-interface local-interface;
```

```
hold-time {  
  down seconds;  
  up seconds;  
}  
}
```

Hierarchy Level

[edit *interfaces name unit num*]

Description

Consider another local interface's state changes when evaluating the status of the named interface.

Options

local-interface <i>local-interface</i>	The device includes the state of this local interface when evaluating the status (up or down) of the named interface. Specify <i>local-interface</i> by its logical interface name (for example, ae0.100).
hold-time (down up) <i>seconds</i>	Time to wait in seconds when the local-interface state changes before using that state to evaluate the status of the named interface.

Required Privilege Level

- interface—To view this statement in the configuration.
- interface-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 21.3R1.

RELATED DOCUMENTATION

ip-prefix-routes

IN THIS SECTION

- [Syntax | 1606](#)
- [Hierarchy Level | 1606](#)
- [Description | 1606](#)
- [Options | 1608](#)
- [Required Privilege Level | 1609](#)
- [Release Information | 1609](#)

Syntax

```
ip-prefix-routes {  
    advertise (direct-nexthop | gateway-address);  
    encapsulation (vxlan | mpls);  
    do-not-advertise-community;  
    <export routing-policy-name>;  
    gateway-interface interface-name;  
    vni number;  
}
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols evpn]  
[edit routing-instances routing-instance-name protocols evpn]
```

Description

In an Ethernet VPN (EVPN) environment, enable the device to advertise the IP prefix associated with a specified customer domain as a type-5 route to remote data centers. You might also enable IP prefix routes in a metro transport network. You use this feature when the Layer 2 domain does not exist at the remote data centers or metro network peering points.

We support two models for implementing type-5 routes:

- Pure type-5 route without overlay next hop and type-2 route
- Type-5 route with gateway IRB interface as overlay next hop and type-2 route

A pure type-5 route advertises the summary IP prefix and includes a BGP extended community called a router MAC. The router MAC carries the MAC address of the sending switch and provides next-hop reachability information for the prefix. In contrast, a standard type-5 route requires a gateway IP address as a next-hop overlay and a supporting type-2 route to provide recursive route resolution.

QFX5110 and QFX10000 switches currently support only fully resolved next-hop—that is, EVPN pure type-5—routes. QFX10000 switches support only EVPN Virtual Extensible LAN (VXLAN) with type-5 routes (not MPLS encapsulation).

On QFX5110, QFX5120, and EX4650 switches, when you configure pure type-5 routing in an overlay EVPN-VXLAN network, you must also configure the ["overlay-ecmp" on page 1638](#) statement at the [edit forwarding-options vxlan-routing] hierarchy level. Configuring the overlay-ecmp statement causes the Packet Forwarding Engine to restart, which interrupts all forwarding operations. As a result, we strongly recommend you put that configuration in place before the EVPN-VXLAN network becomes operational.

EX9200 switches support both type-5 routes but only pure type-5 routes with MPLS encapsulation. EX9200 switches support both MPLS and VXLAN encapsulation with the standard type-5 route that includes a gateway IP address as the next-hop overlay.



CAUTION: We introduced pure type-5 routing for EVPN-VXLAN in Junos OS Release 15.1X53-D30 for QFX10002 switches only. In that release, this statement is `ip-prefix-support forwarding-mode symmetric`. Starting with Junos OS Release 15.1X53-D60, the statement is `ip-prefix-routes advertise direct-nexthop`. When you upgrade to Junos OS Release 15.1X53-D60 or later, the upgrade process automatically updates any configuration with the original `ip-prefix-support` statement to the new `ip-prefix-routes` statement.

NOTE: Starting in Junos OS Release 15.1X53-D60, QFX10000 switches support pure type-5 routing.

By default, Juniper Networks devices that support pure type-5 routes don't advertise IP prefixes with a mask length of /32 as pure type-5 routes. To advertise /32 prefixes, you must set up a term in an export

policy that explicitly accepts these prefixes into the routing table. The following are excerpts from a sample configuration:

```
user@qfx10002# show policy-options policy-statement slash32
term 1 {
    from {
        protocol bgp;
        route-filter 10.44.44.44/31 orlonger;
    }
    then accept;
}
user@qfx10002# show routing-instances EVPN-A protocols evpn
ip-prefix-routes {
    advertise direct-nexthop;
    encapsulation vxlan;
    vni 5000;
    export slash32;
    ...
}
```

Options

The `advertise`, `encapsulation` and `vni number` options are required. The `export routing-policy-name` option is optional.

advertise direct-nexthop

Enable the switch to send IP prefix information using an EVPN pure type-5 route, which includes a router MAC extended community used to send the MAC address of the switch. This router MAC extended community provides next-hop reachability without requiring an overlay next-hop or supporting type-2 route.

NOTE: For pure route type-5, QFX5110 and QFX10000 switches support only VXLAN encapsulation, and EX9200 switches support only MPLS encapsulation.

advertise gateway-address (EX9200 switches only)

Enable the switch to advertise a gateway address in exported IP prefix routes. This gateway address provides overlay next-hop reachability.

NOTE: You must also specify a gateway address by including the `gateway-interface interface-name` statement.

**encapsulation
(vxlan | mpls)**

Specify to encapsulate forwarded traffic in VXLAN or MPLS for transmission to the remote data center.

NOTE: The same type of encapsulation must be used end to end. Only VXLAN encapsulation is supported on QFX10000 and QFX5110 switches.

**donot-advertise-
community**

Starting with Junos release 19.4R1, for when a type-5 routes is created from the type-2 MAC and IP advertisement that was learned from a remote PE device, the default behavior is to include the community information from the type-2 route (if any) in the type-5 route. You can prevent behavior this by enabling the `donot-advertise-community` option.

**export *routing-
policy-name***

(Optional) Specify the name of the routing policy configured at the `[edit policy-options policy-statement policy-statement-name]` hierarchy level to apply to the routes for the specified customer domain. Applying an export policy allows you to further control the IP prefixes to advertise or to suppress through EVPN type-5 routes for each customer. You can apply a separate export routing policy to one or more customer domains. This allows each customer to each have its own policy.

**gateway-interface
*interface-name***

Specify the gateway interface to use as a next-hop overlay for a standard type-5 route. You must use this option in conjunction with the `advertise gateway-address` option.

vni *number*

Specify the identifier associated with a customer domain. Each customer domain must have a unique identifier.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 15.1X53-D60 and Junos OS Release 17.2R1.

Support for logical systems on MX Series routers added in Junos OS Release 17.4R1.

RELATED DOCUMENTATION

[Understanding EVPN Pure Type 5 Routes | 22](#)

[EVPN Overview for Switches | 878](#)

[policy-statement](#)

ip-prefix-support

IN THIS SECTION

- [Syntax | 1610](#)
- [Hierarchy Level | 1610](#)
- [Description | 1611](#)
- [Options | 1611](#)
- [Required Privilege Level | 1612](#)
- [Release Information | 1612](#)

Syntax

```
ip-prefix-support {  
    encapsulation vxlan;  
    <export routing-policy-name>;  
    forwarding-mode symmetric;  
    vni number;  
}
```

Hierarchy Level

```
[edit routing-instances routing-instance-name protocols evpn]
```

Description

In an Ethernet VPN (EVPN) Virtual Extensible LAN (VXLAN) environment, enable the provider edge (PE) switch to forward the IP prefix associated with a specified customer domain in scenarios where the Layer 2 domain does not extend across data centers. Such routes are referred to as EVPN pure type-5 routes. On QFX switches, only fully resolved next-hop routes are supported. The data packets are sent as Layer 2 frames that are encapsulated in the VXLAN header. A unique VXLAN numerical identifier is associated with each customer domain.

When the advertisement of the type-5 route is enabled, you can optionally use an export routing policy configured at the `[edit policy-options policy-statement policy-name]` hierarchy level to further control the IP prefixes to advertise or suppress through EVPN type-5 routes for each customer. The policy is applied to the routes that reside in the customer's inet.0 routing table by specifying a *routing-instance-name* configured at the `[edit routing-instances]` hierarchy level. This *routing-instance-name* refers to a Layer 3 virtual routing and forwarding (VRF) domain for a specific customer. The corresponding name of the table is *instance.name.inet.0*. This table is created by default when you configure a routing instance.



CAUTION: This statement is deprecated starting with Junos OS Release 15.1x53-D60 and has been replaced with [ip-prefix-routes](#). The statement `ip-prefix-support forwarding-mode symmetric` is now `ip-prefix-routes advertise direct-nexthop`. Any configuration with the original `ip-prefix-support` statement is automatically upgraded to the new `ip-prefix-routes` statement when you upgrade to Junos OS Release 15.1x53-D60 or later.

Options

The forwarding-mode `symmetric`, encapsulation `vxlan` and vni *number* options are required. The export *policy-name* option is optional.

encapsulation vxlan	Specify to encapsulate forwarded traffic in VXLAN for transmission to the other data center.
export <i>routing-policy-name</i>	(Optional) Specify the name of the routing policy configured at the <code>[edit policy-options policy-statement <i>policy-name</i>]</code> hierarchy level to apply to the routes for the specified customer domain. Applying an export policy allows you to further control the IP prefixes to advertise or to suppress for each customer. You can apply a separate export routing policy to one or more customer domains. This allows each customer to each have its own policy.
forwarding-mode symmetric	Enable the receiver to resolve the next-hop route using only information carried in the type-5 route.

vni *number* Specify the identifier associated with a customer domain. Each customer domain must have a unique identifier.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 15.1x53-D30.

RELATED DOCUMENTATION

[Understanding EVPN Pure Type 5 Routes | 22](#)

[policy-statement](#)

label-allocation

IN THIS SECTION

- [Syntax | 1612](#)
- [Hierarchy Level | 1613](#)
- [Description | 1613](#)
- [Options | 1613](#)
- [Required Privilege Level | 1613](#)
- [Release Information | 1613](#)

Syntax

```
label-allocation per-instance | per-bridge-domain;
```

Hierarchy Level

```
[edit routing-instances routing-instance-name protocols evpn]
```

Description

Specifies the MPLS label allocation setting for the EVPN routing instance.

Options

`per-instance`—Allocates a single MPLS label for the EVPN routing instance.

`per-bridge-domain`—Allocates a single MPLS label for each bridge domain. By default, `per-bridge-domain` is enabled on the PTX10008 Packet Transport Routers.

Required Privilege Level

`routing`—To view this statement in the configuration.

`routing-control`—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 13.2.

RELATED DOCUMENTATION

[Configuring EVPN Routing Instances](#) | 10

leaf

IN THIS SECTION

- [Syntax](#) | 1614
- [Hierarchy Level](#) | 1614

- [Description | 1614](#)
- [Options | 1615](#)
- [Required Privilege Level | 1615](#)
- [Release Information | 1615](#)

Syntax

```
leaf {
    replicator-activation-delay seconds;
}
```

Hierarchy Level

```
[edit logical-systems name routing-instances name protocols evpn assisted-replication],
[edit routing-instances name protocols evpn assisted-replication],
[edit protocols evpn assisted-replication]
```

Description

Configure a device in an EVPN network into the assisted replication (AR) leaf role. AR leaf devices might also be called AR client devices.

AR helps to optimize broadcast, unknown unicast, and multicast (BUM) traffic in EVPN networks. To enable AR, you configure devices in the EVPN network to operate as AR replicator and AR leaf devices.

AR replicator devices help perform BUM traffic replication and forwarding tasks for AR leaf devices. When an AR leaf device receives BUM traffic, instead of always using ingress replication to the EVPN core, it forwards the traffic on an AR tunnel to an AR replicator device that can better handle the replication load. The AR replicator device then replicates and forwards the traffic to other overlay tunnels. You configure a loopback interface IP address on each AR replicator device for the AR tunnel.

You can configure multiple AR replicator devices in an EVPN network, and AR leaf devices load-balance among the available AR replicator devices as follows:

- AR leaf devices in the QFX5000 line of switches automatically designate a particular AR replicator device for a VLAN or VXLAN network identifier (VNI) to load-balance among the available AR replicators.

- AR leaf devices in the QFX10000 line of switches actively load-balance among the available AR replicators based on traffic flow levels within a VLAN or VNI.

Options

replicator-activation-delay

Interval (in seconds) to wait after receiving an AR replicator route from an AR replicator before starting to forward traffic to it. You might want to adjust the delay to make sure AR replicator devices have fully learned the current EVPN state from the network after an AR replicator goes down and comes back up again.

- **Default:** 10 seconds
- **Range:** 0 through 180 seconds

Required Privilege Level

routing

Release Information

Statement introduced in Junos OS Release 18.4R2 and 19.4R1.

RELATED DOCUMENTATION

[Assisted Replication Multicast Optimization in EVPN Networks | 752](#)

[replicator | 1649](#)

[show evpn multicast-snooping assisted-replication next-hops | 1880](#)

[show evpn multicast-snooping assisted-replication multihomed-peers | 1876](#)

[show evpn multicast-snooping assisted-replication replicators | 1883](#)

loop-detect (EVPN)

IN THIS SECTION

● [Syntax | 1616](#)

- [Hierarchy Level | 1616](#)
- [Description | 1616](#)
- [Options | 1617](#)
- [Required Privilege Level | 1618](#)
- [Release Information | 1618](#)

Syntax

```
loop-detect {  
  enhanced {  
    interface name {  
      loop-detect-action (interface-down | laser-off);  
      revert-interval seconds;  
      transmit-interval (10m | 10s | 1m | 1s);  
      vlan-id vlan-id;  
    }  
  }  
}
```

Hierarchy Level

```
[edit logical-systems name protocols],  
[edit protocols]
```

Description

Configure loop detection on the server-facing Layer 2 interfaces of the leaf devices in an EVPN-VXLAN fabric. This feature can detect the following types of Ethernet loops:

- A loop between two interfaces in different Ethernet segments (ESs). This loop is typically caused by miswiring fabric components.
- A loop between two interfaces with the same Ethernet segment identifier (ESI). This loop is typically caused by miswiring a third-party switch to the fabric.

After you've enabled loop detection, the interfaces periodically send multicast loop-detection protocol data units (PDUs). If a loop detection-enabled interface receives a PDU, it detects a loop and triggers the configured action to break the loop. For example, if you configured the interface-down action, the device brings the interface down. After you have repaired the loop condition and the revert-interval timer expires, the device reverts the loop-detection action and brings the interface up again.

If needed, you can revert the status on one or all interfaces where a loop was detected, either before the revert-interval timer expires or if you didn't configure revert-interval. To do so, use the following command:

```
clear loop-detect enhanced interface [interface-name]
```

If you don't specify an interface name, the clear command reverts the interface status for any interfaces where the device performed a loop detection action.

Use the following command to see loop detection status information for an interface or all interfaces with loop detection enabled:

```
show loop-detect enhanced interface [interface-name]
```

Options

**interface
name** Specify the name of the server-facing Layer 2 interface on which you are enabling loop detection.

loop-detect-action If a loop is detected, you can configure the following action:

- Values:
 - interface-down—The device brings down the interface on which the loop was detected.
 - laser-off—The device turns off the optics transmit laser.

NOTE: We do not support the laser-off action on EX4300-48MP, QFX51XX and QFX52XX switches.

revert-interval Specify the interval, in seconds, in which the interface will revert to its previous state after the loop condition is fixed. For example, if the loop-detect-action is interface-down, after

the loop condition is repaired and the interval expires, the device brings the interface up again.

- **Default:** 0

NOTE: If you retain this default setting (0) and a loop detection event triggers the interface-down action, you must enter the `clear loop-detect enhanced interface <interface-name>` command to bring the interface back up. Until you issue this command, the interface will remain down.

- **Range:** 0 through 300 seconds

transmit-interval

The interval in seconds or minutes at which the interface transmits loop detection PDUs.

- **Default:** 1 second
- **Values:**
 - 10 minutes
 - 10 seconds
 - 1 minute
 - 1 second

vlan-id vlan-id

(Required for trunk interfaces or interfaces with enterprise style configuration) Specify the VLAN identifier for the interface.

- **Range:** 1 through 4094

Required Privilege Level

routing

Release Information

Statement introduced in Junos OS Release 20.4R1.

RELATED DOCUMENTATION

[clear loop-detect enhanced interface](#) | 1734

[show loop-detect enhanced interface](#) | [1911](#)

mac-ip-table-size

IN THIS SECTION

- [Syntax](#) | [1619](#)
- [Hierarchy Level](#) | [1619](#)
- [Description](#) | [1619](#)
- [Options](#) | [1620](#)
- [Required Privilege Level](#) | [1620](#)
- [Release Information](#) | [1620](#)

Syntax

```
mac-ip-table-size number;
```

Hierarchy Level

```
[edit logical-systems name routing-instances routing-instance-name protocols evpn],  
[edit routing-instances routing-instance-name protocols evpn],  
[edit routing-instances routing-instance-name bridge-domains bridge-domain-name bridge-options],  
[edit routing-instances routing-instance-name vlans vlan-name switch-options]
```

Description

Limit the number of entries that can be added to the MAC-IP address bindings database. Depending on the Juniper Networks device, this limit can be applied to EVPN routing instances, bridge domains configured in a virtual-switch routing instance, or VLANs configured in a virtual-switch routing instance. When the specified maximum number of entries is reached, no additional entries are added to the MAC-IP bindings database.

To apply this limit systemwide, use the [global-mac-ip-limit](#) statement at the [edit protocols l2-learning] hierarchy level.

Options

number Maximum number of entries that can be added to the MAC-IP address bindings database.

- **Range:** 16 through 1,048,575.
- **Default:** 8192

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 17.4R2.

Statement introduced in Junos OS Release 18.1R1 for QFX Series switches.

RELATED DOCUMENTATION

[EVPN Proxy ARP and ARP Suppression, and Proxy NDP and NDP Suppression](#) | 361

[interface-mac-ip-limit](#) | 1600

mac-pinning (EVPN Routing Instances)

IN THIS SECTION

- [Syntax](#) | 1621
- [Hierarchy Level](#) | 1621
- [Description](#) | 1621
- [Required Privilege Level](#) | 1621

Syntax

```
mac-pinning;
```

Hierarchy Level

```
[edit routing-instances routing-instance-name protocols protocol interface interface-name]  
[edit logical-systems logical-system-name routing-instances routing-instance-name vlans vlan-name  
bridge-domains bridge-name bridge-options-interface interface-name]  
[edit logical-systems logical-system-name routing-instances routing-instance-name vlans vlan-name  
routing-instances routing-instance-name switch-options interface interface-name]
```

Description

Sets mac-pinning on the interface, which makes it static. That is, the MAC-pinned address cannot be moved to any other interface in the bridge domain.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 16.2.

RELATED DOCUMENTATION

[EVPN MAC Pinning Overview | 377](#)

[Configuring EVPN MAC Pinning | 379](#)

[Configuring MAC Pinning for PBB-EVPN](#)

Understanding Layer 2 Learning and Forwarding for Bridge Domains

Understanding Layer 2 Learning and Forwarding for Bridge Domains Functioning as Switches with Layer 2 Trunk Ports

mclag

IN THIS SECTION

- Syntax | 1622
- Hierarchy Level | 1622
- Description | 1622
- Required Privilege Level | 1623
- Release Information | 1623

Syntax

```
mclag {  
    bgp-peer ip-address;  
}
```

Hierarchy Level

```
[edit routing-instances name protocols evpn]
```

Description

Configure parameters that enable the interworking of Ethernet VPN-MPLS (EVPN-MPLS) with a Junos Fusion Enterprise or a multichassis link aggregation group (MC-LAG) topology.

The remaining statements are explained separately. See [CLI Explorer](#).

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 17.4R1.

RELATED DOCUMENTATION

[Understanding EVPN-MPLS Interworking with Junos Fusion Enterprise and MC-LAG](#) | 1359

multicast-mode (EVPN)

IN THIS SECTION

- [Syntax](#) | 1623
- [Hierarchy Level](#) | 1624
- [Description](#) | 1624
- [Options](#) | 1624
- [Required Privilege Level](#) | 1624
- [Release Information](#) | 1624

Syntax

```
multicast-mode client | ingress-replication;
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols],
[edit protocols evpn],
[edit routing-instances routing-instance-name protocols evpn]
```

Description

Configure the multicast server mode for delivering traffic and packets for Ethernet VPN (EVPN). This statement is required for a VXLAN EVPN instance.

NOTE: If you configure the `multicast-mode` statement, then you must also configure the `encapsulation vxlan` statement.

Options

client	Use the client as the multicast mode for delivering traffic and multicast packets across routers and switches.
ingress-replication	Use ingress replication as the multicast mode for delivering broadcast, unknown unicast, and multicast (BUM) traffic and multicast packets across routers and switches.

- **Default:** ingress-replication

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 14.1X53-D30.

Support for logical systems on MX Series routers added in Junos OS Release 17.4R1.

RELATED DOCUMENTATION

[Understanding EVPN with VXLAN Data Plane Encapsulation | 398](#)

[EVPN Over VXLAN Encapsulation Configuration Overview for QFX Series and EX4600 Switches | 430](#)

[Using a Default Layer 3 Gateway to Route Traffic in an EVPN-VXLAN Overlay Network | 508](#)

[Example: Configure an EVPN-VXLAN Centrally-Routed Bridging \(CRB\) Using MX Routers as Spines | 580](#)

multicast-replication

IN THIS SECTION

- [Syntax | 1625](#)
- [Hierarchy Level | 1626](#)
- [Description | 1626](#)
- [Default | 1626](#)
- [Options | 1626](#)
- [Required Privilege Level | 1627](#)
- [Release Information | 1628](#)

Syntax

```
multicast-replication {  
    evpn {  
        irb (local-only | local-remote | oism);  
        smet-nexthop-limit smet-nexthop-limit;  
    }  
    ingress;  
    local-latency-fairness;  
}
```

Hierarchy Level

```
[edit forwarding-options]
```

Description

Configure the mode of multicast replication that helps to optimize multicast latency.

NOTE: The multicast-replication statement is supported only on platforms with the enhanced-ip mode enabled.

Default

This statement is disabled by default.

Options

NOTE: The ingress and the local-latency-fairness options do not apply to EVPN configurations.

ingress	Complete ingress replication of the multicast data packets where all the egress Packet Forwarding Engines receive packets from the ingress Packet Forwarding Engines directly.
local-latency-fairness	Complete parallel replication of the multicast data packets.
evpn irb local-only local-remote oism	<div>Enable IPv4 inter-VLAN multicast forwarding in an EVPN-VXLAN network in one of these modes:</div> <ul style="list-style-type: none">• evpn irb local-only—Use this mode with a collapsed IP fabric, also known as an <i>edge-routed bridging overlay</i>. In this mode, the PFE on the leaf devices in the fabric performs local multicast routing at the fabric edge. The spine devices, also called <i>lean spines</i>, primarily act as IP transit devices for the fabric.• evpn irb local-remote—Use this mode with a two-layer IP fabric, also known as a <i>centrally-routed bridging overlay</i>. In this mode, the spine devices in the fabric centrally route the multicast traffic between VLANs. The spine devices forward the

routed VLAN traffic into the EVPN core toward interested receivers. The spine devices use a PIM designated router (DR) to avoid duplicating packets into the core. The leaf devices forward multicast traffic received on a VLAN to their receivers on that VLAN.

- `evpn irb oism`—Use this mode with a collapsed IP fabric (edge-routed bridging overlay) to also support routing multicast traffic between the fabric and external devices. This mode implements *optimized inter-subnet multicast* (OISM) according to the IETF draft <https://tools.ietf.org/html/draft-ietf-bess-evpn-irb-mcast-04>. In this mode, the leaf devices operate in local-remote mode like in a centrally-routed bridging overlay design. At the same time, the device performs local multicast routing.

OISM also supports:

- Selective forwarding on the source VLAN.
- Routing multicast traffic through an external PIM domain.

Default mode: `evpn irb local-remote`

You can enable only one of these modes at a time.

NOTE: Only `local-remote` and `oism` modes support selective multicast (SMET) forwarding.

`smet-nexthop-limit smet-nexthop-limit`

Configures a limit for the number of SMET next hops for selective multicast forwarding. A PE device uses the SMET next hops list of outgoing interfaces to selectively replicate and forward multicast traffic. When the list of SMET next hops reaches this limit, the PE device stops adding new SMET next hops. At that point, the PE device sends new multicast group traffic to all egress devices.

- **Range:** 10,000 through 40,000
- **Default:** 10,000

Required Privilege Level

`interface`—To view this statement in the configuration.

`interface-control`—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 15.1.

evpn stanza introduced in Junos OS Release 17.3R3 for QFX Series switches.

oism option in evpn irb stanza introduced in Junos OS Release 21.2R1.

RELATED DOCUMENTATION

forwarding-options

[IPv4 Inter-VLAN Multicast Forwarding Modes for EVPN-VXLAN Overlay Networks | 668](#)

[Optimized Inter-Subnet Multicast in EVPN Networks | 773](#)

multicast-snooping-options

IN THIS SECTION

- [Syntax | 1628](#)
- [Hierarchy Level | 1629](#)
- [Description | 1629](#)
- [Options | 1629](#)
- [Required Privilege Level | 1629](#)
- [Release Information | 1629](#)

Syntax

```
multicast-snooping-options {
  flood-groups [ ip-addresses ];
  forwarding-cache {
    threshold suppress value <reuse value>;
  }
  host-outbound-traffic (Multicast Snooping) {
    forwarding-class class-name;
```

```

        dot1p number;
    }
    graceful-restart <restart-duration seconds>;
    ignore-stp-topology-change;
    multichassis-lag-replicate-state;
    nexthop-hold-time milliseconds;
    oism {
        install-star-g-routes;
    }
    traceoptions {
        file filename <files number> <size size> <world-readable | no-world-readable>;
        flag flag <flag-modifier> <disable>;
    }
}

```

Hierarchy Level

```

[edit logical-systems logical-system-name routing-instances routing-instance-name],
[edit routing-instances routing-instance-name],

```

Description

Establish multicast snooping option values.

Options

The remaining statements are explained separately. See [CLI Explorer](#).

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 8.5.

RELATED DOCUMENTATION

Configuring Multicast Snooping

Enabling Bulk Updates for Multicast Snooping

Example: Configuring Multicast Snooping

no-arp-suppression

IN THIS SECTION

- [Syntax | 1630](#)
- [Hierarchy Level | 1630](#)
- [Description | 1631](#)
- [Required Privilege Level | 1631](#)
- [Release Information | 1631](#)

Syntax

```
no-arp-suppression;
```

Hierarchy Level

The instance type 'EVPN' supports under the hierarchy:

```
[edit bridge-domains bridge-domain-name]
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols]
[edit routing-instances instance-name protocols evpn]
[edit routing-instances instance-name bridge-domains domain-name]
```

The instance type 'virtual-switch' supports under hierarchy:

```
[edit logical-systems logical-system-name routing-instances routing-instance-name vlans vlan-name]
```

```
[edit routing-instances routing-instance-name vlans vlan-name]
[edit vlans vlan-name]
```

Description

Disable the suppression of Address Resolution Protocol (ARP) requests from a customer edge (CE) device or Layer 2 VXLAN gateway to a provider edge (PE) device or Layer 3 VXLAN gateway that acts as ARP proxy in an Ethernet VPN-MPLS (EVPN-MPLS) or Ethernet VPN-Virtual Extensible LAN (EVPN-VXLAN) environment.

When disabling ARP suppression, be aware of the following implications;

- If the PE device or Layer 3 VXLAN gateway does not find the MAC-IP address binding in its database, it cannot forward the ARP request. Instead, the device discards the ARP request, and the packets flood through the Layer 2 domain because the ARP request is considered to be Layer 2 broadcast traffic.
- Disabling the suppression of ARP packets on a PE device or Layer 3 VXLAN gateway essentially disables the proxy ARP functionality.

Therefore, we recommend that ARP suppression remains enabled.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 17.2R1 .

Support for logical systems on MX Series routers added in Junos OS Release 17.4R1.

RELATED DOCUMENTATION

| [EVPN Proxy ARP and ARP Suppression, and Proxy NDP and NDP Suppression](#) | 361

no-default-gateway-ext-comm

IN THIS SECTION

- [Syntax | 1632](#)
- [Hierarchy Level | 1632](#)
- [Description | 1632](#)
- [Required Privilege Level | 1632](#)
- [Release Information | 1632](#)

Syntax

```
no-default-gateway-ext-comm;
```

Hierarchy Level

```
[edit routing-instances name protocols evpn]
```

Description

Configure a routing instance to suppress the advertisement of the extended community on the default gateway. An extended community is an 8-octet community used by Ethernet VPNs (EVPNs).

Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 16.1.

RELATED DOCUMENTATION

[Implementing EVPN-VXLAN for Data Centers | 432](#)

[EVPN Multihoming Overview | 96](#)

nsr-phantom-holdtime

IN THIS SECTION

- [Syntax | 1633](#)
- [Hierarchy Level | 1633](#)
- [Description | 1633](#)
- [Options | 1634](#)
- [Required Privilege Level | 1634](#)
- [Release Information | 1634](#)

Syntax

```
nsr-phantom-holdtime seconds;
```

Hierarchy Level

```
[edit routing-options]
```

Description

Specify to hold phantom IP addresses, that is, prevent these routes from being added to routing tables, for a specified period of time. During this hold-time interval, these routes are maintained in the kernel. After the hold time expires, the routes are added to the routing tables.

We strongly recommend that you configure this statement before you perform a graceful Routing Engine switchover (GRES) when nonstop routing (NSR) is enabled. Doing so prevents traffic loss because

routes are added to the routing tables after the hold-time interval expires, but before they are deleted from the kernel during a switchover.

Options

seconds Specify the interval of time to prevent phantom IP addresses from being added to the routing tables. After this interval expires, the routes are added to the routing tables.

BEST PRACTICE: In an EVPN/VXLAN environment with NSR and GRES enabled, the recommended hold-time value is 300 (5 minutes)

- **Range:** 0 through 10000

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 15.1x53-D60.

RELATED DOCUMENTATION

[Understanding Graceful Routing Engine Switchover](#)

[Preventing Traffic Loss in an EVPN/VXLAN Environment With GRES and NSR | 386](#)

[Example: Configuring Nonstop Active Routing on Switches](#)

oism

IN THIS SECTION

● [Syntax | 1635](#)

- Hierarchy Level | 1635
- Description | 1635
- Options | 1635
- Required Privilege Level | 1636
- Release Information | 1636

Syntax

```
oism {
    originate-smet-on-revenue-vlan-too;
    pim-evpn-gateway {
        external-irb name;
    }
    supplemental-bridge-domain-irb irb-interface-name;
}
```

Hierarchy Level

```
[edit routing-instances name protocols evpn]
```

Description

Set optimized inter-subnet multicast (OISM) options in a Layer 3 virtual routing and forwarding (VRF) routing instance on an EVPN fabric device.

Options

originate-smet-on-revenue-vlan-too Originate selective multicast Ethernet tag (SMET) routes on revenue VLANs (or bridge domains) upon receiving a local IGMP report. SMET routes are EVPN Type 6 routes. The revenue VLANs are the customer VLANs that serve the hosts on the access side.

By default, when an OISM device receives a local IGMP join request, it originates a Type 6 route on the OISM supplemental bridge domain (SBD). With this option, the device also sends Type 6 routes on the revenue VLANs.

<code>pim-evpn-gateway</code> <code>external-irb <i>name</i></code>	Configure an IRB interface to act as a PIM EVPN gateway (PEG) on a device where you have enabled OISM. The device uses the PEG interface to route multicast traffic to and from an external PIM domain.
<code>supplemental-bridge-domain-irb <i>irb-interface-name</i></code>	Associate an IRB interface with the OISM SBD on a device where you have enabled OISM.

See [CLI Explorer](#) for all options.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 21.2R1.

RELATED DOCUMENTATION

- [IPv4 Inter-VLAN Multicast Forwarding Modes for EVPN-VXLAN Overlay Networks | 668](#)
- [Optimized Inter-Subnet Multicast in EVPN Networks | 773](#)

oism (Multicast Snooping Options)

IN THIS SECTION

- [Syntax | 1637](#)
- [Hierarchy Level | 1637](#)
- [Description | 1637](#)

- Options | 1637
- Required Privilege Level | 1638
- Release Information | 1638

Syntax

```
oism {  
    install-star-g-routes;  
}
```

Hierarchy Level

[edit [multicast-snooping-options](#)]

Description

Set optimized inter-subnet multicast (OISM) options. You can use OISM to optimize multicast traffic flow in an EVPN-VXLAN fabric.

Options

- | | |
|------------------------------|--|
| install-star-g-routes | <p>Set the Routing Engine to immediately install (*,G) multicast routes (upon receiving an EVPN Type 6 route) on the Packet Forwarding Engine for all of the revenue VLANs in the routing instance.</p> <p>Without this option, the device prioritizes saving resources on the Packet Forwarding Engine by not installing multicast routes until the multicast traffic arrives. The default behavior avoids issues with scaling on lower-end devices.</p> <p>If you configure this option, the device ensures the Packet Forwarding Engine has multicast routing information available before any multicast traffic arrives. This option minimizes packet loss at the onset of a multicast flow. However, the device consumes extra Packet</p> |
|------------------------------|--|

Forwarding Engine resources for routes that are not even being used yet (and might never be used). As a result, when you configure this option, you trade off taking up the extra Packet Forwarding Engine resources to improve network latency.

You set this option globally in the default-switch routing instance. The option acts only on VLANs associated with a Layer 3 virtual routing and forwarding (VRF) instance with OISM enabled.

NOTE: Always configure this option with OISM on switches in the QFX10000 line. These switches easily operate at scale and multicast traffic flows well with this option set. Without this option, QFX10000 switches have an issue with mesh group handling and don't properly install the multicast routes. We don't recommend setting this option with OISM on switches in the QFX5000 line. Consider this option on those switches only if you have very stringent latency requirements and can trade those off latency against supporting higher scaling.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 21.2R1.

RELATED DOCUMENTATION

[Optimized Inter-Subnet Multicast in EVPN Networks](#) | 773

overlay-ecmp

IN THIS SECTION

● [Syntax](#) | 1639

- Hierarchy Level | 1639
- Description | 1639
- Required Privilege Level | 1639
- Release Information | 1640

Syntax

```
overlay-ecmp;
```

Hierarchy Level

```
[edit forwarding-options vxlan-routing ]
```

Description

Enable two-level equal-cost multipath next hops.

On QFX5110, QFX5120, and EX4650 switches, you must configure this statement for Layer 3 Ethernet VPN Virtual Extensible LAN (EVPN-VXLAN) overlay networks when you configure pure type-5 routing. For more information about configuring pure type-5 routes, see ["ip-prefix-routes" on page 1606](#).

NOTE: Configuring this statement causes the Packet Forwarding Engine to restart. Restarting the Packet Forwarding Engine interrupts all forwarding operations. Therefore, we strongly recommend using this configuration statement before the EVPN-VXLAN network becomes operational.

Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 17.4R1.

RELATED DOCUMENTATION

| [vxlan-routing](#) | [1718](#)

pbb-evpn-core

IN THIS SECTION

- [Syntax](#) | [1640](#)
- [Hierarchy Level](#) | [1640](#)
- [Description](#) | [1640](#)
- [Required Privilege Level](#) | [1641](#)
- [Release Information](#) | [1641](#)

Syntax

```
pbb-evpn-core;
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols]  
[edit routing-instances routing-instance-name protocol evpn]
```

Description

Specify that Ethernet VPN (EVPN) is running for Provider Backbone Bridging (PBB).

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 16.1.

Support for logical systems on MX Series routers added in Junos OS Release 17.4R1.

RELATED DOCUMENTATION

| [evpn](#) | [1562](#)

per-esi

IN THIS SECTION

- [Syntax](#) | [1641](#)
- [Hierarchy Level](#) | [1642](#)
- [Description](#) | [1642](#)
- [Options](#) | [1642](#)
- [Required Privilege Level](#) | [1643](#)
- [Release Information](#) | [1643](#)

Syntax

```
per-esi {  
    lacp-oos-on-ndf;  
}
```

Hierarchy Level

```
[edit interfaces name esi df-election-granularity]
```

Description

Configure an aggregated Ethernet interface to be the designated forwarder (DF). Doing this ensures that the DF role of the ESI represents the role of the aggregated Ethernet interface.

NOTE: You cannot enable the `sync-reset` statement at the `[edit interfaces name aggregated-ether-options lacp]` level when you have the `lacp-oos-on-ndf` statement enabled.

NOTE: You cannot enable the `lacp-oos-on-ndf` statement in the following situations:

- When you have configured an all-active ESI.
- When the interface is not an aggregated Ethernet interface.
- When the aggregated Ethernet interface does not have LACP enabled.

The `per-esi` and `per-esi-vlan` statements are exclusive and cannot be configured at the same time. If you configure `per-esi-vlan` statement on an interface that is already configured with the `per-esi`, then the `per-esi` is removed from the configuration. Also, the opposite is true. If you configure `per-esi` statement on an interface that is already configured with the `per-esi-vlan`, then the `per-esi-vlan` is removed from the configuration.

Options

lacp-oos-on-ndf In an EVPN single-active configuration, the provider edge (PE) device that is the non-DF will send LACP out-of-sync packets to the customer edge device (CE). As a result, LACP goes down on the CE device, and the CE device does not use the links connected to the non-DF for sending traffic. If the connection between a CE device and a DF PE device fails, the PE device is re-elected as a DF. If the connection between a CE device and a non-DF PE device fails, the current DF PE device is not changed.

The remaining statements are explained separately. Search for a statement in [CLI Explorer](#) or click a linked statement in the Syntax section for details.

Required Privilege Level

interface

Release Information

Statement introduced in Junos OS Release 20.4R1

per-esi-vlan

IN THIS SECTION

- [Syntax | 1643](#)
- [Hierarchy Level | 1643](#)
- [Description | 1643](#)
- [Required Privilege Level | 1644](#)
- [Release Information | 1644](#)

Syntax

```
per-esi-vlan;
```

Hierarchy Level

```
[edit interfaces name esi df-election-granularity]
```

Description

Configure an aggregated Ethernet interface to have different designated forwarder (DF) election roles. This is the default behavior.

Required Privilege Level

interface

Release Information

Statement introduced in Junos OS Release 20.4R1

proxy-mac

IN THIS SECTION

- [Syntax | 1644](#)
- [Hierarchy Level | 1644](#)
- [Description | 1645](#)
- [Options | 1645](#)
- [Required Privilege Level | 1645](#)
- [Release Information | 1645](#)

Syntax

```
proxy-mac {  
    irb |proxy-mac-address;  
}
```

Hierarchy Level

```
[edit routing-instances name protocols evpn],  
[edit routing-instances name bridge-domains]
```

Description

Specifies the proxy MAC address.

When CE devices connect directly to gateway devices, configure the MAC proxy on the gateway devices with the `irb` option. Junos OS will use the virtual gateway MAC address or the MAC address of the IRB interface.

In a spine-and-leaf network where the routing is performed on the spine devices, configure the MAC proxy on the leaf devices to use the actual MAC address of the IRB interface.

Options

irb	Reply with the manually configured virtual gateway, automatically derived virtual gateway, or IRB MAC address. The virtual gateway MAC address takes precedence over a IRB MAC address. If the MAC address of the IRB interface is not configured, Junos OS uses a MAC address derived from the physical interface.
proxy-mac-address	Reply with the configured proxy MAC address for all requests. A configured proxy MAC address should correspond to the virtual or physical gateway MAC address that is deployed on centralized Layer 3 gateway IRB interfaces.

Required Privilege Level

routing

Release Information

Statement introduced in Junos OS Release 19.1R1.

RELATED DOCUMENTATION

| [ARP and NDP Request with a proxy MAC address](#) | 364

proxy-macip-advertisement

IN THIS SECTION

- [Syntax | 1646](#)
- [Hierarchy Level | 1646](#)
- [Description | 1646](#)
- [Required Privilege Level | 1647](#)
- [Release Information | 1647](#)

Syntax

```
proxy-macip-advertisement;
```

Hierarchy Level

```
[edit interfaces irb unit logical-unit-number ]
```

Description

Enable the proxy advertisement feature on a QFX Series switch that can function as a Layer 3 gateway. With this feature enabled, the Layer 3 gateway advertises the MAC and IP routes (MAC+IP type 2 routes) on behalf of Layer 2 VXLAN gateways.

In an Ethernet VPN-Virtual Extensible LAN (EVPN-VXLAN) centrally-routed bridging overlay (EVPN-VXLAN topology with a two-layer IP fabric), spine devices typically function as Layer 3 VXLAN gateways, and leaf devices typically function as Layer 2 gateways. In this overlay network, the Layer 2 VXLAN gateways can advertise only the MAC routes (EVPN type 2 routes) for the attached hosts. Since the Layer 2 gateways are unable to resolve the MAC-to-IP bindings for the hosts, each of the Layer 3 gateways rely on the Address Resolution Protocol (ARP) and the Neighbor Discovery Protocol (NDP) to discover and install the bindings.

For example, after a Layer 3 gateway receives a host MAC route advertisement from a Layer 2 gateway, and ARP and NDP resolve the MAC-to-IP bindings, the Layer 3 gateway in turn advertises the host MAC and IP routes along with the next hop, which is set to the Layer 2 gateway to which the host is

attached. Upon receipt of this advertisement, Layer 2 and 3 gateways in the topology install the MAC-to-IP bindings along with the associated next hops. When any of these gateways receives a packet with a destination MAC that matches an address in its MAC table, the gateway can check the next hop associated with the MAC address and forward the packet directly to the Layer 2 gateway to which the host is attached. This resulting packet flow eliminates the need for the packet to be forwarded first to a Layer 3 gateway, which then forwards the packet to the Layer 2 gateway.

We recommend you enable this feature in a centrally-routed bridging overlay fabric when any of the leaf devices in the fabric advertise only the MAC address of the connected hosts in their EVPN Type 2 route advertisements. If all of the leaf devices in the fabric can advertise both MAC and IP addresses of hosts in Type 2 advertisements, this setting is optional.

We recommend that you don't use this feature in an EVPN-VXLAN edge-routed bridging overlay (EVPN-VXLAN topology where the IP fabric is collapsed into a single layer of QFX Series switches that function as both Layer 2 and Layer 3 VXLAN gateways). Doing so might result in IP reachability issues in that type of fabric.

Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 15.1X53-D60.

remote-ip-host-routes

IN THIS SECTION

- [Syntax | 1648](#)
- [Hierarchy Level | 1648](#)
- [Description | 1648](#)
- [Options | 1648](#)
- [Required Privilege Level | 1648](#)
- [Release Information | 1648](#)

Syntax

```
remote-ip-host-routes {
    import [ import ... ];
    no-advertise-community;
}
```

Hierarchy Level

```
[edit routing-instances name protocols evpn],
```

Description

Allows you to configure virtual machine traffic optimization (VMTO) for EVPN.

Options

import	Policy to control the creation of remote IP host routes
no-advertise-community	<p>Enable this option to prevent type-2 MAC and IP community advertisements from being included in type-5 route imports, even though the information may be available. This was the default behavior when importing NLRI type-5 routes from remote PEs type prior to Junos 19.4R1. Starting in Junos 19.4R1, the default behavior is to include the <i>community</i> information when importing type-5 routes from type-2 route advertisements (unless this option is enabled).</p>

Required Privilege Level

routing

Release Information

Statement introduced in Junos OS Release 18.4R1.

The option, no-advertise-community, was introduced in Junos OS Release 19.4R1.

replicator

IN THIS SECTION

- [Syntax | 1649](#)
- [Hierarchy Level | 1649](#)
- [Description | 1649](#)
- [Options | 1650](#)
- [Required Privilege Level | 1651](#)
- [Release Information | 1651](#)

Syntax

```
replicator {  
    inet ip;  
    vxlan-encapsulation-source-ip (assisted-replicator-ip | ingress-replication-ip | retain-  
incoming-source-ip);  
}
```

Hierarchy Level

```
[edit logical-systems name routing-instances name protocols evpn assisted-replication],  
[edit routing-instances name protocols evpn assisted-replication],  
[edit protocols evpn assisted-replication]
```

Description

Configure a device in an EVPN network into the assisted replication (AR) replicator role.

AR helps to optimize broadcast, unknown unicast, and multicast (BUM) traffic in EVPN networks. To enable AR, you configure devices in the EVPN network to operate as AR replicator and AR leaf devices.

AR replicator devices take on BUM traffic replication and forwarding tasks for AR leaf devices. When an AR leaf device receives BUM traffic, instead of always performing ingress replication itself to the EVPN core, it forwards the traffic on an AR tunnel to an AR replicator device that can better handle the

replication load. The AR replicator device then replicates and forwards the traffic to other overlay tunnels. You configure a loopback interface IP address on each AR replicator device for the AR tunnel.

You can configure multiple AR replicator devices in an EVPN network, and AR leaf devices load-balance among the available AR replicator devices.

Options

inet *ip-address* (Required) IPv4 address for the AR tunnel to the AR replicator.

This must be a loopback interface configured on the AR replicator device that is part of the EVPN instance.

vxlan-encapsulation-source-ip (Mandatory for EVPN-VXLAN networks with QFX Series AR devices) Specify the type of VXLAN encapsulation source IP address to include in replicated and forwarded traffic.

- Values:
 - ingress-replication-ip—(Mandatory for EVPN-VXLAN networks with QFX Series AR devices) Use the ingress replication IP address.

NOTE: When configuring AR in EVPN-VXLAN networks with QFX Series devices as AR replicators and AR leaf devices, the `vxlan-encapsulation-source-ip` statement must be set to `ingress-replication-ip`. The `ingress-replication-ip` option supports AR on AR replicator devices that are not capable of retaining the source IP address of the originating AR leaf device (sent on the AR overlay tunnel) when replicating the traffic onto other overlay tunnels. In that case, AR must operate in *extended AR mode* (see ["Extended AR Mode for Multihomed Ethernet Segments" on page 758](#)). You must explicitly set this option to override the default factory configuration setting, `retain-incoming-source-ip`, which is not supported in this case.

AR is currently only supported in EVPN-VXLAN network with QFX Series AR devices operating in extended AR mode.

(Not currently used) The following `vxlan-encapsulation-source-ip` options are reserved for future AR compatibility among AR replicators with different capabilities:

- `assisted-replicator-ip`—Use the AR replicator IP address.
- `retain-incoming-source-ip`—(Default) Retain the VXLAN encapsulation source IP address of the incoming packet.

The remaining statements are explained separately. See [CLI Explorer](#).

Required Privilege Level

routing

Release Information

Statement introduced in Junos OS Release 18.4R2 and 19.4R1.

RELATED DOCUMENTATION

[Assisted Replication Multicast Optimization in EVPN Networks | 752](#)

[show evpn multicast-snooping assisted-replication next-hops | 1880](#)

[show evpn multicast-snooping assisted-replication replicators | 1883](#)

route-distinguisher

IN THIS SECTION

- [Syntax | 1652](#)
- [Hierarchy Level | 1652](#)
- [Description | 1652](#)
- [Options | 1653](#)
- [Required Privilege Level | 1654](#)
- [Release Information | 1654](#)

Syntax

```
route-distinguisher (as-number:id | ip-address:id);
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name],
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols
l2vpn mesh-group mesh-group-name],
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols vpls
mesh-group mesh-group-name],
[edit protocols evpn interconnect]
[edit routing-instances routing-instance-name],
[edit routing-instances routing-instance-name protocols evpn interconnect]
[edit routing-instances routing-instance-name protocols l2vpn mesh-group mesh-group-name],
[edit routing-instances routing-instance-name protocols vpls mesh-group mesh-group-name]
```

Description

Specify an identifier attached to a route, enabling you to distinguish to which VPN or virtual private LAN service (VPLS) the route belongs. Each routing instance must have a unique route distinguisher (RD) associated with it. The RD is used to place bounds around a VPN so that the same IP address prefixes can be used in different VPNs without having them overlap. If the instance type is vrf, the route-distinguisher statement is required.

For Layer 2 VPNs and VPLS, if you configure the `l2vpn-use-bgp-rules` statement, you must configure a unique RD for each PE router participating in the routing instance.

For Ethernet VPN (EVPN), you must configure a unique RD for each PE router participating in the routing instance to ensure that the prefixes generated by different PEs are unique.

For other types of VPNs, we recommend that you use a unique RD for each provider edge (PE) router participating in specific routing instance. Although you can use the same RD on all PE routers for the same VPN routing instance, if you use a unique RD, you can determine the customer edge (CE) router from which a route originated within the VPN.

For Layer 2 VPNs and VPLSs, if you configure mesh groups, the RD in each mesh group must be unique.



CAUTION: We strongly recommend that if you change an RD that has already been configured, or change the routing-instance type from virtual-router to vrf, make the change during a maintenance window, as follows:

1. Deactivate the routing instance.
2. Change the RD.
3. Activate the routing instance.

This is not required if you are configuring the RD for the first time.

Options

as-number: number—*as-number* is an assigned AS number, and *number* is any 2-byte or 4-byte value. The AS number can be from 1 through 4,294,967,295. If the AS number is a 2-byte value, the administrative number is a 4-byte value. If the AS number is 4-byte value, the administrative number is a 2-byte value. An RD consisting of a 4-byte AS number and a 2-byte administrative number is defined as a type 2 RD in RFC 4364 *BGP/MPLS IP VPNs*.

NOTE: In Junos OS Release 9.1 and later, the numeric range for AS numbers is extended to provide BGP support for 4-byte AS numbers, as defined in RFC 4893, *BGP Support for Four-octet AS Number Space*. All releases of Junos OS support 2-byte AS numbers. To configure an RD that includes a 4-byte AS number, append the letter “L” to the end of the AS number. For example, an RD with the 4-byte AS number 7,765,000 and an administrative number of 1,000 is represented as 77765000L:1000.

In Junos OS Release 9.2 and later, you can also configure a 4-byte AS number using the AS dot notation format of two integer values joined by a period: *<16-bit high-order value in decimal>.<16-bit low-order value in decimal>*. For example, the 4-byte AS number of 65,546 in the plain-number format is represented as 1.10 in AS dot notation format.

number: id—Number and identifier expressed in one of these formats: *16-bit number.32-bit identifier* or *32-bit number.16-bit identifier*.

ip-address: id—IP address (*ip-address* is a 4-byte value) within your assigned prefix range and a 2-byte value for the *id*. The IP address can be any globally unique unicast address.

- **Range:** 0 through 4,294,967,295 ($2^{32} - 1$). If the router you are configuring is a BGP peer of a router that does not support 4-byte AS numbers, you need to configure a local AS number. For more information, see *Using 4-Byte Autonomous System Numbers in BGP Networks Technology Overview*.

NOTE: For Ethernet VPN (EVPN), an RD that includes zero as the *id* value is reserved for the default EVPN routing instance by default. Because the same RD cannot be assigned for two routing instances, using a *ip-address.id* RD for another routing instance (default-switch), where the *id* value is zero, throws a commit error.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced before Junos OS Release 7.4.

Support at [edit routing-instances *routing-instance-name* protocols vpls mesh-group *mesh-group-name*] hierarchy level introduced in Junos OS Release 11.2.

Support at [edit routing-instances *routing-instance-name* protocols l2vpn mesh-group *mesh-group-name*] hierarchy level introduced in Junos OS Release 13.2.

Support at the following hierarchy levels introduced in Junos OS Release 20.3R1 on QFX Series switches: [edit protocols evpn interconnect] and [edit routing-instances *routing-instance-name* protocols evpn interconnect].

RELATED DOCUMENTATION

Example: Configuring BGP Route Target Filtering for VPNs

Example: Configuring FEC 129 BGP Autodiscovery for VPWS

[Configuring EVPN Routing Instances | 10](#)

Configuring Routing Instances on PE Routers in VPNs

[Configuring an MPLS-Based Layer 2 VPN \(CLI Procedure\)](#)

[Configuring an MPLS-Based Layer 3 VPN \(CLI Procedure\)](#)

path-selection

service-type

IN THIS SECTION

- [Syntax | 1655](#)
- [Hierarchy Level | 1655](#)
- [Description | 1655](#)
- [Options | 1656](#)
- [Required Privilege Level | 1656](#)
- [Release Information | 1656](#)

Syntax

```
service-type (vlan-aware | vlan-based | vlan-bundle);
```

Hierarchy Level

```
name],                                     [edit logical-systems name routing-instances mac-vrf routing-instance
name],                                     name],
instance name],                           [edit logical-systems name tenants name routing-instances mac-vrf routing-
instance name],                           instance name],
[edit routing-instances mac-vrf routing-instance name],
[edit tenants name routing-instances mac-vrf routing-instance name]
```

Description

Select the type of VLAN service that the MAC-VRF routing instance provides. The selected option controls how the routing instance maps VLANs to the forwarding instances on the device.

Options

- vlan-aware** Map one or more routing instances of type Virtual Switch to many VLAN IDs (VIDs) and multiple bridge tables. Each bridge table corresponds to a different VLAN.
- vlan-based** Map one routing instance of type EVPN to one VLAN. Only one bridge table corresponds to one VLAN. If the VLAN consists of multiple VLAN IDs (VIDs), then you must configure VLAN translation for packets that are traveling to the Ethernet segment. You do this when you use a different VID per Ethernet segment on a provider edge device.
- vlan-bundle** Map multiple broadcast domains to a single bridge domain. Multiple VLANs map to a single EVPN instance (EVI) and share the same bridge table in the MAC-VRF table. This configuration reduces the number of routes and labels stored in the table.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 20.4R1.

RELATED DOCUMENTATION

[MAC-VRF Routing Instance Type Overview](#) | 459

traceoptions (Protocols EVPN)

IN THIS SECTION

- [Syntax](#) | 1657
- [Hierarchy Level](#) | 1657
- [Description](#) | 1657
- [Options](#) | 1657

- Required Privilege Level | 1659
- Release Information | 1659

Syntax

```
traceoptions {
    file filename <files number> <size size> <world-readable | no-world-readable>;
    flag flag <flag-modifier>;
}
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols]
[edit routing-instances routing-instance-name protocols evpn]
```

Description

Trace traffic flowing through an EVPN routing instance.

Options

file filename—Name of the file to receive the output of the tracing operation. Enclose the name in quotation marks (" ").

files number—(Optional) Maximum number of trace files. When a trace file named *trace-file* reaches the maximum size as specified by the *size* option, it is renamed *trace-file.0*. When *trace-file* again reaches the maximum size, *trace-file.0* is renamed *trace-file.1* and *trace-file* is renamed *trace-file.0*. This renaming scheme continues until the maximum number of trace files is reached. Then the oldest trace file is overwritten.

If you specify a maximum number of files, you also must specify a maximum file size with the *size* option.

- **Range:** 2 through 1000 files
- **Default:** 2 files

flag *flag*—Tracing operation to perform. To specify more than one tracing operation, include multiple flag statements. You can specify the following tracing flags:

- all—All EVPN tracing options
- error—Error conditions
- general—General events
- mac-database—MAC route database in the EVPN routing instance
- nlri—EVPN advertisements received or sent by means of BGP
- normal—Normal events
- oam—OAM messages
- policy—Policy processing
- route—Routing information
- state—State transitions
- task—Routing protocol task processing
- timer—Routing protocol timer processing
- topology—EVPN topology changes caused by reconfiguration or advertisements received from other provider edge (PE) routers using BGP

flag-modifier—(Optional) Modifier for the tracing flag. You can specify the following modifiers:

- detail—Provide detailed trace information.
- disable—Disable this trace flag.
- receive—Trace received packets.
- send—Trace sent packets.

no-world-readable—Do not allow any user to read the log file.

size *size*—(Optional) Maximum size of each trace file, in kilobytes (KB), megabytes (MB), or gigabytes (GB). When a trace file named *trace-file* reaches this size, it is renamed *trace-file.0*. When *trace-file* again reaches the maximum size, *trace-file.0* is renamed *trace-file.1* and *trace-file* is renamed *trace-file.0*. This renaming scheme continues until the maximum number of trace files (as specified by the files option) is reached. Then the oldest trace file is overwritten.

If you specify a maximum file size, you also must specify a maximum number of trace files with the `files` option.

- **Syntax:** `yk` to specify kilobytes, `ym` to specify megabytes, or `yg` to specify gigabytes
- **Range:** 10 KB through the maximum file size supported on your system
- **Default:** 1 MB

`world-readable`—Allow any user to read the log file.

Required Privilege Level

`routing`—To view this statement in the configuration.

`routing-control`—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 13.2 .

Support for logical systems on MX Series routers added in Junos OS Release 17.4R1.

RELATED DOCUMENTATION

[Tracing EVPN Traffic and Operations](#) | 42

translation-vni

IN THIS SECTION

- [Syntax](#) | 1660
- [Hierarchy Level](#) | 1660
- [Description](#) | 1660
- [Options](#) | 1662
- [Required Privilege Level](#) | 1662
- [Release Information](#) | 1662

Syntax

```
translation-vni vni (1..16777214);
```

Hierarchy Level

```
[edit routing-instances routing-instance-name vlans vlan-name vxlan]
[edit routing-instances routing-instance-name vxlan]
[edit vlans vlan-name vxlan]
```

Description

Configure a VXLAN network identifier (VNI) to which a Juniper Networks switch will translate a VNI in a packet that it is forwarding. The Juniper Networks switch translates the VNI while it is forwarding traffic in the following seamless EVPN-VXLAN stitching use cases:

- From one EVPN-VXLAN point of delivery (POD) to another within a data center.
- From one EVPN-VXLAN data center to another. This use case is also known as data center interconnect (DCI).

In these use cases, a Juniper Networks device that supports the seamless EVPN-VXLAN stitching feature can serve as either a spine or super spine device that interconnects the PODs or data centers through an EVPN-VXLAN WAN network.

When configuring the interconnection, you can set up a single routing instance of type virtual-switch or evpn on each spine or super spine device. Or, you can use the default switching instance.

The type of instance that you use determines the hierarchy level at which you must include the translation-vni configuration statement. [Table 52 on page 1660](#) outlines the instance types and the hierarchy levels at which you must include translation-vni for each type.

Table 52: Interconnection Instance Types and Hierarchy Levels at Which to Include translation-vni Configuration Statement

Instance Types	Hierarchy Levels
Routing instance of type virtual-switch	[edit routing-instances routing-instance-name vlans vlan-name vxlan]

Table 52: Interconnection Instance Types and Hierarchy Levels at Which to Include translation-vni Configuration Statement (*Continued*)

Instance Types	Hierarchy Levels
Routing instance of type evpn	[edit routing-instances <i>routing-instance-name</i> vxlan]
Default switching instance	[edit vlans <i>vlan-name</i> vxlan]

The Juniper Networks switch translates the VNI only if you have included the translated VNI in the interconnected VNI list. For information about configuring this list, see ["interconnected-vni-list" on page 1597](#).

To understand how VNI translation works, Data Centers 1 and 2 are interconnected through an EVPN-VXLAN WAN network. In this topology, Super Spine 1 connects Data Center 1 and the WAN network, and Super Spine 2 connects Data Center 2 and the WAN network.

A routing instance named `evpn-vxlan` is configured on both Super Spines 1 and 2. In this routing instance, VLAN 100 is configured on each super spine device. However, the VNI mapped to VLAN 100 is different on each super spine device.

[Table 53 on page 1661](#) provides a summary of the VNI-related attributes configured on each super spine device.

Table 53: Summary of VNI-Related Configurations on Super Spines 1 and 2

VNI-Related Configurable Attributes	Super Spine 1 evpn-vxlan Routing Instance Configuration	Super Spine 2 evpn-vxlan Routing Instance Configuration
VLAN ID	100	100
VNI	100	300
Translated VNI	200	200
Interconnected VNI List	Includes VNI 200	Includes VNI 200

As shown in [Table 53 on page 1661](#), Super Spine 1 maps VNI 100 to VLAN 100, and Super Spine 2 maps VNI 300 to VLAN 100. Both super spine devices include VNI 200 in their interconnected VNI lists. As a result, when the super spine devices forward VLAN 100 traffic to the WAN network, the VNI is translated to 200.

Options

vni Numeric value that represents the VNI to which the VNI in a packet is translated.

- **Range:** 1through 16777214

Required Privilege Level

admin—To view this statement in the configuration.

admin-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 20.3R1.

RELATED DOCUMENTATION

| [interconnect](#) | [1595](#)

vlan-id (routing instance)

IN THIS SECTION

- [Syntax](#) | [1663](#)
- [Hierarchy Level](#) | [1663](#)
- [Description](#) | [1663](#)
- [Options](#) | [1663](#)
- [Required Privilege Level](#) | [1663](#)
- [Release Information](#) | [1663](#)

Syntax

```
vlan-id (vlan-id | all | none);
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name],  
[edit routing-instances routing-instance-name]  
[edit routing-instances routing-instance-name instance-type]
```

Description

Specify 802.1Q VLAN tag IDs to a routing instance.

Options

vlan-id—A valid VLAN identifier.

- **Range:** For 4-port Fast Ethernet PICs, 512 through 1023. For 1-port and 10-port Gigabit Ethernet PICs configured to handle VPLS traffic, 512 through 4094.

all—Include all VLAN identifiers specified on the logical interfaces included in the routing instance.

none—Include no VLAN identifiers for the routing instance.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 13.2.

RELATED DOCUMENTATION

[Configuring EVPN Routing Instances | 10](#)

[Configuring EVPN Routing Instances on EX9200 Switches | 13](#)

vpn-apply-export

IN THIS SECTION

- [Syntax | 1664](#)
- [Hierarchy Level | 1664](#)
- [Description | 1664](#)
- [Required Privilege Level | 1664](#)
- [Release Information | 1665](#)

Syntax

```
vpn-apply-export;
```

Hierarchy Level

```
[edit logical-systems logical-system-name protocols bgp],  
[edit logical-systems logical-system-name protocols bgp group group-name],  
[edit logical-systems logical-system-name protocols bgp group group-name neighbor neighbor],  
[edit protocols bgp],  
[edit protocols bgp group group-name],  
[edit protocols bgp group group-name neighbor neighbor]
```

Description

Apply both the VRF export and BGP group or neighbor export policies (VRF first, then BGP) before routes from the vrf or l2vpn routing tables are advertised to other PE routers.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced before Junos OS Release 7.4.

RELATED DOCUMENTATION

| *Configuring Policies for the VRF Table on PE Routers in VPNs*

vpws-service-id

IN THIS SECTION

- [Syntax | 1665](#)
- [Hierarchy Level | 1665](#)
- [Description | 1666](#)
- [Options | 1666](#)
- [Required Privilege Level | 1666](#)
- [Release Information | 1666](#)

Syntax

```
vpws-service-id {  
    local service-id;  
    remote servivce-id;  
}
```

Hierarchy Level

```
[edit routing-instance instance-type protocols evpn interface interface-name]
```

Description

Specify the local and remote Ethernet VPN (EVPN)-Virtual Private Wire Service (VPWS) service identifiers. These service identifiers are unique to an EVPN and are used to identify the endpoints of the EVPN-VPWS network. These endpoints are autodiscovered by BGP and are used to exchange the service labels (learned from the respective provider edge (PE) routers) that are used by autodiscovered routes per EVI route type. Depending on the mode of operation of the PE routers in the EVPN-VPWS network, these two endpoints of the can be colocated on the same PE router or on different PE routers.

Options

- local** Unique local VPWS service identifier of the EVPN-VPWS network. This identifies the logical interface of the PE router forwarding traffic to the PE router with a remote VPWS service identifier in the EVPN-VPWS network.
- **Range:** 1 through 16,777,215
- remote** Unique remote VPWS service identifier of the EVPN-VPWS network. This identifies the logical interface of the PE router receiving the traffic from the PE router with a local VPWS service identifier in the EVPN-VPWS network.
- **Range:** 1 through 16,777,215

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 17.1.

RELATED DOCUMENTATION

[Overview of VPWS with EVPN Signaling Mechanisms | 1046](#)

[EVPN Multihoming Overview | 96](#)

[Configuring VPWS with EVPN Signaling Mechanisms | 1068](#)

[Example: Configuring VPWS with EVPN Signaling Mechanisms | 1070](#)

[show evpn vpws-instance | 1903](#)

vrf-export

IN THIS SECTION

- [Syntax | 1667](#)
- [Hierarchy Level | 1667](#)
- [Description | 1667](#)
- [Default | 1668](#)
- [Options | 1668](#)
- [Required Privilege Level | 1668](#)
- [Release Information | 1668](#)

Syntax

```
vrf-export [ policy-names ];
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name],  
[edit logical-systems logical-system-name routing-instances routing-instance-name  
protocols  
vpls mesh-group mesh-group-name]  
[edit protocols evpn interconnect]  
[edit routing-instances routing-instance-name]  
[edit routing-instances routing-instance-name protocols evpn interconnect]  
[edit routing-instances routing-instance-name protocols vpls mesh-group mesh-group-name]  
[edit switch-options]
```

Description

Specify how routes are exported from the local device's routing table (*routing-instance-name*.inet.0) to the remote device. If the value `vrf` is specified for the `instance-type` statement included in the routing instance configuration, this statement is required.

You can configure multiple export policies on the router or switch.

Default

If the instance-type is vrf, vrf-export is a required statement. The default action is to reject.

Options

policy-names—Names for the export policies.

Required Privilege Level

routing— To view this statement in the configuration.

routing-control— To add this statement to the configuration.

Release Information

Statement introduced before Junos OS Release 7.4.

Support at the following hierarchy levels introduced in Junos OS Release 20.3R1 on QFX Series switches: [edit protocols evpn interconnect] and [edit routing-instances *routing-instance-name* protocols evpn interconnect].

RELATED DOCUMENTATION

[Implementing EVPN-VXLAN for Data Centers | 432](#)

instance-type

Configuring Policies for the VRF Table on PE Routers in VPNs

vrf-import

IN THIS SECTION

- [Syntax | 1669](#)
- [Hierarchy Level | 1669](#)
- [Description | 1669](#)
- [Options | 1669](#)

- Required Privilege Level | 1670
- Release Information | 1670

Syntax

```
vrf-import [ policy-names ];
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name],
[edit logical-systems logical-system-name routing-instances routing-instance-name
protocols                                vpls mesh-group mesh-group-name]
[edit protocols evpn interconnect]
[edit routing-instances routing-instance-name]
[edit routing-instances routing-instance-name protocols evpn interconnect]
[edit routing-instances routing-instance-name protocols vpls mesh-group mesh-group-name]
[edit switch-options]
```

Description

Specify how routes are imported into the routing table (*routing-instance-name*.inet.0) of the local device from the remote device.

You can configure multiple import policies on the device.

One of the following statements are required for importing routes:

- **vrf-target** - When you configure only the `vrf-target` statement without the `vrf-import` statement, by default all routes matching the specified target community are accepted.
- **vrf-import** - When you configure only the `vrf-import` statement, there is no default action. Only routes accepted in the configured `vrf-import` policy statement are imported.

Options

policy-names—Names for the import policies.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced before Junos OS Release 7.4.

Support at the following hierarchy levels introduced in Junos OS Release 20.3R1 on QFX Series switches: [edit protocols evpn interconnect] and [edit routing-instances *routing-instance-name* protocols evpn interconnect].

RELATED DOCUMENTATION

[Implementing EVPN-VXLAN for Data Centers | 432](#)

instance-type

Configuring Policies for the VRF Table on PE Routers in VPNs

vrf-target

IN THIS SECTION

- [Syntax | 1671](#)
- [Hierarchy Level | 1671](#)
- [Description | 1671](#)
- [Options | 1671](#)
- [Required Privilege Level | 1672](#)
- [Release Information | 1672](#)

Syntax

```
vrf-target {
    community;
    auto
    import community-name;
    export community-name;
}
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name],
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols
l2vpn mesh-group mesh-group-name],
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols vpls
mesh-group mesh-group-name],
[edit protocols evpn interconnect],
[edit routing-instances routing-instance-name protocols evpn interconnect],
[edit routing-instances routing-instance-name protocols evpn vni-options],
[edit routing-instances routing-instance-name protocols l2vpn mesh-group mesh-group-name],
[edit routing-instances routing-instance-name protocols vpls mesh-group mesh-group-name],
[edit switch-options]
```

Description

Specify a virtual routing and forwarding (VRF) target community. If you configure the *community* option only, default VRF import and export policies are generated that accept and tag routes with the specified target community. The purpose of the *vrf-target* statement is to simplify the configuration by allowing you to configure most statements at the [edit routing-instances] hierarchy level. In effect, this statement configures a single policy for import and a single policy for export to replace the per-VRF policies for every community.

You can still create more complex policies by explicitly configuring VRF import and export policies using the *import* and *export* options.

Options

community—Community name.

auto—Automatically derives the route target (RT). The auto-derived route targets have higher precedence over manually configured RT in vrf-target, vrf-export policies, and vrf-import policies.

NOTE: Auto-derived route targets are supported only in virtual switch and EVPN routing instances.

import community-name—Allowed communities accepted from neighbors.

export community-name—Allowed communities sent to neighbors.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced before Junos OS Release 7.4.

auto option added in Junos OS Release 19.1R1 for MX series.

Support at the following hierarchy levels introduced in Junos OS Release 20.3R1 on QFX Series switches: [edit protocols evpn interconnect] and [edit routing-instances *routing-instance-name* protocols evpn interconnect].

RELATED DOCUMENTATION

Configuring Policies for the VRF Table on PE Routers in VPNs

Example: Configuring FEC 129 BGP Autodiscovery for VPWS

VXLAN Configuration Statements

IN THIS CHAPTER

- decapsulate-inner-vlan | 1674
- encapsulate-inner-vlan | 1675
- encapsulation vxlan | 1677
- extended-vni-all | 1678
- extended-vni-list | 1680
- ingress-node-replication (EVPN) | 1682
- interface-num | 1683
- loopback-port | 1685
- next-hop (VXLAN Routing) | 1687
- policy-set | 1689
- riot-loopback | 1691
- static-remote-vtep-list | 1692
- tunnel-inspection | 1696
- virtual-gateway-address | 1698
- virtual-gateway-v4-mac | 1700
- virtual-gateway-v6-mac | 1702
- vni | 1704
- vni-options | 1706
- vnid (EVPN) | 1707
- vtep-source-interface | 1708
- vxlan | 1710
- vxlan-disable-copy-tos-decap | 1712
- vxlan-disable-copy-tos-encap | 1714
- vxlan-gbp-profile | 1716
- vxlan-routing | 1718

decapsulate-inner-vlan

IN THIS SECTION

- [Syntax | 1674](#)
- [Hierarchy Level | 1674](#)
- [Description | 1674](#)
- [Default | 1674](#)
- [Options | 1674](#)
- [Required Privilege Level | 1675](#)
- [Release Information | 1675](#)

Syntax

```
decapsulate-inner-vlan
```

Hierarchy Level

```
[edit routing-instances routing-instance-name vlan vlan-name vxlan],  
[edit vlan vlan-name vxlan]
```

Description

Configure the switch to de-encapsulate a preserved original VLAN tag (in the inner Ethernet packet) from a VXLAN encapsulated packet.

Default

A preserved VLAN tag is dropped when the packet is de-encapsulated.

Options

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 14.1X53-D10.

RELATED DOCUMENTATION

[Understanding VXLANs | 414](#)

Manually Configuring VXLANs on QFX Series and EX4600 Switches

Examples: Manually Configuring VXLANs on QFX Series and EX4600 Switches

[encapsulate-inner-vlan | 1675](#)

encapsulate-inner-vlan

IN THIS SECTION

- [Syntax | 1675](#)
- [Hierarchy Level | 1676](#)
- [Description | 1676](#)
- [Default | 1676](#)
- [Required Privilege Level | 1676](#)
- [Release Information | 1676](#)

Syntax

```
encapsulate-inner-vlan
```

Hierarchy Level

```
[edit routing-instances routing-instance-name vlan vlan-name vxlan],
[edit vlan vlan-name vxlan]
```

Description

Configure the switch to preserve the original VLAN tag (in the inner Ethernet packet) when performing Virtual Extensible LAN (VXLAN) encapsulation.

NOTE: ARP suppression is automatically disabled when you encapsulate VLAN traffic for EVPN-VXLAN by configuring `encapsulate-inner-vlan` and its equivalent converse statement `decapsulate-accept-inner-vlan`. As a result, you may observe ARP packets being sent by the device.

Default

The original tag is dropped when the packet is encapsulated.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 14.1R2.

RELATED DOCUMENTATION

[Understanding VXLANs | 414](#)

Manually Configuring VXLANs on QFX Series and EX4600 Switches

Examples: Manually Configuring VXLANs on QFX Series and EX4600 Switches

decapsulate-accept-inner-vlan

encapsulation vxlan

IN THIS SECTION

- [Syntax | 1677](#)
- [Hierarchy Level | 1677](#)
- [Description | 1677](#)
- [Required Privilege Level | 1678](#)
- [Release Information | 1678](#)

Syntax

```
encapsulation vxlan;
```

Hierarchy Level

```
{edit protocols evpn},  
[edit routing-instances routing-instance-name protocols evpn]
```

Description

Configure a VXLAN encapsulation type. This statement is required for a VXLAN EVPN instance.

NOTE: If you configure the `encapsulation vxlan` statement, then you must also configure the [extended-vni-list](#) statement.

NOTE: The `encapsulation vxlan` statement is an exclusive command. You cannot configure the `encapsulation vxlan` statement with the [extended-vlan-list](#) statement, or other commands associated with MPLS EVPN instances.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 16.1.

RELATED DOCUMENTATION

[Understanding EVPN with VXLAN Data Plane Encapsulation | 398](#)

[EVPN Over VXLAN Encapsulation Configuration Overview for QFX Series and EX4600 Switches | 430](#)

[Example: Configure an EVPN-VXLAN Centrally-Routed Bridging \(CRB\) Using MX Routers as Spines | 580](#)

extended-vni-all

IN THIS SECTION

- [Syntax | 1678](#)
- [Hierarchy Level | 1679](#)
- [Description | 1679](#)
- [Required Privilege Level | 1679](#)
- [Release Information | 1679](#)

Syntax

```
extended-vni-all;
```

Hierarchy Level

For QFX5100 and EX4600 switches:

```
{edit protocols evpn}
```

For MX Series routers:

```
[edit routing-instances routing-instance-name protocols evpn]
```

Description

Include all VXLAN Network Identifiers (VNIs) as part of the Virtual Switch (VS) instance. By specifying `all`, you bypass the `commit check` process and all configured bridge domains (BDs) in the Ethernet Virtual Private Network (EVPN) are considered extended.

NOTE: The [extended-vni-list](#) statement is an exclusive command. You cannot configure the `extended-vni-list` statement with either the [extended-vlan-list](#) or `extended-vni-all` statements.

Required Privilege Level

`routing`—To view this statement in the configuration.

`routing-control`—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 14.1X53-D30.

RELATED DOCUMENTATION

[Example: Configure an EVPN-VXLAN Centrally-Routed Bridging \(CRB\) Using MX Routers as Spines](#) | 580

[Understanding EVPN with VXLAN Data Plane Encapsulation](#) | 398

[EVPN Over VXLAN Encapsulation Configuration Overview for QFX Series and EX4600 Switches](#) | 430

[extended-vni-list](#) | 1680

extended-vni-list

IN THIS SECTION

- [Syntax | 1680](#)
- [Hierarchy Level | 1680](#)
- [Description | 1680](#)
- [Options | 1681](#)
- [Required Privilege Level | 1681](#)
- [Release Information | 1681](#)

Syntax

```
extended-vni-list [list of VNIs / all];
```

Hierarchy Level

For MX Series routers, EX9200, and QFX Series switches:

```
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols],  
[edit routing-instances routing-instance-name protocols evpn]
```

Description

Establishes which VXLAN Network Identifiers (VNI) will be part of the Virtual Switch (VS) instance. When you issue the `commit check` command to verify the candidate configuration syntax without committing it, it also checks if the specified VNI(s) is associated with a bridge domain (BD) (`bridge-domain name vxlan vni`).

There are different broadcast, unknown unicast, and multicast (BUM) replication options available in Ethernet Virtual Private Network (EVPN). By using the `extended-vni-list` statement, you forgo a multicast underlay in favor of EVPN and VXLAN ingress-replication.

NOTE: The `extended-vni-list` statement is an exclusive command. You cannot configure the `extended-vni-list` statement with either the `extended-vlan-list` or `extended-vni-all` statements.

NOTE: If you configure the `extended-vni-list` statement, then you must also configure the `encapsulation vxlan` statement.

NOTE: The `extended-vni-list` statement is optional for routing-instances of type `mac-vrf`. In the `mac-vrf` type routing instance, all VNI are extended by default. If you configure an `extended-vni-list` within a specific `mac-vrf` type routing instance, you negate the default behavior for that routing instance.

Options

- | | |
|---------------------|--|
| list of VNIs | Specify a single VNI or list of VNIs as part of the VS instance, for example <code>extended-vni-list [10-50 60 70]</code> . |
| all | Include all VNIs as part of the VS instance. By specifying <code>all</code> , you bypass the <code>commit check</code> process and all configured BDs in the EVPN are considered extended. |

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 14.1X53-D15.

Support for logical systems on MX Series routers added in Junos OS Release 17.4R1.

RELATED DOCUMENTATION

[Example: Configure an EVPN-VXLAN Centrally-Routed Bridging \(CRB\) Using MX Routers as Spines | 580](#)

[Understanding EVPN with VXLAN Data Plane Encapsulation | 398](#)

[EVPN Over VXLAN Encapsulation Configuration Overview for QFX Series and EX4600 Switches | 430](#)

[show configuration protocols evpn](#)

[Implementing EVPN-VXLAN for Data Centers | 432](#)

ingress-node-replication (EVPN)

IN THIS SECTION

- [Syntax | 1682](#)
- [Hierarchy Level | 1682](#)
- [Description | 1682](#)
- [Required Privilege Level | 1683](#)
- [Release Information | 1683](#)

Syntax

```
ingress-node-replication;
```

Hierarchy Level

```
[edit routing-instances routing-instance-name vlans vlan-name vxlan],
[edit vlans vlan-name vxlan]
```

Description

EVPN A/A with VXLAN encapsulation is based on the local bias for traffic coming from the access layer (redundant Layer 2 gateway function through a pair of top-of-rack switches). Because the traffic has no MPLS label, the split-horizon filtering rule for multi-home Ethernet segment is modified to be based on the IP address of the EVPN provider edge (PE) instead of the MPLS ES-label. This is called local bias for EVPN-VXLAN. Each EVPN PE tracks the IP address of its peer multihomed EVPN PE that share the

same Ethernet segment. This is the source VTEP IP address (outer SIP) for each VXLAN packet received from other EVPN PE. The local bias filtering rule is enforced on both ingress and egress PEs for the multi-destination traffic. For egress traffic, there is no forwarding of any multi-destination packets to the same multihomed Ethernet segment that an egress PE shares with its ingress PE regardless of the egress PE's DF election status for that Ethernet segment. Ingress traffic is responsible for forwarding multi-destination packets coming from any directly attached access interfaces to the rest of the multi-home Ethernet segments associated with it regardless of the ingress PE's designated forwarder election status on the connected physical device segment.

NOTE: With this statement configured, the device adds all active VTEP interfaces to every bridge domain (VLAN). However, the default routing behavior without this statement results in an optimal flood list with VTEPs and VLANs associated only based on advertised EVPN Type 3 routes. As a result, in general you don't need to configure this statement on EVPN-VXLAN devices.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 14.1X53-D30 .

RELATED DOCUMENTATION

[Understanding EVPN with VXLAN Data Plane Encapsulation](#) | 398

interface-num

IN THIS SECTION

- [Syntax](#) | 1684
- [Hierarchy Level](#) | 1684

- [Description | 1684](#)
- [Options | 1685](#)
- [Required Privilege Level | 1685](#)
- [Release Information | 1685](#)

Syntax

```
interface-num integer;
```

Hierarchy Level

```
[edit forwarding-options vxlan-routing ]
```

Description

Configure the maximum number of physical interfaces reserved for use in an Ethernet VPN-Virtual Extensible LAN (EVPN-VXLAN) overlay network.

Juniper Networks switches support a maximum number of physical interfaces for use in either an underlay network or an EVPN-VXLAN overlay network. By default, a switch allocates 4,000 physical interfaces for use in an EVPN-VXLAN network, which leaves the remaining physical interfaces for use in an underlay network. For example, the default setting of 4,000 physical interfaces for a QFX5110 switch, which supports a maximum of 12,288 physical interfaces, leaves 8,288 physical interfaces for use in an underlay network. The `interface-num` configuration statement enables you to re-allocate the maximum number of physical interfaces reserved for use in an EVPN-VXLAN network.

If you do not plan to use a switch in an EVPN-VXLAN network, you can set the `interface-num` configuration statement to 0, which allocates all physical interfaces for use in underlay networks.

Conversely, if you plan to use a switch in an EVPN-VXLAN network, you can re-allocate a greater number of the physical interfaces to the EVPN-VXLAN network.

NOTE: Changing the default number of physical interfaces reserved for use in an EVPN-VXLAN network causes the Packet Forwarding Engine to restart. Restarting the Packet Forwarding

Engine interrupts all forwarding operations. Therefore, we strongly recommend using this configuration statement before the EVPN-VXLAN network becomes operational.

Options

integer Maximum number of physical interfaces reserved for use in an EVPN-VXLAN overlay network on a switch.

- **Range:** QFX5110 switches: 0 through 12,288. QFX5120 switches: 0 through 14336. The specified value must be a multiple of 2048—for example, 2048, 4096, 6144, and so on.
- **Default:** 4000

Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 17.3R1.

loopback-port

IN THIS SECTION

- [Syntax | 1686](#)
- [Hierarchy Level | 1686](#)
- [Description | 1686](#)
- [Required Privilege Level | 1687](#)
- [Release Information | 1687](#)

Syntax

```
loopback-port loopback-port;
```

Hierarchy Level

```
[edit forwarding-options vxlan-routing ]
```

Description

Define the global loopback aggregated Ethernet interface for VXLAN routing on QFX5210 leaf devices in EVPN-VXLAN edge-routed bridging (ERB) overlay fabrics.

Specify *loopback-port* using its physical interface name, without a specific unit number (such as ae0 rather than ae0.100, for example).

NOTE: The Packet Forwarding Engine restarts when you configure or update this setting.

QFX5210 switches don't have native VXLAN routing hardware support for Layer 3 VXLAN gateway functions. Instead, the switch uses an intermediary loopback port in a two-pass process for routing in and out of VXLAN tunnels (RIOT). You configure a loopback link aggregation group (LAG) and use this statement to assign it as the RIOT loopback LAG port on the device.

Keep the following in mind when you configure the RIOT loopback LAG:

- When you configure the loopback LAG members, you can include any network ports on the switch that you're not using for another purpose. Based on your VXLAN routing traffic load, you can adjust:
 - The number of links in the LAG.
 - The minimum number of links for the LAG to be considered "up".
- You must configure the loopback LAG interface as a member of all VLANs with IRB interfaces used for VXLAN routing.
- Configure the loopback LAG interface:
 - As a trunk interface

- With encapsulation `flexible-ethernet-services` and `flexible-vlan-tagging`
- The device automatically turns off MAC learning on the interface when you set it as the loopback LAG port. In this way, upon receiving packets on the loopback port, the device avoids learning the port's own MAC address.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 21.3R1 for QFX5210 switches.

RELATED DOCUMENTATION

[Using a RIOT Loopback Port to Route Traffic in an EVPN-VXLAN Network | 514](#)

next-hop (VXLAN Routing)

IN THIS SECTION

- [Syntax | 1688](#)
- [Hierarchy Level | 1688](#)
- [Description | 1688](#)
- [Options | 1688](#)
- [Required Privilege Level | 1689](#)
- [Release Information | 1689](#)

Syntax

```
next-hop integer;
```

Hierarchy Level

```
[edit forwarding-options vxlan-routing ]
```

Description

Configure the maximum number of next hops reserved for use in an Ethernet VPN-Virtual Extensible LAN (EVPN-VXLAN) overlay network.

Juniper Networks switches can store a maximum number of next hops for use in either an underlay network or an EVPN-VXLAN overlay network. By default, a switch allocates 8,000 next hops for use in an EVPN-VXLAN network, which leaves the remaining next hops for use in an underlay network. For example, the default setting of 8,000 next hops for a QFX5110 switch, which can store a maximum of 45,056 next hops, leaves 37,056 next hops for use in an underlay network. The `next-hop` configuration statement enables you to re-allocate the maximum number of next hops reserved for use in an EVPN-VXLAN network.

If you do not plan to use a switch in an EVPN-VXLAN network, you can set the `next-hop` configuration statement to 0, which allocates all next hops for use in an underlay network.

Conversely, if you plan to use a switch in an EVPN-VXLAN network, you can re-allocate a greater number of the next hops to the EVPN-VXLAN network.

NOTE: Changing the default number of next hops reserved for use in an EVPN-VXLAN network causes the Packet Forwarding Engine to restart. Restarting the Packet Forwarding Engine interrupts all forwarding operations. Therefore, we strongly recommend using this configuration statement before the EVPN-VXLAN network becomes operational.

Options

integer Maximum number of next hops reserved for use in an EVPN-VXLAN overlay network on a switch.

- **Range:** QFX5110 switches: 0 through 45056. QFX5120 switches: 0 through 61440. The specified value must be a multiple of 4096—for example, 4096, 8192, 12.288, and so on.
- **Default:** 8000

Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 17.3R1.

policy-set

IN THIS SECTION

- [Syntax | 1689](#)
- [Hierarchy Level | 1690](#)
- [Description | 1690](#)
- [Options | 1690](#)
- [Required Privilege Level | 1690](#)
- [Release Information | 1690](#)

Syntax

```
policy-set policy-set-name {
    policy policy-name {
        match {
            source-address any;
            destination-address any;
            application any;
            from-zone trust
```

```

        to-zone untrust;
    }
    then permit;
}
}
}
}
}

```

Hierarchy Level

[edit security policies]

Description

Specify policy that applies for the inner session created by VXLAN inner header

Options

source-address	Source address of the traffic as matching criteria. You can specify one or more IP addresses, address sets, or wildcard addresses.
destination-address	Destination address of the traffic as matching criteria. You can specify one or more IP addresses, address sets, or wildcard addresses.
application	Name of the predefined or custom application or application set used as match criteria.
from-zone	Name of the source zone used as match criteria.
to-zone	Name of the destination zone used as match criteria.

Required Privilege Level

security—To view this statement in the configuration.

security-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 20.1R1.

riot-loopback

IN THIS SECTION

- [Syntax | 1691](#)
- [Hierarchy Level | 1691](#)
- [Description | 1691](#)
- [Required Privilege Level | 1692](#)
- [Release Information | 1692](#)

Syntax

```
riot-loopback;
```

Hierarchy Level

```
[edit vlans name vxlan]
```

Description

Assign a VLAN that enables routing in and out of VXLAN tunnels (RIOT) for EVPN Type 5 routing.

Some Juniper switching platforms don't have native VXLAN routing hardware support for Layer 3 VXLAN gateway functions. Instead, supporting switches can use an intermediary loopback port in a two-pass process for routing in and out of VXLAN tunnels (RIOT). You configure this intermediary port as a loopback link aggregation group (LAG) that is a member of all VLANs with IRB interfaces used for VXLAN routing.

To support EVPN Type 5 routes with the RIOT loopback LAG port, for each Type 5 virtual routing and forwarding (VRF) instance, you:

- Configure an extra VLAN with an IRB interface.
- Map that VLAN to the same VXLAN network identifier (VNI) that you configure for the corresponding EVPN Type 5 VRF instance.

- Configure this statement with the extra VLAN so the Type 5 VRF instance uses the RIOT loopback LAG port for Type 5 VXLAN routing.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 21.3R1.

RELATED DOCUMENTATION

[Using a RIOT Loopback Port to Route Traffic in an EVPN-VXLAN Network | 514](#)

static-remote-vtep-list

IN THIS SECTION

- [Syntax | 1692](#)
- [Hierarchy Level | 1693](#)
- [Description | 1693](#)
- [Options | 1696](#)
- [Required Privilege Level | 1696](#)
- [Release Information | 1696](#)

Syntax

```
static-remote-vtep-list [ remote-vtep-loopback-address ... ];
```

Hierarchy Level

```
[edit bridge-domains name vxlan],
[edit routing-instances name bridge-domains name vxlan],
[edit vlans name vxlan]
```

Description

Statically configure a list of one or more remote virtual tunnel endpoints (VTEPs), and map the list to a particular bridge domain or VLAN.

NOTE: If you have previously configured EVPN-VXLAN on a device, then later configure static VXLAN, you must reboot the device to let static VXLAN take effect.

When configuring a static remote VTEP list, keep the following in mind:

- When specifying remote VTEPs at the bridge domain level in a routing instance, you must also specify the same VTEPs at the global level in the same routing instance. When specifying remote VTEPs at the VLAN level in the default switching instance, you must also specify the same VTEPs at the global level in the default switching instance. To illustrate this point, note the following in sample configurations 1 and 2:
 - Routing instance `rt1` and the default switching instance include the VTEPs with the loopback addresses of `10.1.1.1` and `10.1.1.2`.
 - Bridge domain `bd1` and `VLAN-1` include the VTEPs with the loopback addresses of `10.1.1.1` and `10.1.1.2`.
 - Bridge domain `bd2` and `VLAN-2` include the VTEP with the loopback address of `10.1.1.1`.
 - Bridge domain `bd3` and `VLAN-3` don't include a static remote VTEP list. As a result, `bd3` and `VLAN-3` inherit all VTEPs specified in routing instance `rt1` and the default switching instance (the VTEPs with the loopback addresses of `10.1.1.1` and `10.1.1.2`), respectively.
- To replicate and flood broadcast, unknown unicast, and multicast (BUM) traffic, you must specify the `ingress-node-replication` configuration statement at the `[edit vlans name vxlan]`, `[edit bridge-domains name vxlan]`, or `[edit routing-instances name bridge-domains name vxlan]` hierarchy level. In sample configuration 1, `ingress-node-replication` is configured for all bridge domains in routing instance `rt1`. In sample configuration 2, `ingress-node-replication` is configured for all VLANs in the default switching instance.

Note that this ingress node replication configuration restricts the BUM traffic flood domain to only those VTEPs mapped to a particular bridge domain or VLAN. For example, for sample configurations

1 and 2, the BUM traffic flood domain is restricted as described for each bridge domain and VLAN, respectively:

- bd1 and bd3; VLAN-1 and VLAN-3—BUM traffic is flooded to the VTEPs with the loopback addresses of 10.1.1.1 and 10.1.1.2.
- bd2; VLAN-2—BUM traffic is flooded to the VTEP with the loopback address of 10.1.1.1.

NOTE: To ensure that BUM traffic is flooded to only those VTEPs mapped to a particular VLAN, we strongly recommend that the static VTEP lists on all QFX5XXX switches are symmetric. We provide this guidance to avoid a situation where, for example, switch 1's list includes switches 2 and 3, switch 2's list includes switches 1 and 3, and switch 3's list includes only switch 2. Because of this asymmetric configuration, when switch 1 sends BUM traffic to switch 3, switch 3 will forward the traffic even though switch 1 is not on its list.

Sample Configuration 1 From MX Router (Routing Instance of Type virtual-switch)

```
routing-instances {
  rt1 {
    vtep-source-interface lo0.0;
    remote-vtep-list [ 10.1.1.1 10.1.1.2 ];
    instance-type virtual-switch;
    bridge-domains {
      bd1 {
        vlan-id 101;
        vxlan {
          vni 101;
          ingress-node-replication;
          static-remote-vtep-list [ 10.1.1.1 10.1.1.2 ];
        }
      }
      bd2 {
        vlan-id 102;
        vxlan {
          vni 102;
          ingress-node-replication;
          static-remote-vtep-list [ 10.1.1.1 ];
        }
      }
      bd3 {
        vlan-id 103;
```

```

        vxlan {
            vni 103;
            ingress-node-replication;
        }
    }
}
}
}
}

```

Sample Configuration 2 From QFX5XXX Switch (Default Switching Instance)

```

switch-options {
    vtep-source-interface lo0.0;
    remote-vtep-list [ 10.1.1.1 10.1.1.2 ];
}
vlangs {
    VLAN-1 {
        vlan-id 10001;
        interface xe-0/0/0:0.1001;
        vxlan {
            vni 10001;
            ingress-node-replication;
            static-remote-vtep-list [ 10.1.1.1 10.1.1.2 ];
        }
    }
    VLAN-2 {
        vlan-id 10002;
        interface xe-0/0/0:0.1002;
        vxlan {
            vni 10002;
            ingress-node-replication;
            static-remote-vtep-list [ 10.1.1.1 ];
        }
    }
    VLAN-3 {
        vlan-id 10003;
        interface xe-0/0/0:0.1003;
        vxlan {
            vni 10003;
            ingress-node-replication;
        }
    }
}

```

```
}
}
```

Options

static-remote-vtep-list Specify the loopback address of one or more remote VTEPs in square brackets ([]). You do not need to include a delimiter such as a comma between names.

Required Privilege Level

routing

Release Information

Statement introduced in Junos OS Release 20.4R1.

RELATED DOCUMENTATION

| [Static VXLAN](#) | [1536](#)

tunnel-inspection

IN THIS SECTION

- [Syntax](#) | [1697](#)
- [Hierarchy Level](#) | [1697](#)
- [Description](#) | [1697](#)
- [Options](#) | [1698](#)
- [Required Privilege Level](#) | [1698](#)
- [Release Information](#) | [1698](#)

Syntax

```
tunnel-inspection {
    inspection-profile profile-name {
        vxlan vxlan-name {
            policy-set pset-name;
            vni vni-name;
        }
    }
    traceoptions {
        file <filename> <files files> <match match> <size size> <(world-readable | no-world-readable)>;
        flag name;
        no-remote-trace;
    }
    vni vni-name {
        vni-id vni-id;
        vni-range <vni-range-low to vni-range-high>;
    }
}
```

Hierarchy Level

```
[edit security]
```

Description

Configure security inspection for EVPN- VXLAN tunnel traffic. Configure an outer policy for the outer header and an inner policy for the inner header.

When packet matches security policy, the security device decapsulates the packet to get the inner header. The tunnel inspection profile is applied for the permitted traffic. With inner packet content and the applied tunnel inspection profile, the device performs a policy lookup and performs the stateful inspection for the inner session traffic.

Options

inspection-profile <i>profile-name</i>	Configure a tunnel inspection profile to connect the outer policy and inner policy.
vxlan <i>vxlan-name</i>	VXLAN tunnel identifier.
policy-set <i>pset-name</i>	Policy that applies for the inner session created by VXLAN inner header.
trace-option	Configure traceoption for tunnel inspection.
vni <i>vni-name</i>	VXLAN network identifier (VNI).
vni-id <i>vni-id</i>	VXLAN network identifier (VNI) used to uniquely identify the VXLAN.
vni-range <i>vni-range</i>	VLAN ID range.

Required Privilege Level

security—To view this statement in the configuration.

security-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 20.1R1.

virtual-gateway-address

IN THIS SECTION

- [Syntax | 1699](#)
- [Hierarchy Level | 1699](#)
- [Description | 1699](#)
- [Options | 1699](#)
- [Required Privilege Level | 1699](#)
- [Release Information | 1699](#)

Syntax

```
virtual-gateway-address address;
```

Hierarchy Level

```
[edit interfaces interface-name unit logical-unit-number family family address address]
```

Description

Set the default IPv4 or IPv6 address for the gateway for end hosts. Because you must configure the virtual gateway address using the same IP address as on all of the provider edge (PE) devices (which internally generates the same Virtual Router Redundancy Protocol (VRRP) MAC), the need to proxy for remote gateway IP addresses is eliminated. Every logical integrated routing and bridging (IRB) interface can have a corresponding virtual gateway address. The maximum number of PEs that can have the same virtual gateway address is 64.

To support pinging on the virtual gateway IP address, you must include both the `virtual-gateway-accept-data` statement and the `preferred` statement at the `[edit interfaces irb unit]` hierarchy of the preferred virtual gateway.

Options

address—virtual gateway address. You cannot specify the addresses 0.0.0.0 (default route address) or 255.255.255.255 (broadcast IP address).

Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 16.1.

RELATED DOCUMENTATION

[EVPN with IRB Solution Overview | 946](#)

[Example: Configuring EVPN with IRB Solution | 974](#)

[Configuring the Interface Address](#)

virtual-gateway-v4-mac

IN THIS SECTION

- [Syntax | 1700](#)
- [Hierarchy Level | 1700](#)
- [Description | 1700](#)
- [Options | 1701](#)
- [Required Privilege Level | 1701](#)
- [Release Information | 1702](#)

Syntax

```
virtual-gateway-v4-mac ipv4-mac-address
```

Hierarchy Level

```
[edit dynamic-profiles name interfaces name unit logical-unit-number],  
[edit dynamic-profiles name logical-systems name interfaces name unit logical-unit-number],  
[edit interfaces name unit logical-unit-number]
```

Description

Explicitly configure an IPv4 media access control (MAC) address for a default virtual gateway.

A default virtual gateway is created when you specify a virtual gateway address (VGA) while configuring an integrated routing and bridging (IRB) interface on a Juniper Networks device that functions as a Layer

3 Virtual Extensible LAN (VXLAN) gateway in an Ethernet VPN-Virtual Extensible LAN (EVPN-VXLAN) centrally-routed bridging overlay (EVPN-VXLAN topology with a two-layer IP fabric). Through the IRB interface with which it is configured, the default virtual gateway enables communication between non-virtualized hosts, virtual machines (VMs), and servers in different VXLANs or IP subnetworks.

When you configure a VGA for an IRB interface, the Layer 3 VXLAN gateway automatically generates IPv4 MAC address 00:00:5e:00:01:01 for that particular virtual gateway. (This topic refers to the virtual gateway MAC address as a virtual MAC.) The automatically generated virtual MAC is not included as the source MAC address in packets generated by the Layer 3 VXLAN gateway. Instead, data packets and the source MAC address field in the outer Ethernet header of Address Resolution Protocol (ARP) replies and neighbor advertisement packets include the MAC address for the IRB interface. (This topic refers to the MAC address for the IRB interface as the IRB MAC.)

When an ARP reply includes the IRB MAC as the source MAC address instead of the virtual MAC, an issue might arise in a centrally-routed bridging overlay. This issue might result in the flooding of unknown-unicast packets throughout the domain.

If you explicitly configure a MAC address for a default virtual gateway, the automatically generated virtual MAC is overridden by the configured virtual MAC. That is, when the Layer 3 VXLAN gateway sends data packets, ARP replies, and neighbor advertisement packets, the configured virtual MAC is in the outer Ethernet header of these packets. As a result, the possibility that the domain is flooded with unknown-unicast packets is eliminated.

NOTE: The MAC address range 02:00:00:00:00:00:xy is used for internal communication. Do not use addresses in this range for a manual virtual MAC assignment.

For more information about the flooding issue and its resolution, see ["Understanding the MAC Addresses For a Default Virtual Gateway in an EVPN-VXLAN Overlay Network" on page 524](#).

Options

ipv4-mac-address IPv4 MAC address for the default virtual gateway.

Required Privilege Level

admin—To view this statement in the configuration.

admin-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 14.2R5.

RELATED DOCUMENTATION

| [virtual-gateway-v6-mac](#) | [1702](#)

virtual-gateway-v6-mac

IN THIS SECTION

- [Syntax](#) | [1702](#)
- [Hierarchy Level](#) | [1702](#)
- [Description](#) | [1703](#)
- [Options](#) | [1703](#)
- [Required Privilege Level](#) | [1704](#)
- [Release Information](#) | [1704](#)

Syntax

```
virtual-gateway-v6-mac ipv6-mac-address
```

Hierarchy Level

```
[edit dynamic-profiles name interfaces name unit logical-unit-number],  
[edit dynamic-profiles name logical-systems name interfaces name unit logical-unit-number],  
[edit interfaces name unit logical-unit-number]
```

Description

Explicitly configure an IPv6 media access control (MAC) address for a default virtual gateway.

A default virtual gateway is created when you specify a virtual gateway address (VGA) while configuring an integrated routing and bridging (IRB) interface on a Juniper Networks device that functions as a Layer 3 Virtual Extensible LAN (VXLAN) gateway in an Ethernet VPN-Virtual Extensible LAN (EVPN-VXLAN) centrally-routed bridging overlay (EVPN-VXLAN topology with a two-layer IP fabric). Through the IRB interface with which it is configured, the default virtual gateway enables communication between non-virtualized hosts, virtual machines (VMs), and servers in different VXLANs or IP subnetworks.

When you configure a VGA for an IRB interface, the Layer 3 VXLAN gateway automatically generates IPv6 MAC address 00:00:5E:00:02:01 for that particular virtual gateway. (This topic refers to the virtual gateway MAC address as a virtual MAC.) The automatically generated virtual MAC is not included as the source MAC address in packets generated by the Layer 3 VXLAN gateway. Instead, data packets and the source MAC address field in the outer Ethernet header of Address Resolution Protocol (ARP) replies and neighbor advertisement packets include the MAC address for the IRB interface. (This topic refers to the MAC address for the IRB interface as the IRB MAC.)

When an ARP reply includes the IRB MAC as the source MAC address instead of the virtual MAC, an issue might arise in a centrally-routed bridging overlay. This issue might result in the flooding of unknown-unicast packets throughout the domain.

If you explicitly configure a MAC address for a default virtual gateway, the automatically generated virtual MAC is overridden by the configured virtual MAC. That is, when the Layer 3 VXLAN gateway sends data packets, ARP replies, and neighbor advertisement packets, the configured virtual MAC is in the outer Ethernet header of these packets. As a result, the possibility that the domain is flooded with unknown-unicast packets is eliminated.

NOTE: The MAC address range 02:00:00:00:00:00:xy is used for internal communication. Do not use addresses in this range for a manual virtual MAC assignment.

For more information about the flooding issue and its resolution, see ["Understanding the MAC Addresses For a Default Virtual Gateway in an EVPN-VXLAN Overlay Network"](#) on page 524.

Options

ipv6-mac-address

IPv6 MAC address for the default virtual gateway.

Required Privilege Level

admin—To view this statement in the configuration.

admin-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 14.2R5.

RELATED DOCUMENTATION

| [virtual-gateway-v4-mac](#) | [1700](#)

vni

IN THIS SECTION

- [Syntax](#) | [1704](#)
- [Hierarchy Level](#) | [1705](#)
- [Description](#) | [1705](#)
- [Options](#) | [1705](#)
- [Required Privilege Level](#) | [1705](#)
- [Release Information](#) | [1705](#)

Syntax

```
vni [0-16777214]
```

Hierarchy Level

```
[edit routing-instances routing-instance-name vlans vlan-name vxlan]
[edit vlans vlan-name vxlan]
```

Description

Assign a numeric value to identify a Virtual Extensible LAN (VXLAN). All members of a VXLAN must use the same VNI.

Options

vni Value to specify in the `vni` attribute.

- **Range:** 0 through 16,777,215

NOTE: Starting in Junos OS Release 17.3R3-3, 18.1R3-S3, and 19.1R1, Junos OS supports a VNI value of 0.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 14.1R2.

RELATED DOCUMENTATION

[Understanding VXLANs | 414](#)

Manually Configuring VXLANs on QFX Series and EX4600 Switches

Examples: Manually Configuring VXLANs on QFX Series and EX4600 Switches

vni-options

IN THIS SECTION

- [Syntax | 1706](#)
- [Hierarchy Level | 1706](#)
- [Description | 1706](#)
- [Required Privilege Level | 1707](#)
- [Release Information | 1707](#)

Syntax

```
vni-options vni vxlan-network-identifier {  
    designated-forwarder-election-hold-time seconds;  
    vrf-target {  
        community;  
        auto;  
        import community-name;  
        export community-name;  
    }  
}
```

Hierarchy Level

```
[edit protocols evpn]  
[edit routing-instances routing-instance-name protocols evpn]
```

Description

Configure a designated forwarder election hold time and specific route targets (RTs) for each VXLAN network identifier (VNI).

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 14.1X53-D30 .

RELATED DOCUMENTATION

[EVPN Multihoming Overview | 96](#)

[Understanding EVPN with VXLAN Data Plane Encapsulation | 398](#)

[Example: Configure an EVPN-VXLAN Centrally-Routed Bridging \(CRB\) Using MX Routers as Spines | 580](#)

[extended-vni-list | 1680](#)

vnid (EVPN)

IN THIS SECTION

- [Syntax | 1707](#)
- [Hierarchy Level | 1708](#)
- [Description | 1708](#)
- [Required Privilege Level | 1708](#)
- [Release Information | 1708](#)

Syntax

```
vnid;
```

Hierarchy Level

```
[edit routing-instances routing-instance-name ],
```

Description

Enables having the same MAC frames across multiple VXLAN segments without traffic crossover.

Multicast in VXLAN is implemented as Layer 3 multicast, in which endpoints subscribe to groups. VNID is a 24-bit virtual network identifier that uniquely identifies the VXLAN segment.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 17.1.

RELATED DOCUMENTATION

[Understanding EVPN with VXLAN Data Plane Encapsulation | 398](#)

[Example: Configure an EVPN-VXLAN Centrally-Routed Bridging \(CRB\) Using MX Routers as Spines | 580](#)

[EVPN Type 5 Route with MPLS encapsulation for EVPN-MPLS | 22](#)

[EVPN Type 5 Route with VXLAN encapsulation for EVPN-VXLAN | 21](#)

vtep-source-interface

IN THIS SECTION

- [Syntax | 1709](#)
- [Hierarchy Level | 1709](#)

- [Description | 1709](#)
- [Options | 1709](#)
- [Required Privilege Level | 1710](#)
- [Release Information | 1710](#)

Syntax

```
vtep-source-interface  
interface-name (inet | inet6 );
```

Hierarchy Level

```
[edit switch-options]  
[edit routing-instances routing-instance-name]
```

Description

Configure a source interface for a Virtual Extensible LAN (VXLAN) tunnel. You must provide the name of a logical interface configured on the loopback interface.

If you specify an IPv4 address (`inet` option) as the VTEP source address, the VXLAN tunnel endpoint (VTEP) encapsulates VXLAN packets with an IPv4 outer header and tunnels the packets through an IPv4 underlay network.

If you specify an IPv6 address (`inet6` option) as the VTEP source address, the VTEP encapsulates VXLAN packets with an IPv6 outer header and tunnels the packets through an IPv6 underlay network.

Options

<i>interface-name</i>	Loopback interface name.
<code>inet</code>	IPv4 source address.
<code>inet6</code>	IPv6 source address.

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 14.1X53-D10.

Support at the [edit routing-instances *routing-instance-name*] hierarchy level introduced in Junos OS Release 17.3.

RELATED DOCUMENTATION

[Understanding VXLANs | 414](#)

[EVPN-VXLAN with an IPv6 Underlay | 643](#)

Manually Configuring VXLANs on QFX Series and EX4600 Switches

Examples: Manually Configuring VXLANs on QFX Series and EX4600 Switches

vxlan

IN THIS SECTION

- [Syntax | 1711](#)
- [Hierarchy Level | 1711](#)
- [Description | 1711](#)
- [Options | 1711](#)
- [Required Privilege Level | 1711](#)
- [Release Information | 1712](#)

Syntax

```
vxlan {
  encapsulate-inner-vlan;
  ingress-node-replication;
  multicast-group;
  ovsdb-managed;
  riot-loopback;
  unreachable-vtep-aging-timer
  vni;
  multicast-group multicast-group;
  mtu mtu;}

```

Hierarchy Level

```
[edit vlan name]
[edit bridge-domains bridge-domain-name]
```

Description

Configure support for Virtual Extensible LANs (VXLANs) on a Juniper Networks device.

Options

multicast-group *multicast-group* Multicast group (IPv4 or IPv6 addresses) registered for VXLAN segment.

mtu *mtu* Specify the maximum transmission unit (MTU) size for a VXLAN packet . The range is from 100 to 65535 bytes.

The remaining statements are explained separately. See [CLI Explorer](#).

Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 14.1X53-D10.

`ingress-node-replication` option added for EVPN VXLAN on QFX5100 switches in Junos OS Release 14.1X53-D30.

`mcast-group` *mcast-group* option added for MX series routers with MPC and MIC interfaces in Junos OS Release 17.2.

`mtu` option added for cRPD in Junos OS Release 21.2R1.

`riot-loopback` option added for EVPN-VXLAN on QFX5210 switches in Junos OS Release 21.3R1.

RELATED DOCUMENTATION

[Understanding VXLANs | 414](#)

[Using a RIOT Loopback Port to Route Traffic in an EVPN-VXLAN Network | 514](#)

Manually Configuring VXLANs on QFX Series and EX4600 Switches

Examples: Manually Configuring VXLANs on QFX Series and EX4600 Switches

vxlan-disable-copy-tos-decap

IN THIS SECTION

- [Syntax | 1713](#)
- [Hierarchy Level | 1713](#)
- [Description | 1713](#)
- [Default | 1713](#)
- [Required Privilege Level | 1714](#)
- [Release Information | 1714](#)

Syntax

```
vxlan-disable-copy-tos-decap
```

Hierarchy Level

```
[edit forwarding-options]
```

Description

Disable copying the Type of Service (ToS) field from the outer IP header in a Virtual Extensible LAN (VXLAN) packet to the inner IP header when de-encapsulating the original packet. The ToS field includes values for differentiated services code points (DSCP) and explicit congestion notification (ECN).

In networks where you configure quality of service (QoS) options, traffic moving through the network might include a DSCP value in the ToS field of the IP header for classifying and policing packets. Similarly, network nodes might use the ECN bits in the ToS field of the IP header to enable end-to-end notification of network congestion so the sender can reduce the transmission rate before nodes must start dropping packets. If you have VXLAN tunnels configured in your network, when encapsulating the original packet in the VXLAN header, some devices (such as the QFX5000 line of switches) copy the DSCP and ECN bits from the original packet's IP header (the inner IP header) to the VXLAN IP header (the outer IP header) by default. At the other end of the tunnel, the same types of devices de-encapsulate the packet and copy the outer IP DSCP and ECN bits back to the inner IP header.

However, some devices (such as the QFX10000 line of switches) don't have the ability to copy ToS field bits upon VXLAN encapsulation and de-encapsulation. As a result, you might see unexpected results if the devices on both ends of the tunnel don't consistently copy the ToS bits during encapsulation and de-encapsulation. For example, the decapsulating node might overwrite the inner IP header DSCP bits in a packet with 000000, which inadvertently lowers the priority of the packet to best effort.

To account for these differences in VXLAN encapsulation behavior, you can configure the `vxlan-disable-copy-tos-encap` statement to disable copying ToS field values from inner to outer IP headers during encapsulation. Similarly, you can configure the `vxlan-disable-copy-tos-decap` statement to disable copying ToS values from the outer IP header back to the inner IP header during de-encapsulation.

Default

Copying ToS bits during VXLAN tunnel de-encapsulation is enabled on supporting devices.

NOTE: Some platforms support copying and disabling copy of only the DSCP bits in the ToS field, and not the ECN bits. Other platforms (such as EX4650 and QFX5120 switches) support copying and disabling copy of both the DSCP and ECN bits. See [Feature Explorer](#) for specific platform and release version support details.

Required Privilege Level

admin—To view this statement in the configuration.

admin-control—To add this statement to a configuration.

Release Information

Statement introduced in Junos OS Release 18.4R2 to disable copying the DSCP bits in the ToS field during VXLAN de-encapsulation.

Support added in Junos OS Release 21.1R1 for copying and disabling copy of ECN bits in the ToS field during VXLAN de-encapsulation.

RELATED DOCUMENTATION

[Understanding VXLANs | 414](#)

[vxlan-disable-copy-tos-encap | 1714](#)

vxlan-disable-copy-tos-encap

IN THIS SECTION

- [Syntax | 1715](#)
- [Hierarchy Level | 1715](#)
- [Description | 1715](#)
- [Default | 1716](#)
- [Required Privilege Level | 1716](#)

Syntax

```
vxlan-disable-copy-tos-encap
```

Hierarchy Level

```
[edit forwarding-options]
```

Description

Disable copying the Type of Service (ToS) field from the IP header of a packet to the outer IP header of a Virtual Extensible LAN (VXLAN) packet when encapsulating the packet. The ToS field includes values for differentiated services code points (DSCP) and explicit congestion notification (ECN).

In networks where you configure quality of service (QoS) options, traffic moving through the network might include a DSCP value in the ToS field of the IP header for classifying and policing packets. Similarly, network nodes might use the ECN bits in the ToS field of the IP header to enable end-to-end notification of network congestion so the sender can reduce the transmission rate before nodes must start dropping packets. If you have VXLAN tunnels configured in your network, when encapsulating the original packet in the VXLAN header, some devices (such as the QFX5000 line of switches) copy the DSCP and ECN bits from the original packet's IP header (the inner IP header) to the VXLAN IP header (the outer IP header) by default. At the other end of the tunnel, the same types of devices de-encapsulate the packet and copy the outer IP DSCP and ECN bits back to the inner IP header.

However, some devices (such as the QFX10000 line of switches) don't have the ability to copy ToS field bits upon VXLAN encapsulation and de-encapsulation. As a result, you might see unexpected results if the devices on both ends of the tunnel don't consistently copy the ToS bits during encapsulation and de-encapsulation. For example, the decapsulating node might overwrite the inner IP header DSCP bits in a packet with 000000, which inadvertently lowers the priority of the packet to best effort.

To account for these differences in VXLAN encapsulation behavior, you can configure the `vxlan-disable-copy-tos-encap` statement to disable copying ToS field values from inner to outer IP headers during encapsulation. Similarly, you can configure the `vxlan-disable-copy-tos-decap` statement to disable copying ToS values from the outer IP header back to the inner IP header during de-encapsulation.

Default

Copying ToS bits during VXLAN tunnel encapsulation is enabled on supporting devices.

NOTE: Some platforms support copying and disabling copy of only the DSCP bits in the ToS field, and not the ECN bits. Other platforms (such as EX4650 and QFX5120 switches) support copying and disabling copy of both the DSCP and ECN bits. See [Feature Explorer](#) for specific platform and release version support details.

Required Privilege Level

admin—To view this statement in the configuration.

admin-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 18.4R2 to disable copying the DSCP bits in the ToS field during VXLAN encapsulation.

Support added in Junos OS Release 21.1R1 for copying and disabling copy of ECN bits in the ToS field during VXLAN encapsulation.

RELATED DOCUMENTATION

[Understanding VXLANs | 414](#)

[vxlan-disable-copy-tos-decap | 1712](#)

vxlan-gbp-profile

IN THIS SECTION

- [Syntax | 1717](#)
- [Hierarchy Level | 1717](#)
- [Description | 1717](#)

- [Default | 1718](#)
- [Required Privilege Level | 1718](#)
- [Release Information | 1718](#)

Syntax

```
vxlan-gbp-profile
```

Hierarchy Level

```
[edit chassis forwarding-options]
```

Description

Enable `vxlan-gbp-profile` on the tunnel termination endpoint in your EVPN-VXLAN deployment to support group-based policies. This setting tells the switch to allocate a share of its resources for L2/L3 group-based policies, whereas otherwise the resources would remain committed for use by all other flows. Note that for switches in a virtual chassis, the device must be rebooted for this setting to apply; for stand-alone switches the packet forwarding engine (PFE) will be restarted.

Group-based policies (GBP) make use of existing layer 3 VXLAN network identifiers (VNI), in conjunction with firewall filter policies, to provide micro-segmentation at the level of device or tag, independent of the underlying network topology. For example, IoT devices typically only need access to specific applications on the network, so GBP can keep this traffic isolated by automatically applying security policies without the need for L2 or L3 lookups or ACLs. As such, GBP provides a new approach to network access control and security that is especially valuable for enterprise campuses.

In addition to enabling `vxlan-gbp-profile` on the tunnel termination endpoint, you need to create firewall rules with match conditions for the endpoint devices you want to segregate. Do this on the EX4400 switch in your topology that is deployed in the role of VXLAN gateway for the access layer.

The following match conditions can be used:

```
gbp-dst-tag
```

```
gbp-src-tag
```

At the `[edit firewall family ethernet-switching filter f1 term t1 then]` level, only `gbp-src-tag` is valid.

Default

Not enabled

Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 21.1R1 for EX4400 Series switches.

RELATED DOCUMENTATION

[Example: Micro and Macro Segmentation using Group Based Policy in a VXLAN | 486](#)

vxlan-routing

IN THIS SECTION

- [Syntax | 1718](#)
- [Hierarchy Level | 1719](#)
- [Description | 1719](#)
- [Required Privilege Level | 1719](#)
- [Release Information | 1719](#)

Syntax

```
vxlan-routing {  
    interface-num integer;  
    loopback-port;
```

```

next-hop (VXLAN Routing) integer;
overlay-ecmp;
}

```

Hierarchy Level

```
[edit forwarding-options]
```

Description

Configure Virtual Extensible LAN (VXLAN) routing options on a device.

The remaining statements are explained separately. See [CLI Explorer](#).

Required Privilege Level

interface—To view this statement in the configuration.

interface-control—To add this statement to the configuration.

Release Information

Statement introduced in Junos OS Release 17.3R1.

overlay-ecmp statement introduced in Junos OS Release 17.4R1.

loopback-port statement introduced in Junos OS Release 21.3R1 for QFX5210 switches.

EVPN Operational Commands

IN THIS CHAPTER

- [clear bridge mac-ip-table | 1722](#)
- [clear ethernet-switching mac-ip-table | 1723](#)
- [clear ethernet-switching evpn arp-table | 1724](#)
- [clear ethernet-switching evpn nd-statistics | 1726](#)
- [clear ethernet-switching evpn nd-table | 1727](#)
- [clear evpn duplicate-mac-suppression | 1729](#)
- [clear evpn mac-ip-table | 1730](#)
- [clear evpn nd-table | 1731](#)
- [clear evpn statistics | 1733](#)
- [clear loop-detect enhanced interface | 1734](#)
- [clear mac-vrf forwarding mac-ip-table | 1736](#)
- [clear mac-vrf forwarding mac-learning-log | 1738](#)
- [clear mac-vrf forwarding mac-table | 1739](#)
- [clear mac-vrf forwarding recovery-timeout | 1741](#)
- [clear mac-vrf forwarding redundancy-group | 1742](#)
- [show bridge mac-ip-table | 1743](#)
- [show ethernet-switching mac-ip-table | 1748](#)
- [show ethernet-switching evpn nd-statistics | 1755](#)
- [show ethernet-switching evpn nd-table | 1758](#)
- [show ethernet-switching flood | 1762](#)
- [show ethernet-switching mgrp-policy | 1767](#)
- [show ethernet-switching table | 1773](#)
- [show ethernet-switching-vxlan-tunnel-end-point esi | 1806](#)
- [show-ethernet-switching-vxlan-tunnel-end-point source | 1809](#)
- [show ethernet-switching vxlan-tunnel-end-point svlnh | 1811](#)
- [show evpn arp-table | 1815](#)

- [show evpn database | 1819](#)
- [show evpn flood | 1825](#)
- [show evpn igmp-snooping database | 1827](#)
- [show evpn igmp-snooping proxy | 1831](#)
- [show evpn instance | 1834](#)
- [show evpn ip-prefix-database | 1852](#)
- [show evpn l3-context | 1862](#)
- [show evpn mac-ip-table | 1866](#)
- [show evpn mac-table | 1870](#)
- [show evpn mld-snooping database | 1872](#)
- [show evpn multicast-snooping assisted-replication multihomed-peers | 1876](#)
- [show evpn multicast-snooping assisted-replication next-hops | 1880](#)
- [show evpn multicast-snooping assisted-replication replicators | 1883](#)
- [show evpn multicast-snooping status | 1888](#)
- [show evpn nd-table | 1890](#)
- [show evpn oism | 1894](#)
- [show evpn p2mp | 1896](#)
- [show evpn peer-gateway-macs | 1900](#)
- [show evpn prefix | 1901](#)
- [show evpn vpws-instance | 1903](#)
- [show loop-detect enhanced interface | 1911](#)
- [show mac-vrf forwarding flood | 1917](#)
- [show mac-vrf forwarding flood-group | 1921](#)
- [show mac-vrf forwarding global-information | 1926](#)
- [show mac-vrf forwarding global-mac-count | 1929](#)
- [show mac-vrf forwarding global-mac-ip-count | 1931](#)
- [show mac-vrf forwarding instance | 1932](#)
- [show mac-vrf forwarding instance-mapping | 1938](#)
- [show mac-vrf forwarding interface | 1941](#)
- [show mac-vrf forwarding mac-ip-table | 1946](#)
- [show mac-vrf forwarding mac-table | 1951](#)
- [show mac-vrf forwarding mgrp-policy | 1955](#)

- [show mac-vrf forwarding statistics | 1961](#)
- [show mac-vrf forwarding vlans | 1966](#)
- [show mac-vrf forwarding vxlan-tunnel-end-point esi | 1969](#)
- [show mac-vrf forwarding vxlan-tunnel-end-point remote | 1973](#)
- [show mac-vrf forwarding vxlan-tunnel-end-point svlnh | 1982](#)
- [show mld snooping evpn database | 1985](#)
- [show route forwarding-table | 1989](#)
- [show route table | 2003](#)
- [show vlans evpn nd-table | 2028](#)

clear bridge mac-ip-table

IN THIS SECTION

- [Syntax | 1722](#)
- [Description | 1723](#)
- [Options | 1723](#)
- [Required Privilege Level | 1723](#)
- [Release Information | 1723](#)

Syntax

```
clear bridge mac-ip-table
<address>
<bridge-domain (all | bridge-domain-name)>
<instance instance-name>
<logical-system (all | logical-system-name)>
```

Description

Clear the locally learnt ARP entries for EVPN routing instances where the instance-type is virtual-switch. Remotely learnt entries cannot be cleared

Options

<i>address</i>	(Optional) Clear the specific suppressed MAC address from the MAC-IP address table .
<i>bridge-domain</i> (all <i>bridge-domain-name</i>)	(Optional) Clear the MAC-IP address table for all bridge domains or for a specified bridge domain.
<i>instance</i> <i>instance-name</i>	(Optional) Clear the MAC-IP address table for a specific routing instance.
<i>logical-system</i> (all <i>logical-system-name</i>)	(Optional) Clear the MAC-IP address for all logical systems or on a specific logical system.

Required Privilege Level

maintenance

Release Information

Command introduced in Junos OS Release 17.4R2.

clear ethernet-switching mac-ip-table

IN THIS SECTION

- [Syntax | 1724](#)
- [Description | 1724](#)
- [Options | 1724](#)
- [Required Privilege Level | 1724](#)
- [Release Information | 1724](#)

Syntax

```
clear ethernet-switching mac-ip-table
<address>
<vlan-name (all | vlan-domain-name)>
<instance instance-name>
```

Description

Clear the locally learnt ARP entries for VLANs in routing instances where the instance-type is ethernet switching. Remotely learnt entries cannot be cleared.

Options

- | | |
|---|--|
| <i>address</i> | (Optional) Clear the specific MAC address from the MAC-IP address table . |
| <i>vlan-name (all <i>vlan-domain-name</i>)</i> | (Optional) Clear the MAC-IP address table for all VLANs or for the specified VLAN. |
| <i>instance instance-name</i> | (Optional) Clear the MAC-IP address table for a specific routing instance. |

Required Privilege Level

maintenance

Release Information

Command introduced in Junos OS Release 17.4R2.

clear ethernet-switching evpn arp-table

IN THIS SECTION

- [Syntax | 1725](#)
- [Description | 1725](#)

- [Options | 1725](#)
- [Required Privilege Level | 1725](#)
- [Output Fields | 1725](#)
- [Sample Output | 1725](#)
- [Release Information | 1726](#)

Syntax

```
clear ethernet-switching evpn arp-table
```

Description

Clear the ARP proxy table from the Ethernet VPN (EVPN) for integrated routing and bridging (IRB) interfaces. This command applies to EVPN instances of type `virtual-switch`.

Options

none Clear information about the ARP proxy tables for an EVPN.

Required Privilege Level

clear

Output Fields

When you enter this command, you are provided feedback on the status of your request.

Sample Output

clear ethernet-switching evpn arp-table

```
user@host> clear ethernet-switching evpn arp-table
```


Release Information

Command introduced in Junos OS Release 17.3R1.

clear ethernet-switching evpn nd-statistics

IN THIS SECTION

- [Syntax | 1726](#)
- [Description | 1726](#)
- [Options | 1726](#)
- [Required Privilege Level | 1727](#)
- [Output Fields | 1727](#)
- [Sample Output | 1727](#)
- [Release Information | 1727](#)

Syntax

```
clear ethernet-switching evpn nd-statistics
```

Description

Clear the Neighbor Discovery (ND) proxy table statistics for integrated routing and bridging (IRB) interfaces participating in the Ethernet VPN (EVPN). ND proxy is a kernel module that implements IPv6 Neighbor Discovery proxying over Ethernet-like access networks.

Options

none Clear the ND proxy table statistics for IRB interfaces participating in the EVPN.

Required Privilege Level

clear

Output Fields

When you enter this command, the ND proxy table statistics for IRB interfaces participating in the EVPN are cleared.

Sample Output

clear ethernet-switching evpn nd-statistics

```
user@host> clear ethernet-switching evpn nd-statistics
```

Release Information

Command introduced in Junos OS Release 17.3R1.

clear ethernet-switching evpn nd-table

IN THIS SECTION

- [Syntax | 1728](#)
- [Description | 1728](#)
- [Options | 1728](#)
- [Required Privilege Level | 1728](#)
- [Output Fields | 1728](#)
- [Sample Output | 1728](#)
- [Release Information | 1728](#)

Syntax

```
clear ethernet-switching evpn nd-table
```

Description

Clear the Neighbor Discovery (ND) proxy table from the Ethernet VPN (EVPN) for integrated routing and bridging (IRB) interfaces. This command applies to EVPN instances of type `virtual-switch`.

Options

none Clear information about the ND proxy tables for an EVPN.

Required Privilege Level

clear

Output Fields

When you enter this command, you are provided feedback on the status of your request.

Sample Output

```
clear ethernet-switching evpn nd-table
```

```
user@host> clear ethernet-switching evpn nd-table
```

Release Information

Command introduced in Junos OS Release 17.3R1.

clear evpn duplicate-mac-suppression

IN THIS SECTION

- [Syntax | 1729](#)
- [Description | 1729](#)
- [Options | 1729](#)
- [Required Privilege Level | 1730](#)
- [Release Information | 1730](#)

Syntax

```
clear evpn duplicate-mac-suppression
<instance instance | l2-domain-id l2-domain-id | logical-system (all | logical-system-name) | mac-
address mac-address>
```

Description

Clear suppressed duplicate MAC address in the EVPN network.

Options

instance <i>instance</i>	(Optional) Clear suppressed MAC address for a specific routing instance. On MX Series routers , the routing instance can be an EVPN instance or virtual-switch instance. On QFX Series switches, the routing instance can be an EVPN instance, virtual-switch instance or an implicit default-switch instance.
l2-domain-id <i>l2-domain-id</i>	(Optional) Clear suppressed MAC address for a specific L2 domain.
logical-system (all <i>logical-system-name</i>)	(Optional) Clear suppressed MAC address on all logical systems or on a specific logical system.
mac-address <i>mac-address</i>	(Optional) Clear a specific suppressed MAC address.

Required Privilege Level

maintenance

Release Information

Command introduced in Junos OS Release 17.4.

RELATED DOCUMENTATION

[Overview of MAC Mobility | 16](#)

[Changing Duplicate MAC Address Detection Settings | 19](#)

clear evpn mac-ip-table

IN THIS SECTION

- [Syntax | 1730](#)
- [Description | 1731](#)
- [Options | 1731](#)
- [Required Privilege Level | 1731](#)
- [Release Information | 1731](#)

Syntax

```
clear evpn mac-ip-table  
<address>  
<instance instance-name>  
<logical-system (all | logical-system-name)>
```

Description

Clear the locally learned ARP entries for routing instances where the instance-type is evpn. Remotely learned entries cannot be cleared.

Options

<i>address</i>	(Optional) Clear the specific MAC address from the MAC-IP address table.
<i>instance instance-name</i>	(Optional) Clear the MAC-IP address table for a specific routing instance.
<i>logical-system (all logical-system-name)</i>	(Optional) Clear the MAC-IP address for all logical systems or on a specific logical system.

Required Privilege Level

maintenance

Release Information

Command introduced in Junos OS Release 17.4R2.

clear evpn nd-table

IN THIS SECTION

- [Syntax | 1732](#)
- [Description | 1732](#)
- [Options | 1732](#)
- [Required Privilege Level | 1732](#)
- [Output Fields | 1732](#)
- [Sample Output | 1732](#)
- [Release Information | 1732](#)

Syntax

```
clear evpn nd-table
```

Description

Clear the Neighbor Discovery (ND) proxy table from the Ethernet VPN (EVPN). ND proxy is a kernel module that implements IPv6 Neighbor Discovery proxying over Ethernet-like access networks.

Options

none Clear learned ND proxy tables from the EVPN.

Required Privilege Level

clear

Output Fields

When you enter this command, you are provided feedback on the status of your request.

Sample Output

```
clear evpn nd-table
```

```
user@host> clear evpn nd-table
```

Release Information

Command introduced in Junos OS Release 16.2.

clear evpn statistics

IN THIS SECTION

- [Syntax | 1733](#)
- [Description | 1733](#)
- [Options | 1733](#)
- [Required Privilege Level | 1733](#)
- [Output Fields | 1734](#)
- [Sample Output | 1734](#)
- [Release Information | 1734](#)

Syntax

```
clear evpn statistics instance evpn-instance  
<logical-system (all | logical-system-name)>
```

Description

Clear EVPN statistics of all interfaces in a routing instance of type EVPN.

Options

<i>evpn-instance</i>	Name of the EVPN routing instance.
logical-system (all <i>logical-system-name</i>)	(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level

maintenance

Output Fields

This command produces no output.

Sample Output

clear evpn statistics

```
user@host> clear evpn statistics instance evpn-instance
```

Release Information

Command introduced in Junos OS Release 17.2.

RELATED DOCUMENTATION

| *clear bridge statistics*

clear loop-detect enhanced interface

IN THIS SECTION

- [Syntax | 1735](#)
- [Description | 1735](#)
- [Options | 1735](#)
- [Required Privilege Level | 1735](#)
- [Sample Output | 1736](#)
- [Release Information | 1736](#)

Syntax

```
clear loop-detect enhanced interface  
<interface-name>
```

Description

Restore a specified interface or all interfaces to their prior state after the device detects a loop and applies a configured action to break the loop.

In an EVPN-VXLAN fabric, you can configure loop detection on leaf devices for server-facing Layer 2 interfaces using the `loop-detect` statement in the `[edit protocols]` hierarchy. With that statement, you specify the action that the leaf device performs on the interface when it detects a loop, such as bringing the interface down. By default, the interface remains in the state imposed by the loop detection action until you run this command to manually restore the prior state. For example, if you configured the loop detection feature to bring the interface down upon detecting a loop, then this command brings the interface up again. You can also configure the `revert-interval` option to set a timer for after the loop condition has been repaired; when the timer expires, the device reverts the interface to its state prior to the loop detection action.

Use this command after you've repaired the loop condition to return the interface or interfaces to their prior state in either of these cases:

- You didn't configure `revert-interval`. In this case, you must run this command to restore the interface state.
- You want to restore the interface to its prior state before the configured `revert-interval` expires.

Options

<i>interface-name</i>	Interface to revert to the state prior to the loop detection action.
	If you don't specify an interface name, this command defaults to restoring the state of all interfaces affected by the loop detection action.

Required Privilege Level

clear

Sample Output

clear loop-detect enhanced interface

```
user@host> clear loop-detect enhanced interface  
  
user@host>
```

clear loop-detect enhanced interface *interface-name*

```
user@host> clear loop-detect enhanced interface xe-0/0/10.0  
  
user@host>
```

Release Information

Command introduced in Junos OS Release 20.4R1.

RELATED DOCUMENTATION

| [loop-detect \(EVPN\)](#) | [1615](#)

clear mac-vrf forwarding mac-ip-table

IN THIS SECTION

- [Syntax](#) | [1737](#)
- [Description](#) | [1737](#)
- [Options](#) | [1737](#)
- [Additional Information](#) | [1737](#)
- [Required Privilege Level](#) | [1737](#)
- [Release Information](#) | [1737](#)

Syntax

```
clear mac-vrf forwarding mac-ip-table
<address>
<instance routing instance>
<ipv4>
<ipv6>
<vlan-name vlan name>
```

Description

Clear all learned media access control (MAC) address to IP address (MAC+IP) mappings from MAC-VRF type routing instances.

Options

none Clear all learned MAC+IP addresses learned through ARP from the device.

ipv4 *ipv4-address* (Optional) Clear the specified (learned) IPv4 address from the device.

ipv6 *ipv6-address* (Optional) Clear the specified (learned) IPv6 address from the device.

instance *mac-vrf-routing -nstance-name* (Optional) Clear MAC addresses from the specified MAC-VRF routing instance.

vlan-name *VLAN-Name / all* (Optional) Clear MAC address learned on the specified VLAN or all VLANs.

Additional Information

On MX Series routers and the EX9200 line of switches, this command is an alias for the `clear bridge mac-ip-table` command.

On QFX Series switches, this command is an alias for the `clear ethernet switching mac-ip-table` command.

Required Privilege Level

clear

Release Information

Command introduced in Junos OS Release 20.4R1.

RELATED DOCUMENTATION

[clear mac-vrf forwarding mac-table | 1739](#)

clear mac-vrf forwarding mac-learning-log

IN THIS SECTION

- [Syntax | 1738](#)
- [Description | 1738](#)
- [Options | 1738](#)
- [Sample Output | 1738](#)
- [Required Privilege Level | 1739](#)
- [Release Information | 1739](#)

Syntax

```
clear mac-vrf forwarding mac-learning-log
```

Description

(QFX Series switches only) Clear the mac learning log. This action applies to all entries in the mac learning log, regardless of source.

Options

There are no options for this command.

Sample Output

The device displays no output when you issue this command.

Required Privilege Level

clear

Release Information

Command introduced in Junos OS Release 20.4R1.

RELATED DOCUMENTATION

[clear mac-vrf forwarding redundancy-group | 1742](#)

clear mac-vrf forwarding mac-table

IN THIS SECTION

- [Syntax | 1739](#)
- [Description | 1740](#)
- [Options | 1740](#)
- [Additional Information | 1740](#)
- [Required Privilege Level | 1740](#)
- [Release Information | 1740](#)

Syntax

```
clear mac-vrf forwarding mac-table  
<address>  
<instance routing instance>  
<interface interface name>  
<isid isis id>  
<persistent-learning>  
<vlan-id vlan id>  
<vlan-name vlan name>
```

Description

Clear all learned media access control (MAC) addresses from the MAC-VRF type routing instances.

Options

none Clear all learned MAC address entries from the device.

address (Optional) Clear the specified MAC address from the device.

instance *routing instance* (Optional) Clear MAC addresses from the specified MAC-VRF routing instance.

interface *interface name* (Optional) Clear the MAC table for the specified interface.

isid *isis id* (Optional) Clear MAC addresses learned on a specified IS-IS ID. Range: (256..16777215).

persistent-learning (Optional) Clear persistent MAC entries.

vlan-id *VLAN ID* (Optional) Clear MAC addresses learned on the specified VLAN id. Range: (0..4095).

vlan-name (all | *VLAN Name*) (Optional) Clear MAC addresses learned on the specified VLAN or all VLAN names.

Additional Information

On MX Series routers and the EX9200 line of switches, this command is an alias for the `clear bridge evpn` command.

On QFX Series switches, this command is an alias for the `clear ethernet switching evpn` command.

Required Privilege Level

clear

Release Information

Command introduced in Junos OS Release 20.4R1.

RELATED DOCUMENTATION

[clear mac-vrf forwarding mac-ip-table | 1736](#)

[clear mac-vrf forwarding redundancy-group | 1742](#)

clear mac-vrf forwarding recovery-timeout

IN THIS SECTION

- [Syntax | 1741](#)
- [Description | 1741](#)
- [Options | 1741](#)
- [Required Privilege Level | 1741](#)
- [Release Information | 1741](#)

Syntax

```
clear mac-vrf forwarding recovery-timeout  
<interface interface name>
```

Description

Clear all port errors that the device registered due to MAC address limits. If you configure MAC limiting with the `shutdown` option, the device shuts down the affected interfaces until you clear the error.

Options

No options exist for this command.

<code>interface <i>interface name</i></code>	(Optional) Clear only those port errors that the device has registered on a specific interface due to MAC address limits.
--	---

Required Privilege Level

clear

Release Information

Command introduced in Junos OS Release 20.4R1.

RELATED DOCUMENTATION

[clear mac-vrf forwarding mac-ip-table | 1736](#)

clear mac-vrf forwarding redundancy-group

IN THIS SECTION

- [Syntax | 1742](#)
- [Description | 1742](#)
- [Options | 1742](#)
- [Required Privilege Level | 1743](#)
- [Release Information | 1743](#)

Syntax

```
clear mac-vrf forwarding redundancy-group  
<redundancy-group-id>  
<arp-statistics>  
<nd-statistics>
```

Description

Clear redundancy group statistics counters within all MAC-VRF routing instances.

Options

none Clear all redundancy group statistics counters.

redundancy-group-id (Optional) Clear the counters for the specified redundancy group.

arp-statistics (Optional) Clear multichassis aggregated-Ethernet ARP synchronization statistics.

nd-statistics (Optional) Clear multichassis aggregated-Ethernet neighbor discovery protocol (ND) synchronization statistics.

Required Privilege Level

clear

Release Information

Command introduced in Junos OS Release 20.4R1.

RELATED DOCUMENTATION

[clear mac-vrf forwarding mac-table | 1739](#)

show bridge mac-ip-table

IN THIS SECTION

- [Syntax | 1743](#)
- [Description | 1744](#)
- [Options | 1744](#)
- [Required Privilege Level | 1744](#)
- [Output Fields | 1744](#)
- [Sample Output | 1746](#)
- [Release Information | 1747](#)

Syntax

```
show bridge mac-ip-table
<bridge-domain (all | bridge-domain-name)> <instance instance-name>
<brief | count | detail | detail | extensive>
<instance>
<kernel >
<logical-system (all | logical-system-name)>
```

Description

Displays the MAC-IP address for all IPv4 (ARP) and IPv6 (ND) bindings for bridge domains on a virtual switch.

Options

bridge-domain (all <i>bridge-domain-name</i>)	(Optional) Display the MAC-IP address entries for all bridge domains or for a specified bridge domain.
brief detail extensive summary	(Optional) Display the specified level of output.
instance <i>instance-name</i>	(Optional) Display the MAC-IP address information for the specified routing instance.
kernel	(Optional) Display the MAC-IP address entries in the kernel
logical-system < <i>logical-system-name</i> >	(Optional) Display the MAC-IP address information for the specified logical system or all logical systems.

Required Privilege Level

maintenance

Output Fields

[Table 54 on page 1744](#) lists the output fields for the `show bridge mac-ip-table` command. Output fields are listed in the approximate order in which they appear.

Table 54: show bridge mac-ip-table Output Fields

Field Name	Field Description	Level of Output
IP address	IP address associated with the EVPN routing instance.	All levels
MAC address	MAC address	All levels

Table 54: show bridge mac-ip-table Output Fields (*Continued*)

Field Name	Field Description	Level of Output
Flags	<p>MAC IP flags: Identifies : statically installed MAC-IP entries.</p> <ul style="list-style-type: none"> • S—Statically installed MAC-IP entries. • D—Dynamic installed MAC-IP address entries. • L—Locally learned MAC-IP address entries • R— MAC-IP address entries that are learnt from a remote device via the control plane. • K—MAC-IP address entries that are installed in the kernel. • RT—Destination Route that corresponds to an installed entry. • AD—Advertised to remote device. • RE—ARP/ND entry has expired and another arp or network solicitation request has been sent. • R0—Router • OV—Override • NAD—The MAC-IP address entries are not being advertised to remote devices. 	Brief, detail
Logical Interface	The logical interface associated with the routing instance.	Brief, detail

Table 54: show bridge mac-ip-table Output Fields (*Continued*)

Field Name	Field Description	Level of Output
Active Source	The source of the learned MAC-IP address entry. Displays the ESI or router ID.	All levels
Routing Instance	Information on the Routing instance where the MAC-IP address entry was learned. <ul style="list-style-type: none"> • Bridging domain • MAC-IP local mask • MAC-IP remote mask • MAC-IP RPD flags • MAC-IP flags 	Extensive
BD ID	Bridge Domain ID	Brief

Sample Output

show bridge mac-ip-table

```

user@host> show bridge mac-ip-table
MAC IP flags (S - Static, D - Dynamic, L - Local , R - Remote, K - Kernel, RT - Dest Route,
              AD - Advt to remote, RE - Re-ARP/ND, RO - Router, OV - Override)
Routing instance : evpn
Bridging domain : bd1
  IP                MAC                Flags                Logical                Active
  address            address            address            Interface            source
  10.1.1.1           00:11:00:00:00:01  DR,K
81.1.1.1
  10.1.1.2           00:22:00:00:00:01  DR,K
81.2.2.2
  10.1.1.3           00:33:00:00:00:01  DL,K,AD            ge-0/0/2.1

```

```

10.1.1.4          00:44:00:00:00:01    DR,K
81.4.4.4

```

show bridge mac-ip-table extensive

```

user@host> show bridge mac-ip-table extensive
IP address: 10.1.1.1
MAC address: 00:11:00:00:00:01
Routing instance: evpn
Bridging domain: bd1
MAC-IP local mask: 0x0000000000000000
MAC-IP remote mask: 0x8000000000000000
MAC-IP RPD flags: 0x00000081
MAC-IP flags: remote,kernel

```

show bridge mac-ip-table kernel

```

user@host> show bridge mac-ip-table kernel

```

BD	IP	MAC	Logical	Flags
id	address	address	Interface	
2	192.168.1.1	00:21:59:aa:77:f0	irb.1	0x00000209
2	192.168.1.100	00:00:5e:00:01:01	irb.1	0x00000609
3	192.168.2.1	00:21:59:aa:77:f0	irb.2	0x00000209
3	192.168.2.100	00:00:5e:00:01:01	irb.2	0x00000609
4	192.168.3.1	00:21:59:aa:77:f0	irb.3	0x00000209
4	192.168.3.100	00:00:5e:00:01:01	irb.3	0x00000609
5	192.168.4.1	00:21:59:aa:77:f0	irb.4	0x00000209
5	192.168.4.100	00:00:5e:00:01:01	irb.4	0x00000609

Release Information

Command introduced in Junos OS Release 17.4R2.

show ethernet-switching mac-ip-table

IN THIS SECTION

- [Syntax | 1748](#)
- [Description | 1748](#)
- [Options | 1748](#)
- [Required Privilege Level | 1749](#)
- [Output Fields | 1749](#)
- [Sample Output | 1751](#)
- [Release Information | 1755](#)

Syntax

```
show ethernet-switching mac-ip-table
<brief | count | detail | detail | extensive>
<instance>
<kernel >
<vlan-name (all | vlan-name)>
```

Description

Displays the MAC-IP address for all IPv4 (ARP) and IPv6 (ND) bindings for VLANs in routing instances where the instance-type is ethernet-switching.

Options

brief | detail | extensive | summary (Optional) Display the specified level of output.

instance *instance-name* (Optional) Display the MAC-IP address information for the specified routing instance.

kernel (Optional) Display the MAC-IP address entries in the kernel

vlan-name (all | *vlan-name*) (Optional) Display the MAC-IP address entries for all VLANs or for a specified VLAN.

Required Privilege Level

maintenance

Output Fields

[Table 55 on page 1749](#) lists the output fields for the `show ethernet-switching mac-ip-table` command. Output fields are listed in the approximate order in which they appear.

Table 55: show ethernet-switching mac-ip-table Output Fields

Field Name	Field Description	Level of Output
IP address	IP address associated with the ethernet switching instance type.	All levels
MAC address	MAC address	All levels

Table 55: show ethernet-switching mac-ip-table Output Fields *(Continued)*

Field Name	Field Description	Level of Output
Flags	<p>MAC IP flags: Identifies : statically installed MAC-IP entries.</p> <ul style="list-style-type: none"> • S—Statically installed MAC-IP entries. • D—Dynamic installed MAC-IP address entries. • L—Locally learned MAC-IP address entries • R— MAC-IP address entries that are learnt from a remote device via the control plane. • K—MAC-IP address entries that are installed in the kernel. • RT—Destination Route that corresponds to an installed entry. • AD—Advertised to remote device. • RE—ARP/ND entry has expired and another arp or network solicitation request has been sent. • R0—Router • OV—Override • UR—Unresolved MAC-IP address entries. • RTS—Device skipped adding an EVPN Type 2 route in favor of an EVPN Type 5 route with a matching prefix (extensive 	Brief, detail

Table 55: show ethernet-switching mac-ip-table Output Fields (Continued)

Field Name	Field Description	Level of Output
	output displays dest_route_skipped).	
Logical Interface	The logical interface associated with the routing instance.	Brief, detail
Active Source	The source of the learned MAC-IP address entry. Displays the ESI or router ID.	All levels
Routing Instance	Information on the Routing instance where the MAC-IP address entry was learned. <ul style="list-style-type: none"> • Bridging domain • MAC-IP local mask • MAC-IP remote mask • MAC-IP RPD flags • MAC-IP flags 	Extensive
BD ID	Bridge Domain ID	Brief

Sample Output

show ethernet-switching mac-ip-table

```

user@host> show ethernet-switching mac-ip-table
vlan-name v100
MAC IP flags (S - Static, D - Dynamic, L - Local , R - Remote, K - Kernel, RT - Dest Route,
              AD - Advt to remote, RE - Re-ARP/ND, RO - Router, OV - Override)
Routing instance : default-
```

```

Bridging domain : v100
  IP          MAC          Flags          Logical          Active
  address     address     address       Interface       source
10.1.1.1      00:11:00:00:00:01    DR,K,RT      vtep.32769
101.1.1.1     2001::192:100:1:1    00:00:10:00:11:00    DR,K,RT      vtep.32769
101.1.1.1     fe80::205:8600:6471:e600    00:00:10:00:11:00    DR,K,RT      vtep.32769
101.1.1.1     10.0.1.2             00:00:10:00:22:00    S,K
101.1.1.1     2001::192:100:1:2     00:00:10:00:22:00    S,K
101.1.1.1     fe80::205:8600:6471:d900    00:00:10:00:22:00    S,K
10.0.1.3      00:00:10:00:33:00    DR,K,RT      vtep.32771
103.1.1.1     2001::192:100:1:3     00:00:10:00:33:00    DR,K,RT      vtep.32771
103.1.1.1     fe80::205:8600:6471:7400    00:00:10:00:33:00    DR,K,RT      vtep.32771
103.1.1.1     10.0.1.4             00:00:10:00:44:00    DR,K,RT      vtep.32770
104.1.1.1     2001::192:100:1:4     00:00:10:00:44:00    DR,K,RT      vtep.32770
104.1.1.1     fe80::205:8600:6471:c000    00:00:10:00:44:00    DR,K,RT      vtep.32770
104.1.1.1     10.0.1.254           40:00:10:11:11:00    S,K          esi.1874
05:00:00:00:64:00:00:00:64:00
10.0.1.254    60:00:10:11:11:00    S,K          esi.1874
05:00:00:00:64:00:00:00:64:00

```

show ethernet-switching mac-ip-table extensive

```

user@host> show ethernet-switching mac-ip-table
extensive
IP address: 10.1.1.1
MAC address: 00:11:00:00:00:01
Routing instance: default
Bridging domain: v100
MAC-IP local mask: 0x0000000000000000
MAC-IP remote mask: 0x8000000000000000

```

MAC-IP RPD flags: 0x00000081

MAC-IP flags: remote,kernel

show ethernet-switching mac-ip-table kernel

```
user@host> show ethernet-switching mac-ip-table
```

kernel

BD	IP	MAC	Logical	Flags
id	address	address	Interface	
2	192.168.1.1	00:21:59:aa:77:f0	irb.1	0x00000209
2	192.168.1.100	00:00:5e:00:01:01	irb.1	0x00000609
3	192.168.2.1	00:21:59:aa:77:f0	irb.2	0x00000209
3	192.168.2.100	00:00:5e:00:01:01	irb.2	0x00000609
4	192.168.3.1	00:21:59:aa:77:f0	irb.3	0x00000209
4	192.168.3.100	00:00:5e:00:01:01	irb.3	0x00000609
5	192.168.4.1	00:21:59:aa:77:f0	irb.4	0x00000209
5	192.168.4.100	00:00:5e:00:01:01	irb.4	0x00000609

show ethernet-switching mac-ip-table vlan-name v1 (with EVPN Type 2 and EVPN Type 5 route coexistence)

```
user@device> show ethernet-switching mac-ip-table vlan-name v1
```

MAC IP flags (S - Static, D - Dynamic, L - Local , R - Remote, Lp - Local Proxy,
Rp - Remote Proxy, K - Kernel, RT - Dest Route, (N)AD - (Not) Advt to remote,
RE - Re-ARP/ND, RO - Router, OV - Override, Ur - Unresolved,
RTS - Dest Route Skipped, RGW - Remote Gateway)

Routing instance : default-switch

Bridging domain : v1

IP	MAC	Flags	Logical	Active
address	address		Interface	source
10.1.1.254	10:10:10:00:00:01	S,K	irb.1	
2001:db8::a:1:1:fffe	20:20:20:00:00:01	S,K,RTS	irb.1	
10.1.1.2	c8:e7:f0:4b:d1:00	DR,K,RTS	vtep.32769	
192.168.22.22				
2001:db8::a:1:1:2	c8:e7:f0:4b:d1:00	DR,K,RTS	vtep.32769	
192.168.22.22				
fe80::cae7:f000:14b:d100	c8:e7:f0:4b:d1:00	DR,K,RT	vtep.32769	
192.168.22.22				

10.1.1.1	dc:38:e1:e0:30:c0	S,K	irb.1
2001:db8::a:1:1:1	dc:38:e1:e0:30:c0	S,K	irb.1
fe80::de38:e100:1e0:30c0	dc:38:e1:e0:30:c0	S,K	irb.1

show ethernet-switching mac-ip-table extensive (with EVPN Type 2 and EVPN Type 5 route coexistence)

```
user@device> show ethernet-switching mac-ip-table extensive c8:e7:f0:4b:d1:00

IP address: 10.1.1.2
MAC address: c8:e7:f0:4b:d1:00
Routing instance: default-switch
Bridging domain: v1
Logical interface: vtep.32769
IP address          MAC address          Flags          Logical Interface          Active source
192.168.22.22
MAC-IP local mask: 0x0000000000000000
MAC-IP remote mask: 0x0000000000000000
MAC-IP remote-proxy mask: 0x0000000000000000
MAC-IP RPD flags: 0x08000081
MAC-IP flags: remote,kernel,dest_route_skipped

IP address: 2001:db8::a:1:1:2
MAC address: c8:e7:f0:4b:d1:00
Routing instance: default-switch
Bridging domain: v1
Logical interface: vtep.32769
IP address          MAC address          Flags          Logical Interface          Active source
192.168.22.22
MAC-IP local mask: 0x0000000000000000
MAC-IP remote mask: 0x0000000000000000
MAC-IP remote-proxy mask: 0x0000000000000000
MAC-IP RPD flags: 0x08000101
MAC-IP flags: remote,kernel,dest_route_skipped

IP address: fe80::cae7:f000:14b:d100
MAC address: c8:e7:f0:4b:d1:00
```

```
Routing instance: default-switch
Bridging domain: v1
Logical interface: vtep.32769
IP address          MAC address          Flags          Logical Interface          Active source
192.168.22.22
MAC-IP local mask: 0x0000000000000000
MAC-IP remote mask: 0x0000000000000000
MAC-IP remote-proxy mask: 0x0000000000000000
MAC-IP RPD flags: 0x08000101
MAC-IP flags: remote, kernel, dest_route
. . .
```

Release Information

Command introduced in Junos OS Release 17.4R2.

show ethernet-switching evpn nd-statistics

IN THIS SECTION

- [Syntax | 1756](#)
- [Description | 1756](#)
- [Options | 1756](#)
- [Required Privilege Level | 1756](#)
- [Output Fields | 1756](#)
- [Sample Output | 1757](#)
- [Release Information | 1757](#)

Syntax

```
show ethernet-switching evpn nd-statistics
<brief | count | detail | extensive | summary | display xml>
```

Description

Show Neighbor Discovery (ND) proxy table statistics for integrated routing and bridging (IRB) interfaces participating in the Ethernet VPN (EVPN). ND proxy is a kernel module that implements IPv6 Neighbor Discovery proxying over Ethernet-like access networks.

Options

- none** Display information about ND proxy table statistics for all IRB interfaces participating in the EVPN.
- brief | count | detail | extensive | summary | display xml** (Optional) Display the specified level of output.

Required Privilege Level

view

Output Fields

[Table 56 on page 1756](#) lists the output fields for the `show ethernet-switching evpn nd-statistics` command. Output fields are listed in the approximate order in which they appear.

Table 56: show ethernet-switching evpn nd-statistics Output Fields

Field Name	Field Description	Level of Output
Interface	IRB interface on which the statistics has been applied	All levels
EVPN	Name of the EVPN	All levels
Bridge domain	Name of the bridge domain for the EVPN	All levels

Table 56: show ethernet-switching evpn nd-statistics Output Fields (*Continued*)

Field Name	Field Description	Level of Output
ND routes add received	Total number of additional ND routes received	All levels
ND routes del received	Total number of deleted ND routes received	All levels
ND routes dropped	Total number of ND routes dropped	All levels
MAC+IPv6s sent to peer	Total number of MAC and IPv6 addresses sent to peer device	All levels

Sample Output

show ethernet-switching evpn nd-statistics

```

user@host> show ethernet-switching evpn nd-statistics
interface irb.0

Interface : irb.0      EVPN : evpn1    Bridge domain: vlan10
ND routes add received      : 2
ND routes del received      : 0
ND routes dropped           : 0
MAC+IPv6s sent to peer      : 3

```

Release Information

Command introduced in Junos OS Release 17.3R1.

show ethernet-switching evpn nd-table

IN THIS SECTION

- [Syntax | 1758](#)
- [Description | 1758](#)
- [Options | 1758](#)
- [Required Privilege Level | 1759](#)
- [Output Fields | 1759](#)
- [Sample Output | 1760](#)
- [Release Information | 1761](#)

Syntax

```
show ethernet-switching evpn nd-table  
<brief | detail | extensive>  
<count>  
<instance-name>  
<mac-address>  
<vlan-name>
```

Description

Display information about INET entries associated with MAC addresses learned through Network Discovery Protocol (NDP).

Options

- | | |
|-----------------------------------|--|
| none | Display information for all INET entries. |
| brief detail extensive | (Optional) Display the specified level of output. |
| count | (Optional) Display the number of INET addresses learned in a routing instance. |
| instance | (Optional) Display information for a specified instance. |

- mac-address** (Optional) Display information for a specified MAC address.
- vlan-name (all)** (Optional) Display information for a specified VLAN or for all VLANs.

Required Privilege Level

view

Output Fields

[Table 57 on page 1759](#) lists the output fields for the `show ethernet-switching evpn nd-table` command. Output fields are listed in the approximate order in which they appear.

Table 57: show ethernet-switching evpn nd-table Output Fields

Field Name	Field Description	Level of Output
INET address	The INET address related to the INET entries that are added to the NDP table.	All levels
MAC address	MAC addresses learned through NDP.	brief, detail, extensive, instance, mac-addressvlan-name,,
Logical Interface	Logical interface associated with the routing instance in which the NDP INET address is learned.	brief, instance, mac-addressvlan-name,,
Routing instance	Routing instance in which the NDP INET address is learned.	all levels
Bridging domain	Bridging domain in which the NDP INET address is learned.	all levels
Learning interface	Interface on which the NDP INET address is learned.	detail, extensive
Count	Indicates the number of NDP INET addresses learned in a routing instance in a bridge domain.	count

Sample Output

show ethernet-switching evpn nd-table brief

```
user@switch> show ethernet-switching evpn nd-table
brief
INET          MAC          Logical      Routing      Bridging
address       address      interface    instance     domain
8002::2       00:05:86:a0:d5:00  irb.0        evpn1        vlan10
```

show ethernet-switching evpn nd-table detail

```
user@switch> show ethernet-switching evpn nd-table
detail
INET address: 8002::2
MAC address: 00:05:86:a0:d5:00
  Routing instance: evpn1
  Bridging domain: vlan10
  Learning interface: irb.0
```

show ethernet-switching evpn nd-table extensive

```
user@switch> show ethernet-switching evpn nd-table
extensive
INET address: 8002::2
MAC address: 00:05:86:a0:d5:00
  Routing instance: evpn1
  Bridging domain: vlan10
  Learning interface: irb.0
```

show ethernet-switching evpn nd-table count

```
user@switch> show ethernet-switching evpn nd-table
count
1 ND INET addresses learned in routing instance evpn1 bridge domain vlan10
```

show ethernet-switching evpn nd-table instance evpn1

```

user@switch> show ethernet-switching evpn nd-table
instance evpn1
INET          MAC          Logical   Routing   Bridging
address       address      interface instance   domain
8002::2
              00:05:86:a0:d5:00
  irb.0       evpn1      vlan10

```

show ethernet-switching evpn nd-table instance 00:05:86:90:bd:f0 (MAC address)

```

user@switch> show ethernet-switching evpn nd-table
INET          MAC          Logical   Routing   Bridging
address       address      interface instance   domain
8002::2
              00:05:86:a0:d5:00
  irb.0       evpn1      __evpn1__

```

show ethernet-switching evpn nd-table vlan-name vlan10

```

user@switch> show ethernet-switching evpn nd-table
vlan-name vlan10

INET          MAC          Logical   Routing   Bridging
address       address      interface instance   domain
8002::2       00:05:86:a0:d5:00
  irb.0       evpn1      vlan10

```

Release Information

Command introduced in Junos OS Release 17.4R2.

NOTE: Starting with Junos OS Release 17.4R2, the `show ethernet-switching evpn nd-table` command replaces the `show vlans evpn nd-table` command.

show ethernet-switching flood

IN THIS SECTION

- [Syntax | 1762](#)
- [Description | 1762](#)
- [Options | 1762](#)
- [Required Privilege Level | 1763](#)
- [Sample Output | 1763](#)
- [Release Information | 1767](#)

Syntax

```
show ethernet-switching flood
<brief | detail | extensive>
<event-queue>
<instance instance-name>
<logical-system logical-system-name>
<route (all-ce-flood | all ve-flood | alt-root-flood | bd-flood | mlp-flood | re-flood)>
<vlan-name vlan-name>
```

Description

(EX Series switches and QFX Series switches only) Display Ethernet-switching flooding information.

Options

none	Display all Ethernet-switching flooding information for all VLANs.
brief detail extensive	(Optional) Display the specified level of output.
event-queue	(Optional) Display the queue of pending Ethernet-switching flood events.
instance <i>instance-name</i>	(Optional) Display Ethernet-switching flooding information for the specified routing instance.

logical-system <i>logical-system-name</i>	(Optional) Display Ethernet-switching flooding information for the specified logical system.
route (all-ce-flood all-ve-flood alt-root-flood bd-flood mlp-flood re-flood)	(Optional) Display the following: <ul style="list-style-type: none"> • all-ce-flood—Display the route for flooding traffic to all customer edge routers or switches if no-local-switching is enabled. • all-ve-flood—Display the route for flooding traffic to all VPLS edge routers or switches if no-local-switching is enabled. • alt-root-flood—Display the Spanning Tree Protocol (STP) alt-root flooding route used for the interface. • bd-flood—Display the route for flooding traffic of a VLAN if no-local-switching is not enabled. • mlp-flood—Display the route for flooding traffic to MAC learning chips. • re-flood—Display the route for Routing Engine flooding to all interfaces.
vlan-name <i>vlan-name</i>	(Optional) Display Ethernet-switching flooding information for the specified VLAN.

Required Privilege Level

view

Sample Output

show ethernet-switching flood

```

user@host> show ethernet-switching flood
Name: __juniper_private1__
CEs: 0
VEs: 0
Name: default-switch
CEs: 9
VEs: 0
VLAN Name: VLAN101
Flood Routes:
  Prefix      Type      Owner      NhType      NhIndex
  0x3057b/51  FLOOD_GRP_COMP_NH __all_ces__ comp      12866

```

```

    0x30004/51 FLOOD_GRP_COMP_NH __re_flood__    comp    12863
VLAN Name: VLAN102
Flood Routes:
  Prefix      Type      Owner      NhType      NhIndex
  0x3057c/51 FLOOD_GRP_COMP_NH __all_ces__    comp    12875
  0x30005/51 FLOOD_GRP_COMP_NH __re_flood__    comp    12872
VLAN Name: VLAN103
Flood Routes:
  Prefix      Type      Owner      NhType      NhIndex
  0x3057d/51 FLOOD_GRP_COMP_NH __all_ces__    comp    12884
  0x30006/51 FLOOD_GRP_COMP_NH __re_flood__    comp    12881

```

show ethernet-switching flood brief

```

user@host> show ethernet-switching flood brief
Name                Active CEs      Active VEs
__juniper_private1__ 0
default-switch      9

```

show ethernet-switching flood detail

```

user@host> show ethernet-switching flood detail
Name: __juniper_private1__
CEs: 0
VEs: 0
Name: default-switch
CEs: 9
VEs: 0
VLAN Name: VLAN101
Flood Routes:
  Prefix      Type      Owner      NhType      NhIndex
  0x3057b/51 FLOOD_GRP_COMP_NH __all_ces__    comp    12866
  0x30004/51 FLOOD_GRP_COMP_NH __re_flood__    comp    12863
VLAN Name: VLAN102
Flood Routes:
  Prefix      Type      Owner      NhType      NhIndex
  0x3057c/51 FLOOD_GRP_COMP_NH __all_ces__    comp    12875
  0x30005/51 FLOOD_GRP_COMP_NH __re_flood__    comp    12872
VLAN Name: VLAN103
Flood Routes:

```

Prefix	Type	Owner	NhType	NhIndex
0x3057d/51	FLOOD_GRP_COMP_NH	__all_ces__	comp	12884
0x30006/51	FLOOD_GRP_COMP_NH	__re_flood__	comp	12881

show ethernet-switching flood extensive

```

user@host> show ethernet-switching flood extensive
Name: __juniper_private1__
CEs: 0
VEs: 0
Name: default-switch
CEs: 9
VEs: 0
VLAN Name: VLAN101
  Flood route prefix: 0x3057b/51
  Flood route type: FLOOD_GRP_COMP_NH
  Flood route owner: __all_ces__
  Flood group name: __all_ces__
  Flood group index: 1
  Nexthop type: comp
  Nexthop index: 12866
  Flooding to:
    Name          Type          NhType      Index
    __all_ces__   Group         comp        12860
    Composition: split-horizon
    Flooding to:
      Name          Type          NhType      Index
      ae20.0        CE           ucst        7605

  Flood route prefix: 0x30004/51
  Flood route type: FLOOD_GRP_COMP_NH
  Flood route owner: __re_flood__
  Flood group name: __re_flood__
  Flood group index: 65534
  Nexthop type: comp
  Nexthop index: 12863
  Flooding to:
    Name          Type          NhType      Index
    __all_ces__   Group         comp        12860
    Composition: split-horizon
    Flooding to:

```


Name	Type	NhType	Index
ae20.0	CE	ucst	7605

VLAN Name: VLAN102

Flood route prefix: 0x3057c/51

Flood route type: FLOOD_GRP_COMP_NH

Flood route owner: __all_ces__

Flood group name: __all_ces__

Flood group index: 1

Nexthop type: comp

Nexthop index: 12875

Flooding to:

Name	Type	NhType	Index
__all_ces__	Group	comp	12869

Composition: split-horizon

Flooding to:

Name	Type	NhType	Index
ae20.0	CE	ucst	7605

Flood route prefix: 0x30005/51

Flood route type: FLOOD_GRP_COMP_NH

Flood route owner: __re_flood__

Flood group name: __re_flood__

Flood group index: 65534

Nexthop type: comp

Nexthop index: 12872

Flooding to:

Name	Type	NhType	Index
__all_ces__	Group	comp	12869

Composition: split-horizon

Flooding to:

Name	Type	NhType	Index
ae20.0	CE	ucst	7605

VLAN Name: VLAN103

Flood route prefix: 0x3057d/51

Flood route type: FLOOD_GRP_COMP_NH

Flood route owner: __all_ces__

Flood group name: __all_ces__

Flood group index: 1

Nexthop type: comp

Nexthop index: 12884

Flooding to:

```
Name          Type      NhType      Index
__all_ces__   Group     comp        12878
  Composition: split-horizon
  Flooding to:
    Name      Type      NhType      Index
    ae20.0    CE        ucst        7605

Flood route prefix: 0x30006/51
Flood route type: FLOOD_GRP_COMP_NH
Flood route owner: __re_flood__
Flood group name: __re_flood__
Flood group index: 65534
Nexthop type: comp
Nexthop index: 12881
  Flooding to:
    Name      Type      NhType      Index
    __all_ces__ Group     comp        12878
      Composition: split-horizon
      Flooding to:
        Name      Type      NhType      Index
        ae20.0    CE        ucst        7605

VLAN Name: VLAN104
```

Release Information

Command introduced in Junos OS Release 12.3R2.

show ethernet-switching mgrp-policy

IN THIS SECTION

- [Syntax | 1768](#)
- [Description | 1768](#)
- [Options | 1769](#)
- [Required Privilege Level | 1769](#)
- [Output Fields | 1769](#)

- [Sample Output | 1770](#)
- [Release Information | 1773](#)

Syntax

```
show ethernet-switching mgrp-policy  
<instance instance-name>  
  <mgrp-id mgrp-id>
```

Description

Display information about logical mesh groups, which are part of a mechanism that Juniper Networks uses to control the flooding of broadcast, unknown unicast, and multicast (BUM) traffic.

Checking mesh groups helps you to monitor the state of the seamless EVPN-VXLAN stitching feature that interconnects EVPN-VXLAN points of delivery (PODs) in a data center, or EVPN-VXLAN data centers. In each of these use cases, Juniper Networks devices that support seamless EVPN-VXLAN stitching act as a spine or super spine device that interconnects the PODs or data centers.

Juniper Networks supports the following default mesh groups for a VLAN. All interfaces that you configure on a spine or super spine device can be assigned to a particular mesh group.

- CE mesh group, to which all directly connected interfaces are assigned.
- VE mesh group, to which all virtual tunnel endpoint (VTEP) interfaces for the local POD or data center are assigned.
- WAN mesh group, to which all remote VTEP interfaces within a POD or data center, or interconnected PODs or data centers are assigned.
- RE mesh group, to which interfaces are typically not assigned.

When a BUM packet enters an interface on the spine or super spine device and that packet needs to be flooded, the mesh group to which the interface belongs is identified. The mesh group associated with a particular VLAN determines the outgoing groups to which the packet should be flooded. For example, if a BUM packet enters an interface assigned to the CE mesh group, the outgoing flood groups might be the following:

- All interfaces in the CE mesh group except the interface through which the packet entered.
- The VE and WAN mesh groups.

Additional factors that determine the outgoing flood groups include the following:

- Whether the spine or super spine device that receives a packet acts as a designated forwarder (DF) or non-DF.
- The local bias feature, also known as split horizon, which eliminates duplication of traffic to its destination in another POD or data center.

Options

none	Display the flood groups associated with all mesh groups.
instance <i>instance-name</i>	(Optional) Display the flood groups associated with the mesh groups for a specified routing or switching instance.
mgrp-id <i>mgrp-id</i>	(Optional) Display interfaces associated with a specified mesh group for a particular routing or switching instance.

Required Privilege Level

admin

Output Fields

[Table 58 on page 1769](#) lists the output fields for the `show ethernet-switching mgrp-policy` command. [Table 59 on page 1770](#) lists the output fields for the `show ethernet-switching mgrp-policy instance` command. [Table 60 on page 1770](#) lists the output fields for the `show ethernet-switching mgrp-policy instance mgrp-id` command. Output fields are listed in the approximate order in which they appear.

Table 58: show ethernet-switching mgrp-policy Output Fields

Field Name	Field Description
Role	<p>Possible roles include either DF (designated forwarder) or NDF (non-designated forwarder).</p> <p>The output is organized by role. Mesh groups listed under each role provide a summary of the groups to which a BUM packet received by a spine or super spine device that acts as a DF or non-DF will be flooded.</p>
Mesh Group	Name of a mesh group. For each mesh group, the output displays the outgoing flood groups to which a BUM packet will be flooded.

Table 59: show ethernet-switching mgrp-policy instance Output Fields

Field Name	Field Description
Role	Possible roles include either DF (designated forwarder) or NDF (non-designated forwarder). The output displays the role for each mesh group.
Mesh Group	Name of a mesh group. For each mesh group, the output displays: <ul style="list-style-type: none"> Flags associated with the mesh group. The outgoing flood groups to which a BUM packet will be flooded.

Table 60: show ethernet-switching mgrp-policy instance mgrp-id Output Fields

Field Name	Field Description
Mesh Group	Name of a mesh group. For the mesh group, the output displays associated flags.
Interfaces	Interfaces associated with the mesh group. When a spine or super spine device floods a BUM packet to the next hop, they will do so through these interfaces.

Sample Output

show ethernet-switching mgrp-policy

```

user@switch> show ethernet-switching mgrp-policy
Role: DF

Mesh Group:  __all_ces__
              __all_ces__
              __ves__
              __wan_flood__
              __icl_flood__
              __etree_leaf__

Mesh Group:  __esi_flood__

Mesh Group:  __etree_leaf__

```

__all_ces__

__ves__

Mesh Group: __icl_flood__

__all_ces__

Mesh Group: __mlp_flood__

Mesh Group: __re_flood__

__all_ces__

__ves__

__wan_flood__

__icl_flood__

__etree_leaf__

Mesh Group: __ves__

__all_ces__

__wan_flood__

__etree_leaf__

Mesh Group: __wan_flood__

__all_ces__

__ves__

Role: NDF

Mesh Group: __all_ces__

__all_ces__

__ves__

__wan_flood__

__icl_flood__

Mesh Group: __ar_flood__

__all_ces__

__ar_flood__

__wan_flood__

Mesh Group: __esi_flood__

Mesh Group: __etree_leaf__

Mesh Group: __icl_flood__

```
Mesh Group:    __mlp_flood__
```

```
Mesh Group:    __re_flood__
                __all_ces__
                __ves__
                __wan_flood__
```

```
Mesh Group:    __ves__
```

```
Mesh Group:    __wan_flood__
                __all_ces__
```

show ethernet-switching mgrp-policy instance

```
user@switch> show ethernet-switching mgrp-policy
instance evpn-vxlan
Role: DF
Mesh Group:    __all_ces__Flags: 0xa ( split-horizon, shared-RT)
                __all_ces__
                __ves__
                __wan_flood__

Role: DF
Mesh Group:    __mlp_flood__Flags: 0xc ( shared-RT, no-local-flooding)

Role: DF
Mesh Group:    __re_flood__Flags: 0x4c ( shared-RT, no-local-flooding, permanent-comp-nh)
                __all_ces__
                __ves__
                __wan_flood__

Role: DF
Mesh Group:    __ves__Flags: 0xc ( shared-RT, no-local-flooding)
                __all_ces__
                __wan_flood__

Role: DF
Mesh Group:    __wan_flood__Flags: 0xc ( shared-RT, no-local-flooding)
                __all_ces__
                __ves__
```

```

Role: Non-DF
Mesh Group:    __ves__Flags: 0xc ( shared-RT, no-local-flooding)

Role: Non-DF
Mesh Group:    __wan_flood__Flags: 0xc ( shared-RT, no-local-flooding)
               __all_ces__

```

show ethernet-switching mgrp-policy instance mgrp-id

```

user@switch> show ethernet-switching mgrp-policy
instance evpn-vxlan mgrp-id 0
Mesh Group:    __ves__Flags: 0xc ( shared-RT, no-local-flooding)
Interfaces
  vtep.32769
  vtep.32770
  vtep.32771
  vtep.32772

```

Release Information

Command introduced in Junos OS Release 20.3R1.

show ethernet-switching table

IN THIS SECTION

- [Syntax \(QFX Series, QFabric, NFX Series and EX4600\) | 1774](#)
- [Syntax \(EX Series\) | 1774](#)
- [Syntax \(EX Series, MX Series and QFX Series\) | 1774](#)
- [Syntax \(SRX Series\) | 1775](#)
- [Description | 1775](#)
- [Options | 1775](#)
- [Additional Information | 1776](#)
- [Required Privilege Level | 1777](#)

- [Output Fields | 1777](#)
- [Sample Output | 1782](#)
- [Sample Output | 1801](#)
- [Sample Output | 1802](#)
- [Sample Output | 1804](#)
- [Sample Output | 1804](#)
- [Release Information | 1805](#)

Syntax (QFX Series, QFabric, NFX Series and EX4600)

```
show ethernet-switching table
<brief | detail | extensive | summary>
<interface interface-name>
<management-vlan>
<sort-by (name | tag)>
<vlan vlan-name>
```

Syntax (EX Series)

```
show ethernet-switching table
<brief | detail | extensive | summary>
<interface interface-name>
<management-vlan>
<persistent-mac <interface interface-name>>
<sort-by (name | tag)>
<vlan vlan-name>
```

Syntax (EX Series, MX Series and QFX Series)

```
show ethernet-switching table
<brief | count | detail | extensive | summary>
<address>
<instance instance-name>
<interface interface-name>
```

```

isid isid
<logical-system logical-system-name>
<persistent-learning (interface interface-name | mac mac-address)>
<address>
<vlan-id (all-vlan | vlan-id)>
<vlan-name (all | vlan-name)>

```

Syntax (SRX Series)

```
show ethernet-switching table (brief | detail | extensive) interface interface-name
```

Description

Displays the Ethernet switching table.

(MX Series routers, EX Series switches only) Displays Layer 2 MAC address information.

Options

For QFX Series, QFabric, NFX Series and EX4600:

none	(Optional) Display brief information about the Ethernet switching table.
brief detail extensive summary	(Optional) Display the specified level of output.
interface <i>interface-name</i>	(Optional) Display the Ethernet switching table for a specific interface.
management-vlan	(Optional) Display the Ethernet switching table for a management VLAN.
persistent-mac <interface <i>interface-name</i>>	(Optional) Display the persistent MAC addresses learned for all interfaces or a specified interface. You can use this command to view entries that you want to clear for an interface that you intentionally disabled.
sort-by (<i>name</i> <i>tag</i>)	(Optional) Display VLANs in ascending order of VLAN IDs or VLAN names.
vlan <i>vlan-name</i>	(Optional) Display the Ethernet switching table for a specific VLAN.

For EX Series, MX Series and QFX Series:

none	Display all learned Layer 2 MAC address information.
brief count detail extensive summary	(Optional) Display the specified level of output.
address	(Optional) Display the specified learned Layer 2 MAC address information.
instance <i>instance-name</i>	(Optional) Display learned Layer 2 MAC addresses for the specified routing instance.
interface <i>interface-name</i>	(Optional) Display learned Layer 2 MAC addresses for the specified interface.
isid <i>isid</i>	(Optional) Display learned Layer 2 MAC addresses for the specified ISID.
logical-system <i>logical-system-name</i>	(Optional) Display Ethernet-switching statistics information for the specified logical system.
persistent-learning (interface <i>interface-name</i> mac <i>mac-address</i>)	(Optional) Display dynamically learned MAC addresses that are retained despite device restarts and interface failures for a specified interface, or information about a specified MAC address.
vlan-id (all-vlan <i>vlan-id</i>)	(Optional) Display learned Layer 2 MAC addresses for all VLANs or for the specified VLAN.
vlan-name (all <i>vlan-name</i>)	(Optional) Display learned Layer 2 MAC addresses for all VLANs or for the specified VLAN.

For SRX Series:

- **none**—(Optional) Display brief information about the Ethernet switching table.
- **brief | detail | extensive**—(Optional) Display the specified level of output.
- **interface-name**—(Optional) Display the Ethernet switching table for a specific interface.

Additional Information

When Layer 2 protocol tunneling is enabled, the tunneling MAC address 01:00:0c:cd:cd:d0 is installed in the MAC table. When the Cisco Discovery Protocol (CDP), Spanning Tree Protocol (STP), or VLAN Trunk Protocol (VTP) is configured for Layer 2 protocol tunneling on an interface, the corresponding protocol MAC address is installed in the MAC table.

Required Privilege Level

view

Output Fields

For QFX Series, QFabric, NFX Series and EX4600:

The following table lists the output fields for the `show ethernet-switching table` command on QFX Series, QFabric, NFX Series and EX4600. Output fields are listed in the approximate order in which they appear.

Table 61: show ethernet-switching table Output Fields on QFX, NFX Series and EX4600

Field Name	Field Description	Level of Output
VLAN	Name of a VLAN.	All levels
MAC address	MAC address associated with the VLAN.	All levels
Type	Type of MAC address: <ul style="list-style-type: none"> • static—The MAC address is manually created. • learn—The MAC address is learned dynamically from a packet's source MAC address. • flood—The MAC address is unknown and flooded to all members. 	All levels
Age	Time remaining before the entry ages out and is removed from the Ethernet switching table.	All levels
Interfaces	Interface associated with learned MAC addresses or with the All-members option (flood entry).	All levels
Learned	For learned entries, the time at which the entry was added to the Ethernet switching table.	detail, extensive

For EX Series switches:

The following table lists the output fields for the `show ethernet-switching table` command on EX Series switches. Output fields are listed in the approximate order in which they appear.

Table 62: show ethernet-switching table Output Fields on EX Series Switches

Field Name	Field Description	Level of Output
VLAN	The name of a VLAN.	All levels
Tag	The VLAN ID tag name or number.	extensive
MAC or MAC address	The MAC address associated with the VLAN.	All levels
Type	<p>The type of MAC address. Values are:</p> <ul style="list-style-type: none"> • static—The MAC address is manually created. • learn—The MAC address is learned dynamically from a packet's source MAC address. • flood—The MAC address is unknown and flooded to all members. • persistent—The learned MAC addresses that will persist across restarts of the switch or interface-down events. 	All levels except persistent-mac
Type	<p>The type of MAC address. Values are:</p> <ul style="list-style-type: none"> • installed—addresses that are in the Ethernet switching table. • uninstalled—addresses that could not be installed in the table or were uninstalled in an interface-down event and will be reinstalled in the table when the interface comes back up. 	persistent-mac
Age	The time remaining before the entry ages out and is removed from the Ethernet switching table.	All levels
Interfaces	Interface associated with learned MAC addresses or All-members (flood entry).	All levels
Learned	For learned entries, the time which the entry was added to the Ethernet switching table.	detail, extensive

Table 62: show ethernet-switching table Output Fields on EX Series Switches *(Continued)*

Field Name	Field Description	Level of Output
Nexthop index	The next-hop index number.	detail, extensive
persistent-mac	installed indicates MAC addresses that are in the Ethernet switching table and uninstalled indicates MAC addresses that could not be installed in the table or were uninstalled in an interface-down event (and will be reinstalled in the table when the interface comes back up).	

For EX Series, MX Series and QFX Series:

The table describes the output fields for the `show ethernet-switching table` command on EX Series, MX Series and QFX Series. Output fields are listed in the approximate order in which they appear.

Table 63: show ethernet-switching table Output fields on EX, MX and QFX Series

Field Name	Field Description
Routing instance	Name of the routing instance.
VLAN name	Name of the VLAN.
MAC address	MAC address or addresses learned on a logical interface.
MAC flags	<p>Status of MAC address learning properties for each interface:</p> <ul style="list-style-type: none"> • S—Static MAC address is configured. • D—Dynamic MAC address is configured. • L—Locally learned MAC address is configured. • SE—MAC accounting is enabled. • NM—Non-configured MAC. • R—Locally learned MAC address is configured. • O—OVSDB learned MAC address is configured.

Table 63: show ethernet-switching table Output fields on EX, MX and QFX Series *(Continued)*

Field Name	Field Description
Age	This field is not supported.
Logical interface	Name of the logical interface. Name of the VTEP logical interface learned over remote VTEP.
SVLBNH/VENH Index	NOTE: This field appears on QFX5XXX switches that support dynamic load balancing in an EVPN-VXLAN network. Next-hop index number associated with the MAC address of a multihomed remote device in an EVPN-VXLAN network. This index number appears when the Logical Interface column displays esi.nnnn. The index number can be an SVLBNH, a VENH, or a remote virtual tunnel endpoint (VTEP). To get more information about SVLBNHs, VENHs, and remote VTEPs, see "show ethernet-switching vxlan-tunnel-end-point svlbnh" on page 1811 .
Active source	IP address or Ethernet segment identifier (ESI) of remote entity on which MAC address is learned.
MAC count	Number of MAC addresses learned on the specific routing instance or interface.
Learning interface	Name of the logical interface on which the MAC address was learned.
Learning VLAN	VLAN ID of the routing instance or VLAN in which the MAC address was learned.
Layer 2 flags	Debugging flags signifying that the MAC address is present in various lists.
Epoch	Spanning-tree-protocolepoch number identifying when the MAC address was learned. Used for debugging.
Sequence number	Sequence number assigned to this MAC address. Used for debugging.
Learning mask	Mask of the Packet Forwarding Engines where this MAC address was learned. Used for debugging.

Table 63: show ethernet-switching table Output fields on EX, MX and QFX Series (Continued)

Field Name	Field Description
IPC generation	Creation time of the logical interface when this MAC address was learned. Used for debugging.

For SRX Series:

[Table 64 on page 1781](#) lists the output fields for the `show ethernet-switching table` command. Output fields are listed in the approximate order in which they appear.

Table 64: show ethernet-switching table Output Fields on SRX Series

Field Name	Field Description
VLAN	The name of a VLAN.
MAC address	The MAC address associated with the VLAN.
Type	<p>The type of MAC address. Values are:</p> <ul style="list-style-type: none"> • static—The MAC address is manually created. • learn—The MAC address is learned dynamically from a packet's source MAC address. • flood—The MAC address is unknown and flooded to all members.
Age	The time remaining before the entry ages out and is removed from the Ethernet switching table.
Interfaces	Interface associated with learned MAC addresses or All-members (flood entry).
Learned	For learned entries, the time which the entry was added to the Ethernet switching table.

Sample Output

show ethernet-switching table (Enhanced Layer 2 Software on QFX Series, QFabric, NFX Series and EX460)

```
user@switch> show ethernet-switching table
```

MAC flags (S - static MAC, D - dynamic MAC, L - locally learned, P - Persistent static
SE - statistics enabled, NM - non configured MAC, R - remote PE MAC, O - ovsdb MAC)

Ethernet switching table : 2 entries, 2 learned

Routing instance : default-switch

Vlan	MAC	MAC	Age	Logical
name	address	flags		interface
vlan1	b0:c6:9a:ca:3c:01	D	-	ae1.0
vlan1	b0:c6:9a:ca:3c:03	D	-	ae1.0

MAC flags (S - static MAC, D - dynamic MAC, L - locally learned, P - Persistent static
SE - statistics enabled, NM - non configured MAC, R - remote PE MAC, O - ovsdb MAC)

Ethernet switching table : 2 entries, 2 learned

Routing instance : default-switch

Vlan	MAC	MAC	Age	Logical
name	address	flags		interface
vlan10	b0:c6:9a:ca:3c:01	D	-	ae1.0
vlan10	b0:c6:9a:ca:3c:03	D	-	ae1.0

MAC flags (S - static MAC, D - dynamic MAC, L - locally learned, P - Persistent static
SE - statistics enabled, NM - non configured MAC, R - remote PE MAC, O - ovsdb MAC)

Ethernet switching table : 2 entries, 2 learned

Routing instance : default-switch

Vlan	MAC	MAC	Age	Logical
name	address	flags		interface
vlan2	b0:c6:9a:ca:3c:01	D	-	ae1.0
vlan2	b0:c6:9a:ca:3c:03	D	-	ae1.0

MAC flags (S - static MAC, D - dynamic MAC, L - locally learned, P - Persistent static
SE - statistics enabled, NM - non configured MAC, R - remote PE MAC, O - ovsdb MAC)

Ethernet switching table : 2 entries, 2 learned

Routing instance : default-switch

Vlan	MAC	MAC	Age	Logical
name	address	flags		interface
vlan3	b0:c6:9a:ca:3c:01	D	-	ae1.0
vlan3	b0:c6:9a:ca:3c:03	D	-	ae1.0

MAC flags (S - static MAC, D - dynamic MAC, L - locally learned, P - Persistent static
SE - statistics enabled, NM - non configured MAC, R - remote PE MAC, O - ovsdb MAC)

Ethernet switching table : 2 entries, 2 learned

Routing instance : default-switch

Vlan	MAC	MAC	Age	Logical
name	address	flags		interface
vlan4	b0:c6:9a:ca:3c:01	D	-	ae1.0
vlan4	b0:c6:9a:ca:3c:03	D	-	ae1.0

show ethernet-switching table (QFX Series switches, SVLBNH/VENH field)

user@switch> **show ethernet-switching table**

MAC flags (S - static MAC, D - dynamic MAC, L - locally learned, P - Persistent static
SE - statistics enabled, NM - non configured MAC, R - remote PE MAC, O - ovsdb MAC)

Ethernet switching table : 16 entries, 16 learned Routing instance : default-switch

Vlan	MAC	MAC	Logical	SVLBNH/	Active
name	address	flags	interface	VENH Index	source
vlanBLACK	00:00:5e:00:53:01	DR	esi.1773	1782	
05:00:00:02:9a:00:00:00:68:00					
vlanBLACK	00:00:5e:00:53:02	DR	esi.1773	1782	
05:00:00:02:9a:00:00:00:68:00					
vlanBLACK	00:00:5e:00:53:80	D	vtep.32772		
10.0.0.1					
vlanBLACK	00:00:5e:00:53:00	D	vtep.32769		
10.0.1.1					
vlanBLUE	00:00:5e:00:53:01	DR	esi.1772	1782	
05:00:00:02:9a:00:00:00:66:00					
vlanBLUE	00:00:5e:00:53:02	DR	esi.1772	1782	
05:00:00:02:9a:00:00:00:66:00					
vlanBLUE	00:00:5e:00:53:80	D	vtep.32772		

```

10.0.0.1
  vlanBLUE      00:00:5e:00:53:00  D      vtep.32769
10.0.1.1
  vlanGREEN     00:00:5e:00:53:01  DR      esi.1774      1782
05:00:00:02:9a:00:00:00:67:00
  vlanGREEN     00:00:5e:00:53:02  DR      esi.1774      1782
05:00:00:02:9a:00:00:00:67:00
  vlanGREEN     00:00:5e:00:53:80  D      vtep.32772
10.0.0.1
  vlanGREEN     00:00:5e:00:53:00  D      vtep.32769
10.0.1.1
  vlanRED       00:00:5e:00:53:01  DR      esi.1771      1782
05:00:00:02:9a:00:00:00:65:00
  vlanRED       00:00:5e:00:53:02  DR      esi.1771      1782
05:00:00:02:9a:00:00:00:65:00
  vlanRED       00:00:5e:00:53:80  D      vtep.32772
10.0.0.1

```

show ethernet-switching table (QFX Series, QFabric, NFX Series and EX460)(QFX5100, QFX5110, QFX5120, QFX5200, QFX5210, QFX10002, QFX10002-60C, and QFX10016)

```

user@switch> show ethernet-switching table
Ethernet-switching table: 57 entries, 17 learned

```

VLAN	MAC address	Type	Age	Interfaces
F2	*	Flood		- All-members
F2	00:00:05:00:00:03	Learn	0	xe-0/0/44.0
F2	00:19:e2:50:7d:e0	Static		- Router
Linux	*	Flood		- All-members
Linux	00:19:e2:50:7d:e0	Static		- Router
Linux	00:30:48:90:54:89	Learn	0	xe-0/0/47.0
T1	*	Flood		- All-members
T1	00:00:05:00:00:01	Learn	0	xe-0/0/46.0
T1	00:00:5e:00:01:00	Static		- Router
T1	00:19:e2:50:63:e0	Learn	0	xe-0/0/46.0
T1	00:19:e2:50:7d:e0	Static		- Router
T10	*	Flood		- All-members
T10	00:00:5e:00:01:09	Static		- Router
T10	00:19:e2:50:63:e0	Learn	0	xe-0/0/46.0
T10	00:19:e2:50:7d:e0	Static		- Router
T111	*	Flood		- All-members
T111	00:19:e2:50:63:e0	Learn	0	xe-0/0/15.0

```

T111      00:19:e2:50:7d:e0 Static      - Router
T111      00:19:e2:50:ac:00 Learn        0 xe-0/0/15.0
T2        *                          Flood      - All-members
T2        00:00:5e:00:01:01 Static      - Router
T2        00:19:e2:50:63:e0 Learn        0 xe-0/0/46.0
T2        00:19:e2:50:7d:e0 Static      - Router
T3        *                          Flood      - All-members
T3        00:00:5e:00:01:02 Static      - Router
T3        00:19:e2:50:63:e0 Learn        0 xe-0/0/46.0
T3        00:19:e2:50:7d:e0 Static      - Router
T4        *                          Flood      - All-members
T4        00:00:5e:00:01:03 Static      - Router
T4        00:19:e2:50:63:e0 Learn        0 xe-0/0/46.0

```

[output truncated]

show ethernet-switching table (Private VLANs on QFX Series, QFabric, NFX Series and EX460)

```

user@switch> show ethernet-switching table
Ethernet-switching table: 10 entries, 3 learned
  VLAN      MAC address      Type      Age Interfaces
  pvlan     *                          Flood      - All-members
  pvlan     00:10:94:00:00:02 Replicated - xe-0/0/28.0
  pvlan     00:10:94:00:00:35 Replicated - xe-0/0/46.0
  pvlan     00:10:94:00:00:46 Replicated - xe-0/0/4.0
  c2        *                          Flood      - All-members
  c2        00:10:94:00:00:02 Learn        0 xe-0/0/28.0
  c1        *                          Flood      - All-members
  c1        00:10:94:00:00:46 Learn        0 xe-0/0/4.0
  __pvlan_pvlan_xe-0/0/46.0__ *      Flood      - All-members
  __pvlan_pvlan_xe-0/0/46.0__ 00:10:94:00:00:35 Learn        0 xe-0/0/46.0

```

show ethernet-switching table brief (QFX Series, QFabric, NFX Series and EX460)

```

user@switch> show ethernet-switching table brief
Ethernet-switching table: 57 entries, 17 learned
  VLAN      MAC address      Type      Age Interfaces
  F2        *                          Flood      - All-members
  F2        00:00:05:00:00:03 Learn        0 xe-0/0/44.0

```

```

F2          00:19:e2:50:7d:e0 Static      - Router
Linux       *                Flood       - All-members
Linux       00:19:e2:50:7d:e0 Static      - Router
Linux       00:30:48:90:54:89 Learn       0 xe-0/0/47.0
T1          *                Flood       - All-members
T1          00:00:05:00:00:01 Learn       0 xe-0/0/46.0
T1          00:00:5e:00:01:00 Static      - Router
T1          00:19:e2:50:63:e0 Learn       0 xe-0/0/46.0
T1          00:19:e2:50:7d:e0 Static      - Router
T10         *                Flood       - All-members
T10         00:00:5e:00:01:09 Static      - Router
T10         00:19:e2:50:63:e0 Learn       0 xe-0/0/46.0
T10         00:19:e2:50:7d:e0 Static      - Router
T111        *                Flood       - All-members
T111        00:19:e2:50:63:e0 Learn       0 xe-0/0/15.0
T111        00:19:e2:50:7d:e0 Static      - Router
T111        00:19:e2:50:ac:00 Learn       0 xe-0/0/15.0
T2          *                Flood       - All-members
T2          00:00:5e:00:01:01 Static      - Router
T2          00:19:e2:50:63:e0 Learn       0 xe-0/0/46.0
T2          00:19:e2:50:7d:e0 Static      - Router
T3          *                Flood       - All-members
T3          00:00:5e:00:01:02 Static      - Router
T3          00:19:e2:50:63:e0 Learn       0 xe-0/0/46.0
T3          00:19:e2:50:7d:e0 Static      - Router
T4          *                Flood       - All-members
T4          00:00:5e:00:01:03 Static      - Router
T4          00:19:e2:50:63:e0 Learn       0 xe-0/0/46.0

```

[output truncated]

show ethernet-switching table detail (QFX Series, QFabric, NFX Series and EX460)

```

user@switch> show ethernet-switching table detail
Ethernet-switching table: 57 entries, 17 learned
F2, *
  Interface(s): xe-0/0/44.0
  Type: Flood
  Nexthop index: 0

F2, 00:00:05:00:00:03
  Interface(s): xe-0/0/44.0

```

Type: Learn, Age: 0, Learned: 2:03:09
 Nexthop index: 0

F2, 00:19:e2:50:7d:e0

Interface(s): Router
 Type: Static
 Nexthop index: 0

Linux, *

Interface(s): xe-0/0/47.0
 Type: Flood
 Nexthop index: 0

Linux, 00:19:e2:50:7d:e0

Interface(s): Router
 Type: Static
 Nexthop index: 0

Linux, 00:30:48:90:54:89

Interface(s): xe-0/0/47.0
 Type: Learn, Age: 0, Learned: 2:03:08
 Nexthop index: 0

T1, *

Interface(s): xe-0/0/46.0
 Type: Flood
 Nexthop index: 0

T1, 00:00:05:00:00:01

Interface(s): xe-0/0/46.0
 Type: Learn, Age: 0, Learned: 2:03:07
 Nexthop index: 0

T1, 00:00:5e:00:01:00

Interface(s): Router
 Type: Static
 Nexthop index: 0

T1, 00:19:e2:50:63:e0

Interface(s): xe-0/0/46.0
 Type: Learn, Age: 0, Learned: 2:03:07
 Nexthop index: 0

```

T1, 00:19:e2:50:7d:e0
  Interface(s): Router
  Type: Static
  Nexthop index: 0

T10, *
  Interface(s): xe-0/0/46.0
  Type: Flood
  Nexthop index: 0

T10, 00:00:5e:00:01:09
  Interface(s): Router
  Type: Static
  Nexthop index: 0

T10, 00:19:e2:50:63:e0
  Interface(s): xe-0/0/46.0
  Type: Learn, Age: 0, Learned: 2:03:08
  Nexthop index: 0

T10, 00:19:e2:50:7d:e0
  Interface(s): Router
  Type: Static
  Nexthop index: 0

T111, *
  Interface(s): xe-0/0/15.0
  Type: Flood
  Nexthop index: 0
[output truncated]

```

show ethernet-switching table extensive (QFX Series, QFabric, NFX Series and EX460)

```

user@switch> show ethernet-switching table extensive
Ethernet-switching table: 57 entries, 17 learned

F2, *
  Interface(s): xe-0/0/44.0
  Type: Flood
  Nexthop index: 0

F2, 00:00:05:00:00:03

```

Interface(s): xe-0/0/44.0
 Type: Learn, Age: 0, Learned: 2:03:09
 Nexthop index: 0

F2, 00:19:e2:50:7d:e0

Interface(s): Router
 Type: Static
 Nexthop index: 0

Linux, *

Interface(s): xe-0/0/47.0
 Type: Flood
 Nexthop index: 0

Linux, 00:19:e2:50:7d:e0

Interface(s): Router
 Type: Static
 Nexthop index: 0

Linux, 00:30:48:90:54:89

Interface(s): xe-0/0/47.0
 Type: Learn, Age: 0, Learned: 2:03:08
 Nexthop index: 0

T1, *

Interface(s): xe-0/0/46.0
 Type: Flood
 Nexthop index: 0

T1, 00:00:05:00:00:01

Interface(s): xe-0/0/46.0
 Type: Learn, Age: 0, Learned: 2:03:07
 Nexthop index: 0

T1, 00:00:5e:00:01:00

Interface(s): Router
 Type: Static
 Nexthop index: 0

T1, 00:19:e2:50:63:e0

Interface(s): xe-0/0/46.0
 Type: Learn, Age: 0, Learned: 2:03:07
 Nexthop index: 0


```

T1, 00:19:e2:50:7d:e0
  Interface(s): Router
  Type: Static
  Nexthop index: 0

T10, *
  Interface(s): xe-0/0/46.0
  Type: Flood
  Nexthop index: 0

T10, 00:00:5e:00:01:09
  Interface(s): Router
  Type: Static
  Nexthop index: 0

T10, 00:19:e2:50:63:e0
  Interface(s): xe-0/0/46.0
  Type: Learn, Age: 0, Learned: 2:03:08
  Nexthop index: 0

T10, 00:19:e2:50:7d:e0
  Interface(s): Router
  Type: Static
  Nexthop index: 0

T111, *
  Interface(s): xe-0/0/15.0
  Type: Flood
  Nexthop index: 0
[output truncated]

```

show ethernet-switching table interface (QFX Series, QFabric, NFX Series and EX460)

```

user@switch> show ethernet-switching table interface xe-0/0/1
Ethernet-switching table: 1 unicast entries

```

VLAN	MAC address	Type	Age	Interfaces
V1	*	Flood	-	All-members
V1	00:00:05:00:00:05	Learn	0	xe-0/0/1.0

show ethernet-switching table (EX Series switches)

```

user@switch> show ethernet-switching table
Ethernet-switching table: 57 entries, 15 learned, 2 persistent

```

VLAN	MAC address	Type	Age	Interfaces
F2	*	Flood		- All-members
F2	00:00:05:00:00:03	Learn	0	ge-0/0/44.0
F2	00:19:e2:50:7d:e0	Static		- Router
Linux	*	Flood		- All-members
Linux	00:19:e2:50:7d:e0	Static		- Router
Linux	00:30:48:90:54:89	Learn	0	ge-0/0/47.0
T1	*	Flood		- All-members
T1	00:00:05:00:00:01	Persistent	0	ge-0/0/46.0
T1	00:00:5e:00:01:00	Static		- Router
T1	00:19:e2:50:63:e0	Persistent	0	ge-0/0/46.0
T1	00:19:e2:50:7d:e0	Static		- Router
T10	*	Flood		- All-members
T10	00:00:5e:00:01:09	Static		- Router
T10	00:19:e2:50:63:e0	Learn	0	ge-0/0/46.0
T10	00:19:e2:50:7d:e0	Static		- Router
T111	*	Flood		- All-members
T111	00:19:e2:50:63:e0	Learn	0	ge-0/0/15.0
T111	00:19:e2:50:7d:e0	Static		- Router
T111	00:19:e2:50:ac:00	Learn	0	ge-0/0/15.0
T2	*	Flood		- All-members
T2	00:00:5e:00:01:01	Static		- Router
T2	00:19:e2:50:63:e0	Learn	0	ge-0/0/46.0
T2	00:19:e2:50:7d:e0	Static		- Router
T3	*	Flood		- All-members
T3	00:00:5e:00:01:02	Static		- Router
T3	00:19:e2:50:63:e0	Learn	0	ge-0/0/46.0
T3	00:19:e2:50:7d:e0	Static		- Router
T4	*	Flood		- All-members
T4	00:00:5e:00:01:03	Static		- Router
T4	00:19:e2:50:63:e0	Learn	0	ge-0/0/46.0

[output truncated]

show ethernet-switching table brief (EX Series switches)

```

user@switch> show ethernet-switching table brief
Ethernet-switching table: 57 entries, 15 learned, 2 persistent entries

```

VLAN	MAC address	Type	Age	Interfaces
F2	*	Flood		- All-members
F2	00:00:05:00:00:03	Learn	0	ge-0/0/44.0
F2	00:19:e2:50:7d:e0	Static		- Router
Linux	*	Flood		- All-members
Linux	00:19:e2:50:7d:e0	Static		- Router
Linux	00:30:48:90:54:89	Learn	0	ge-0/0/47.0
T1	*	Flood		- All-members
T1	00:00:05:00:00:01	Persistent	0	ge-0/0/46.0
T1	00:00:5e:00:01:00	Static		- Router
T1	00:19:e2:50:63:e0	Persistent	0	ge-0/0/46.0
T1	00:19:e2:50:7d:e0	Static		- Router
T10	*	Flood		- All-members
T10	00:00:5e:00:01:09	Static		- Router
T10	00:19:e2:50:63:e0	Learn	0	ge-0/0/46.0
T10	00:19:e2:50:7d:e0	Static		- Router
T111	*	Flood		- All-members
T111	00:19:e2:50:63:e0	Learn	0	ge-0/0/15.0
T111	00:19:e2:50:7d:e0	Static		- Router
T111	00:19:e2:50:ac:00	Learn	0	ge-0/0/15.0
T2	*	Flood		- All-members
T2	00:00:5e:00:01:01	Static		- Router
T2	00:19:e2:50:63:e0	Learn	0	ge-0/0/46.0
T2	00:19:e2:50:7d:e0	Static		- Router
T3	*	Flood		- All-members
T3	00:00:5e:00:01:02	Static		- Router
T3	00:19:e2:50:63:e0	Learn	0	ge-0/0/46.0
T3	00:19:e2:50:7d:e0	Static		- Router
T4	*	Flood		- All-members
T4	00:00:5e:00:01:03	Static		- Router
T4	00:19:e2:50:63:e0	Learn	0	ge-0/0/46.0

[output truncated]

show ethernet-switching table detail (EX Series switches)

```

user@switch> show ethernet-switching table detail
Ethernet-switching table: 5 entries, 2 learned entries
VLAN: default, Tag: 0, MAC: *, Interface: All-members
  Interfaces:
    ge-0/0/11.0, ge-0/0/20.0, ge-0/0/30.0, ge-0/0/36.0, ge-0/0/3.0
  Type: Flood
  Nexthop index: 1307

VLAN: default, Tag: 0, MAC: 00:1f:12:30:b8:83, Interface: ge-0/0/3.0
  Type: Learn, Age: 0, Learned: 20:09:26
  Nexthop index: 1315

VLAN: v1, Tag: 101, MAC: *, Interface: All-members
  Interfaces:
    ge-0/0/31.0
  Type: Flood
  Nexthop index: 1313

VLAN: v1, Tag: 101, MAC: 00:1f:12:30:b8:89, Interface: ge-0/0/31.0
  Type: Learn, Age: 0, Learned: 20:09:25
  Nexthop index: 1312

VLAN: v2, Tag: 102, MAC: *, Interface: All-members
  Interfaces:
    ae0.0
  Type: Flood
  Nexthop index: 1317

```

show ethernet-switching table extensive (EX Series switches)

```

user@switch> show ethernet-switching table extensive
Ethernet-switching table: 3 entries, 1 learned, 5 persistent entries

VLAN: v1, Tag: 10, MAC: *, Interface: All-members
  Interfaces:
    ge-0/0/14.0, ge-0/0/1.0, ge-0/0/2.0, ge-0/0/3.0, ge-0/0/4.0,
    ge-0/0/5.0, ge-0/0/6.0, ge-0/0/7.0, ge-0/0/8.0, ge-0/0/10.0,
    ge-0/0/0.0

```

```

Type: Flood
Nexthop index: 567

VLAN: v1, Tag: 10, MAC: 00:21:59:c6:93:22, Interface: Router
Type: Static
Nexthop index: 0

VLAN: v1, Tag: 10, MAC: 00:21:59:c9:9a:4e, Interface: ge-0/0/14.0
Type: Learn, Age: 0, Learned: 18:40:50
Nexthop index: 564

```

show ethernet-switching table persistent-mac (EX Series switches)

```

user@switch> show ethernet-switching table persistent-mac

```

VLAN	MAC address	Type	Interface
default	00:10:94:00:00:02	installed	ge-0/0/42.0
default	00:10:94:00:00:03	installed	ge-0/0/42.0
default	00:10:94:00:00:04	installed	ge-0/0/42.0
default	00:10:94:00:00:05	installed	ge-0/0/42.0
default	00:10:94:00:00:06	installed	ge-0/0/42.0
default	00:10:94:00:05:02	uninstalled	ge-0/0/16.0
default	00:10:94:00:06:03	uninstalled	ge-0/0/16.0
default	00:10:94:00:07:04	uninstalled	ge-0/0/16.0

show ethernet-switching table persistent-mac interface ge-0/0/16.0 (EX Series switches)

VLAN	MAC address	Type	Interface
default	00:10:94:00:05:02	uninstalled	ge-0/0/16.0
default	00:10:94:00:06:03	uninstalled	ge-0/0/16.0
default	00:10:94:00:07:04	uninstalled	ge-0/0/16.0

show ethernet-switching table (EX Series, MX Series and QFX Series)

```

user@host> show ethernet-switching table

```

MAC flags (S - static MAC, D - dynamic MAC, L - locally learned
SE - statistics enabled, NM - non configured MAC, R - remote PE MAC)

Routing instance : default-switch

Vlan name	MAC address	MAC flags	Age	Logical interface
VLAN101	88:e0:f3:bb:07:f0	D	-	ae20.0

MAC flags (S - static MAC, D - dynamic MAC, L - locally learned
SE - statistics enabled, NM - non configured MAC, R - remote PE MAC)

Routing instance : default-switch

Vlan name	MAC address	MAC flags	Age	Logical interface
VLAN102	88:e0:f3:bb:07:f0	D	-	ae20.0

MAC flags (S - static MAC, D - dynamic MAC, L - locally learned
SE - statistics enabled, NM - non configured MAC, R - remote PE MAC)

Routing instance : default-switch

Vlan name	MAC address	MAC flags	Age	Logical interface
VLAN103	88:e0:f3:bb:07:f0	D	-	ae20.0

MAC flags (S - static MAC, D - dynamic MAC, L - locally learned
SE - statistics enabled, NM - non configured MAC, R - remote PE MAC)

Routing instance : default-switch

Vlan name	MAC address	MAC flags	Age	Logical interface
VLAN104	88:e0:f3:bb:07:f0	D	-	ae20.0

MAC flags (S - static MAC, D - dynamic MAC, L - locally learned
SE - statistics enabled, NM - non configured MAC, R - remote PE MAC)

Routing instance : default-switch

Vlan name	MAC address	MAC flags	Age	Logical interface
VLAN1101	00:1f:12:32:f5:c1	D	-	ae0.0

MAC flags (S - static MAC, D - dynamic MAC, L - locally learned
SE - statistics enabled, NM - non configured MAC, R - remote PE MAC)

Routing instance : default-switch

Vlan	MAC	MAC	Age	Logical
name	address	flags		interface
VLAN1102	00:1f:12:32:f5:c1	D	-	ae0.0

MAC flags (S - static MAC, D - dynamic MAC, L - locally learned

SE - statistics enabled, NM - non configured MAC, R - remote PE MAC)

Routing instance : default-switch

Vlan	MAC	MAC	Age	Logical
name	address	flags		interface
VLAN1103	00:1f:12:32:f5:c1	D	-	ae0.0

MAC flags (S - static MAC, D - dynamic MAC, L - locally learned

SE - statistics enabled, NM - non configured MAC, R - remote PE MAC)

Routing instance : default-switch

Vlan	MAC	MAC	Age	Logical
name	address	flags		interface
VLAN1104	00:1f:12:32:f5:c1	D	-	ae0.0

MAC flags (S - static MAC, D - dynamic MAC, L - locally learned

SE - statistics enabled, NM - non configured MAC, R - remote PE MAC)

Routing instance : default-switch

Vlan	MAC	MAC	Age	Logical
name	address	flags		interface
VLAN1105	00:1f:12:32:f5:c1	D	-	ae0.0

MAC flags (S - static MAC, D - dynamic MAC, L - locally learned

SE - statistics enabled, NM - non configured MAC, R - remote PE MAC)

Routing instance : default-switch

Vlan	MAC	MAC	Age	Logical
name	address	flags		interface
VLAN1106	00:1f:12:32:f5:c1	D	-	ae0.0

[...output truncated...]

show ethernet-switching table brief

```
user@host> show ethernet-switching table brief
```

```
MAC flags (S - static MAC, D - dynamic MAC, L - locally learned
```

```
SE - statistics enabled, NM - non configured MAC, R - remote PE MAC)
```

```
Routing instance : default-switch
```

Vlan	MAC	MAC	Age	Logical
name	address	flags		interface
VLAN101	88:e0:f3:bb:07:f0	D	-	ae20.0

```
MAC flags (S - static MAC, D - dynamic MAC, L - locally learned
```

```
SE - statistics enabled, NM - non configured MAC, R - remote PE MAC)
```

```
Routing instance : default-switch
```

Vlan	MAC	MAC	Age	Logical
name	address	flags		interface
VLAN102	88:e0:f3:bb:07:f0	D	-	ae20.0

```
MAC flags (S - static MAC, D - dynamic MAC, L - locally learned
```

```
SE - statistics enabled, NM - non configured MAC, R - remote PE MAC)
```

```
Routing instance : default-switch
```

Vlan	MAC	MAC	Age	Logical
name	address	flags		interface
VLAN103	88:e0:f3:bb:07:f0	D	-	ae20.0

```
MAC flags (S - static MAC, D - dynamic MAC, L - locally learned
```

```
SE - statistics enabled, NM - non configured MAC, R - remote PE MAC)
```

```
Routing instance : default-switch
```

Vlan	MAC	MAC	Age	Logical
name	address	flags		interface
VLAN104	88:e0:f3:bb:07:f0	D	-	ae20.0

```
MAC flags (S - static MAC, D - dynamic MAC, L - locally learned
```

```
SE - statistics enabled, NM - non configured MAC, R - remote PE MAC)
```



```

Routing instance : default-switch
  Vlan          MAC          MAC      Age   Logical
  name          address      flags
  VLAN1101      00:1f:12:32:f5:c1  D        -    ae0.0
[...output truncated...]

```

show ethernet-switching table count

```

user@host> show ethernet-switching table count
0 MAC address learned in routing instance default-switch VLAN VLAN1000
ae26.0:1000

1 MAC address learned in routing instance default-switch VLAN VLAN101
ae20.0:101

MAC address count per learn VLAN within routing instance:
  Learn VLAN ID   MAC count   Static MAC count
          101           1             0

1 MAC address learned in routing instance default-switch VLAN VLAN102
ae20.0:102

MAC address count per learn VLAN within routing instance:
  Learn VLAN ID   MAC count   Static MAC count
          102           1             0

1 MAC address learned in routing instance default-switch VLAN VLAN103
ae20.0:103

MAC address count per learn VLAN within routing instance:
  Learn VLAN ID   MAC count   Static MAC count
          103           1             0

1 MAC address learned in routing instance default-switch VLAN VLAN104
ae20.0:104

MAC address count per learn VLAN within routing instance:
  Learn VLAN ID   MAC count   Static MAC count
          104           1             0

0 MAC address learned in routing instance default-switch VLAN VLAN105

```

```

ae20.0:105

0 MAC address learned in routing instance default-switch VLAN VLAN106
ae20.0:106

0 MAC address learned in routing instance default-switch VLAN VLAN107
ae20.0:107

0 MAC address learned in routing instance default-switch VLAN VLAN108
ae20.0:108

0 MAC address learned in routing instance default-switch VLAN VLAN109
ae20.0:109

0 MAC address learned in routing instance default-switch VLAN VLAN110
ae20.0:110

1 MAC address learned in routing instance default-switch VLAN VLAN1101
ae0.0:1101

MAC address count per learn VLAN within routing instance:
  Learn VLAN ID      MAC count      Static MAC count
        1101             1             0

1 MAC address learned in routing instance default-switch VLAN VLAN1102
ae0.0:1102

MAC address count per learn VLAN within routing instance:
  Learn VLAN ID      MAC count      Static MAC count
        1102             1             0
[...output truncated...]

```

show ethernet-switching table extensive

```

user@host> show ethernet-switching table extensive

MAC address: 88:e0:f3:bb:07:f0
  Routing instance: default-switch
VLAN ID: 101
  Learning interface: ae20.0
  Layer 2 flags: in_hash,in_ifd,in_ifl,in_vlan,in_rtt,kernel,in_ifbd

```

```

Epoch: 0                               Sequence number: 2
Learning mask: 0x00000008

MAC address: 88:e0:f3:bb:07:f0
Routing instance: default-switch
VLAN ID: 102
Learning interface: ae20.0
Layer 2 flags: in_hash,in_ifd,in_ifl,in_vlan,in_rtt,kernel,in_ifbd
Epoch: 0                               Sequence number: 2
Learning mask: 0x00000008

MAC address: 88:e0:f3:bb:07:f0
Routing instance: default-switch
VLAN ID: 103
Learning interface: ae20.0
Layer 2 flags: in_hash,in_ifd,in_ifl,in_vlan,in_rtt,kernel,in_ifbd
Epoch: 0                               Sequence number: 2
Learning mask: 0x00000008

MAC address: 88:e0:f3:bb:07:f0
Routing instance: default-switch
VLAN ID: 104
Learning interface: ae20.0
Layer 2 flags: in_hash,in_ifd,in_ifl,in_vlan,in_rtt,kernel,in_ifbd
Epoch: 0                               Sequence number: 2
Learning mask: 0x00000008

MAC address: 00:1f:12:32:f5:c1
Routing instance: default-switch
VLAN ID: 1101
Learning interface: ae0.0
Layer 2 flags: in_hash,in_ifd,in_ifl,in_vlan,in_rtt,kernel,in_ifbd
Epoch: 0                               Sequence number: 2
Learning mask: 0x00000008

MAC address: 00:1f:12:32:f5:c1
Routing instance: default-switch
VLAN ID: 1102
Learning interface: ae0.0
Layer 2 flags: in_hash,in_ifd,in_ifl,in_vlan,in_rtt,kernel,in_ifbd
Epoch: 0                               Sequence number: 2
Learning mask: 0x00000008

```

```

MAC address: 00:1f:12:32:f5:c1
  Routing instance: default-switch
VLAN ID: 1103
  Learning interface: ae0.0
  Layer 2 flags: in_hash,in_ifd,in_ifl,in_vlan,in_rtt,kernel,in_ifbd
  Epoch: 0                      Sequence number: 2
  Learning mask: 0x00000008

MAC address: 00:1f:12:32:f5:c1
  Routing instance: default-switch
VLAN ID: 1104
  Learning interface: ae0.0
  Layer 2 flags: in_hash,in_ifd,in_ifl,in_vlan,in_rtt,kernel,in_ifbd
  Epoch: 0                      Sequence number: 2
  Learning mask: 0x00000008

```

Sample Output

show ethernet-switching table detail (SRX Series)

```

user@host> show ethernet-switching table detail
Ethernet-switching table: 57 entries, 17 learned
F2, *
Interface(s): ge-0/0/44.0
Type: Flood
F2, 00:00:5E:00:53:AC
Interface(s): ge-0/0/44.0
Type: Learn, Age: 0, Learned: 2:03:09
F2, 00:00:5E:00:53:AA
Interface(s): Router
Type: Static
Linux, *
Interface(s): ge-0/0/47.0
Type: Flood
Linux, 00:00:5E:00:53:AB
Interface(s): Router
Type: Static
Linux, 00:00:5E:00:53:AC
Interface(s): ge-0/0/47.0
Type: Learn, Age: 0, Learned: 2:03:08
T1, *

```

```

Interface(s): ge-0/0/46.0
Type: Flood
T1, 00:00:5E:00:53:AD
Interface(s): ge-0/0/46.0
Type: Learn, Age: 0, Learned: 2:03:07
T1, 00:00:5E:00:53:AE
Interface(s): Router
Type: Static
T1, 00:00:5E:00:53:AF
Interface(s): ge-0/0/46.0
Type: Learn, Age: 0, Learned: 2:03:07
T1, 00:00:5E:00:53:AG
Interface(s): Router
Type: Static
T10, *
Interface(s): ge-0/0/46.0
Type: Flood
T10, 00:00:5E:00:53:AH
Interface(s): Router
Type: Static
T10, 00:00:5E:00:53:AI
Interface(s): ge-0/0/46.0
Type: Learn, Age: 0, Learned: 2:03:08
T10, 00:00:5E:00:53:AJ
Interface(s): Router
Type: Static
T111, *
Interface(s): ge-0/0/15.0
Type: Flood
[output truncated]

```

Sample Output

show ethernet-switching table extensive (SRX Series)

```

user@host> show ethernet-switching table extensive
Ethernet-switching table: 57 entries, 17 learned
F2, *
Interface(s): ge-0/0/44.0
Type: Flood
F2, 00:00:5E:00:53:AC

```

```
Interface(s): ge-0/0/44.0
Type: Learn, Age: 0, Learned: 2:03:09
F2, 00:00:5E:00:53:AA
Interface(s): Router
Type: Static
Linux, *
Interface(s): ge-0/0/47.0
Type: Flood
Linux, 00:00:5E:00:53:AB
Interface(s): Router
Type: Static
Linux, 00:00:5E:00:53:AC
Interface(s): ge-0/0/47.0
Type: Learn, Age: 0, Learned: 2:03:08
T1, *
Interface(s): ge-0/0/46.0
Type: Flood
T1, 00:00:5E:00:53:AD
Interface(s): ge-0/0/46.0
Type: Learn, Age: 0, Learned: 2:03:07
T1, 00:00:5E:00:53:AE
Interface(s): Router
Type: Static
T1, 00:00:5E:00:53:AF
Interface(s): ge-0/0/46.0
Type: Learn, Age: 0, Learned: 2:03:07
T1, 00:00:5E:00:53:AG
Interface(s): Router
Type: Static
T10, *
Interface(s): ge-0/0/46.0
Type: Flood
T10, 00:00:5E:00:53:AH
Interface(s): Router
Type: Static
T10, 00:00:5E:00:53:AI
Interface(s): ge-0/0/46.0
Type: Learn, Age: 0, Learned: 2:03:08
T10, 00:00:5E:00:53:AJ
Interface(s): Router
Type: Static
T111, *
Interface(s): ge-0/0/15.0
```

```
Type: Flood
[output truncated]
```

Sample Output

show ethernet-switching table interface ge-0/0/1 (SRX Series)

```
user@host> show ethernet-switching table interface ge-0/0/1
Ethernet-switching table: 1 unicast entries
VLAN      MAC address      Type      Age Interfaces
V1        *                Flood     - All-members
V1        00:00:5E:00:53:AF Learn     0 ge-0/0/1.0
```

Sample Output

show ethernet-switching table vlan-name v100 (QFX Series)

```
user@host> show ethernet-switching table vlan-name v100
MAC flags (S - static MAC, D - dynamic MAC, L - locally learned, P - Persistent static
          SE - statistics enabled, NM - non configured MAC, R - remote PE MAC, O - ovsdb MAC)

Pseudo mac
Ethernet switching table : 11 entries, 11 learned
Routing instance : default-switch
  Vlan      MAC      MAC      Logical      Active
  name      address   flags    interface    source
  v100      00:00:00:00:02:80 S,NM     vtep.1074102274
112.1.1.1
  v100      00:00:00:00:03:80 S,NM     vtep.1074102275
115.1.1.1
  v100      00:00:00:00:04:80 S,NM     vtep.1074102276
116.1.1.1
  v100      00:00:00:00:05:80 S,NM     vtep.1074102277
114.1.1.1
  v100      00:00:00:00:06:80 S,NM     vtep.1074102278
101.1.1.1
  v100      00:00:00:00:07:80 S,NM     vtep.1074102279
102.1.1.1
  v100      00:00:00:00:08:80 S,NM     vtep.1074102280
```

```

103.1.1.1
  v100          00:00:00:00:09:80   S,NM   vtep.1074102281
104.1.1.1
  v100          00:00:00:00:0a:80   S,NM   vtep.1074102282
113.1.1.1

```

Release Information

Command introduced in Junos OS Release 9.0 for EX Series switches.

Command introduced in Junos OS Release 9.5 for SRX Series.

Options **summary**, **management-vlan**, and **vlan *vlan-name*** introduced in Junos OS Release 9.6 for EX Series switches.

Option **sort-by** and field name **tag** introduced in Junos OS Release 10.1 for EX Series switches.

Command introduced in Junos OS Release 11.1 for the QFX Series.

Output for private VLANs introduced in Junos OS Release 12.1 for the QFX Series.

Option **persistent-mac** introduced in Junos OS Release 11.4 for EX Series switches.

Command introduced in Junos OS Release 12.3R2.

Command introduced in Junos OS Release 12.3R2 for EX Series switches.

Options **logical-system**, **persistent-learning**, and **summary** introduced in Junos OS Release 13.2X50-D10 (ELS).

Output for shared VXLAN load balancing next hop (SVLBNH) and VXLAN encapsulated next hop (VENH) introduced in Junos OS Release 20.3R1 for QFX Series switches.

Output for pseudo VTEP logical interfaces introduced in Junos OS Release 20.3R1 for QFX Series switches.

RELATED DOCUMENTATION

[Example: Setting Up Basic Bridging and a VLAN on Switches](#)

[Example: Setting Up Bridging with Multiple VLANs](#)

[Example: Setting Up Basic Bridging and a VLAN for an EX Series Switch](#)

[Example: Setting Up Bridging with Multiple VLANs for EX Series Switches](#)

[Example: Setting Up Q-in-Q Tunneling on EX Series Switches](#)

[Dynamic Load Balancing in an EVPN-VXLAN Network | 455](#)

[clear ethernet-switching table](#)[show ethernet-switching mac-learning-log](#)

show ethernet-switching-vxlan-tunnel-end-point esi

IN THIS SECTION

- [Syntax | 1806](#)
- [Description | 1806](#)
- [Options | 1806](#)
- [Required Privilege Level | 1807](#)
- [Output Fields | 1807](#)
- [Sample Output | 1808](#)
- [Release Information | 1809](#)

Syntax

```
show ethernet-switching vxlan-tunnel-end-point esi  
<esi-identifier esi-identifier>  
<instance instance>  
<svlbnh>
```

Description

Displays the ESI (Ethernet Segment Identifier) information on VXLAN tunnel endpoints.

Options

none Display brief information about VXLAN tunnel endpoint.

esi-identifier *esi-identifier* (Optional) Display information for a specified ethernet segment identified by the Ethernet segment identifier (ESI) value.

instance (Optional) Display information for a specified instance .

instance

svlbnh (Optional) Display the SVLBNH index for a given ESI-identifier and instance. You must use this option in conjunction with both the esi-identifier and instance options. For detailed information on a particular SVLBNH index, see ["show ethernet-switching vxlan-tunnel-end-point svlbnh" on page 1811](#).

Required Privilege Level

admin

Output Fields

Table 1 lists the output fields for the show ethernet-switching vxlan-tunnel-end-point esi command.

Table 65: show ethernet-switching vxlan-tunnel-end-point esi Output Fields

Field Name	Field Description
ESI	Ethernet Segment Identifier.
RTT	Routing instance in which you have configured the ESI.
VLBNH	VXLAN load balance next hop.
INH	Indirect next hop.
ESI-IFL	ESI logical interface.
LOC-IFL	Local logical interface for a given Ethernet Segment
#RVTEPS	Number of remote VTEPs.
RVTEP-IP	IP address of the remote VTEP.
RVTEP-IFL	Index number of remote VTEP.
VENH	VXLAN encapsulated next hop.

Table 65: show ethernet-switching vxlan-tunnel-end-point esi Output Fields (*Continued*)

Field Name	Field Description
MASK-ID	Position of the RVTEP in the ESI mask. Each RVTEP is assigned a unique position.
FLAGS	Internal flags used for debugging. They help to identify the state of the RVTEP.
MAC-COUNT	Count of MACs learned on RVTEP for a given ESI.
SVLBH	Index number of the SVLBH.

Sample Output

show ethernet-switching vxlan-tunnel-end-point esi

```
user@host> show ethernet-switching vxlan-tunnel-end-point esi
```

```

PESI                                RTT                                VLNBH INH    ESI-IFL    LOC-IFL    #RVTEPs
00:10:11:12:13:14:15:16:17:11 default-switch    1758 131077 esi.1758          2
Aliasing
  RVTEP-IP      RVTEP-IFL    VENH    MASK-ID    FLAGS    MAC-COUNT
  111.1.1.1     vtep.32769   1753    0          2        0
  112.1.1.1     vtep.32773   1757    1          2        0
ESI                                RTT                                VLNBH INH    ESI-IFL    LOC-IFL    #RVTEPs
00:10:11:12:13:14:15:16:17:22 default-switch    1768 131080 esi.1768 ae0.0,    1
Aliasing
  RVTEP-IP      RVTEP-IFL    VENH    MASK-ID    FLAGS    MAC-COUNT
  113.1.1.1     vtep.32774   1767    0          2        0
ESI                                RTT                                VLNBH INH    ESI-IFL    LOC-IFL    #RVTEPs
05:00:00:00:64:00:00:00:64:00 default-switch    1762 131079 esi.1762          3
  RVTEP-IP      RVTEP-IFL    VENH    MASK-ID    FLAGS    MAC-COUNT
  101.1.1.1     vtep.32770   1754    0          2        2
  103.1.1.1     vtep.32771   1755    1          2        2
  102.1.1.1     vtep.32772   1756    2          2        2
ESI                                RTT                                VLNBH INH    ESI-IFL    LOC-IFL    #RVTEPs
05:00:00:00:64:00:00:00:c8:00 default-switch    1760 131078 esi.1760          3
  RVTEP-IP      RVTEP-IFL    VENH    MASK-ID    FLAGS    MAC-COUNT
  101.1.1.1     vtep.32770   1754    0          2        2

```

103.1.1.1	vtep.32771	1755	1	2	2
102.1.1.1	vtep.32772	1756	2	2	2

show ethernet-switching vxlan-tunnel-end-point esi esi-identifier instance svlnh

This command is used to retrieve SVLBNH index for a given esi-identifier and instance.

```
user@host> show ethernet-switching vxlan-tunnel-end-point esi esi-identifier
00:20:21:22:23:24:25:26:27:01 instance default-switch svlnh
RTT                ESI                ESI-IFL  VLNBH INH      SVLBNH
default-switch     00:20:21:22:23:24:25:26:27:01 esi.1815  1815  131094  1816
```

Release Information

Command introduced in Junos OS Release 16.1R1.

svlnh option introduced in Junos OS Release 21.2R1.

show-ethernet-switching-vxlan-tunnel-end-point source

IN THIS SECTION

- [Description | 1809](#)
- [Required Privilege Level | 1810](#)
- [Output Fields | 1810](#)
- [Sample Output | 1811](#)
- [Release Information | 1811](#)

Description

Displays the translated VNI information for VXLAN traffic using data center interconnect (DCI) .

Required Privilege Level

admin

Output Fields

Table 66 on page 1810 lists the output fields for the show ethernet-switching vxlan-tunnel-end-point source command.

Table 66: show ethernet-switching vxlan-tunnel-end-point source Output Fields

Field Name	Field Description
Logical System Name	Name of the logical system on which you have configured the MAC-VRF routing instance
Id	Logical system ID
L2-RTT	Layer 2 routing instance name
SVTEP-IP	IP address of the source VTEP
IFL	Logical interface name
L3-Idx	System index number of the associated Layer 3 interface
SVTEP-Mode	Source VTEP mode
ELP-SVTEP-IP	IP address of the egress link protection for the source VTEP
L2-RTT	Layer 2 routing instance name
Bridge Domain	Name of the Bridge Domain.
VNID	VLAN network ID. This is the original VNID in the data center
Translation-VNID	Translation VLAN network ID. This is the translated VNID that is being used in the packet as it exits the data center.

Table 66: show ethernet-switching vxlan-tunnel-end-point source Output Fields *(Continued)*

Field Name	Field Description
MC-Group-IP	Multicast group IP address

Sample Output

show ethernet-switching vxlan-tunnel-end-point source

```

user@host> show ethernet-switching vxlan-tunnel-end-point source
Logical System Name      Id  SVTEP-IP      IFL  L3-Idx  SVTEP-Mode  ELP-SVTEP-IP
<default>                0   193.168.4.1   lo0.0  0
  L2-RTT                    Bridge Domain      VNID  Translation-VNID  MC-
  Group-IP
    MACVRF-mac-vrf-ep-t2-stchd-transl-1 EP-TYPE-2-VLAN-1000+1000 301000 920200      0.0.0.0
    MACVRF-mac-vrf-ep-t2-stchd-transl-1 EP-TYPE-2-VLAN-1001+1001 301001 920201      0.0.0.0
    MACVRF-mac-vrf-ep-t2-stchd-transl-1 EP-TYPE-2-VLAN-1002+1002 301002 920202      0.0.0.0
    MACVRF-mac-vrf-ep-t2-stchd-transl-1 EP-TYPE-2-VLAN-1003+1003 301003 920203      0.0.0.0
    MACVRF-mac-vrf-ep-t2-stchd-transl-1 EP-TYPE-2-VLAN-1004+1004 301004 920204      0.0.0.0
    MACVRF-mac-vrf-ep-t2-stchd-transl-1 EP-TYPE-2-VLAN-1005+1005 301005 920205      0.0.0.0

```

Release Information

Command introduced in Junos OS Release 21.3R1.

show ethernet-switching vxlan-tunnel-end-point svlbnh

IN THIS SECTION

- [Syntax | 1812](#)
- [Description | 1812](#)
- [Options | 1812](#)

- Required Privilege Level | 1812
- Output Fields | 1812
- Sample Output | 1813
- Sample Output | 1814
- Release Information | 1814

Syntax

```
show ethernet-switching vxlan-tunnel-end-point svlbnh
<brief | detail | extensive>
<overflow>
<svlbnh-index index-number>
```

Description

Display information about a shared VXLAN load balancing next hop (SVLBNH) and the virtual tunnel endpoints (VTEPs) associated with the SVLBNH.

Options

- none** (Same as brief) Display information for all SVLBNHs.
- brief | detail | extensive** (Optional) Display the specified level of output.
- overflow** (Optional) Display the pending requests for SVLBNH allocation.
- svlbnh-index *index-number*** (Optional) Display information about the specified SVLBNH index.

Required Privilege Level

admin

Output Fields

Table 67 on page 1813 lists the output fields for the show ethernet-switching vxlan-tunnel-end-point svlbnh command. Output fields are listed in the approximate order in which they appear.

Table 67: show ethernet-switching vxlan-tunnel-end-point svlbnh Output

Field Name	Field Description
SVLBNH <i>index-number</i>	Index number of the SVLBNH.
RVTEP count	Number of remote VTEPs associated with the SVLBNH.
ESI count	Number of Ethernet segment identifiers (ESIs) associated with the SVLBNH.
RVTEP-IP	IP address of the remote VTEP.
RVTEP-IFL	Index number of remote VTEP.
VENH	VXLAN encapsulated next hop (VENH) associated with the SVLBNH.
ESI	ESIs associated with the SVLBNH.
RTT	Routing instance in which you have configured the ESI.

Sample Output

show ethernet-switching vxlan-tunnel-end-point svlbnh detail

```

user@switch> show ethernet-switching vxlan-tunnel-end-point
svlbnh detail
SVLBNH: 1882          RVTEP count: 2          ESI count: 1
  RVTEP-IP            RVTEP-IFL            VENH
  10.0.5.1            vtep.32770            1859
  10.0.3.1            vtep.32769            1835
  ESI                  RTT
  00:00:00:00:00:00:00:01:03 default-switch

SVLBNH: 1879          RVTEP count: 2          ESI count: 1
  RVTEP-IP            RVTEP-IFL            VENH
  10.0.2.1            vtep.32771            1869
  10.0.3.1            vtep.32769            1835

```



```

ESI                                RTT
00:00:00:00:00:00:00:01:01 default-switch

SVLBNH: 1881      RVTEP count: 2      ESI count: 1
RVTEP-IP          RVTEP-IFL          VENH
10.0.2.1          vtep.32771         1869
10.0.5.1          vtep.32770         1859
ESI                                RTT
00:00:00:00:00:00:00:01:02 default-switch

SVLBNH: 1870      RVTEP count: 3      ESI count: 5
RVTEP-IP          RVTEP-IFL          VENH
10.0.2.1          vtep.32771         1869
10.0.5.1          vtep.32770         1859
10.0.3.1          vtep.32769         1835
ESI                                RTT
00:00:00:00:00:00:00:01:04 default-switch
05:00:00:02:9a:00:00:00:65:00 default-switch
05:00:00:02:9a:00:00:00:66:00 default-switch
05:00:00:02:9a:00:00:00:67:00 default-switch
05:00:00:02:9a:00:00:00:68:00 default-switch

```

Sample Output

show ethernet-switching vxlan-tunnel-end-point svlbnh (Specific SVLBNH)

```

user@switch> show ethernet-switching vxlan-tunnel-end-point
svlbnh svlbnh-index 1882
SVLBNH: 1882      RVTEP count: 2      ESI count: 1
RVTEP-IP          RVTEP-IFL          VENH
10.0.5.1          vtep.32770         1859
10.0.3.1          vtep.32769         1835
ESI                                RTT
00:00:00:00:00:00:00:01:03 default-switch

```

Release Information

Command introduced in Junos OS Release 20.3R1.

RELATED DOCUMENTATION

[Dynamic Load Balancing in an EVPN-VXLAN Network | 455](#)

[show ethernet-switching table | 1773](#)

show evpn arp-table

IN THIS SECTION

- [Syntax | 1815](#)
- [Description | 1815](#)
- [Options | 1815](#)
- [Required Privilege Level | 1816](#)
- [Output Fields | 1816](#)
- [Sample Output | 1817](#)
- [Release Information | 1818](#)

Syntax

```
show evpn arp-table  
<address>  
<brief | count | detail | extensive>  
<instance instance-name>  
<logical-system logical-system-name>
```

Description

Show Ethernet VPN (EVPN) Address Resolution Protocol (ARP) entries associated with learned MAC addresses.

Options

none

Display brief information about the EVPN ARP table.

address	(Optional) Display ARP information for the specified MAC address.
brief count detail extensive	(Optional) Display the specified level of output.
instance <i><instance-name></i>	oot(Optional) Display ARP information for the specified routing instance .
logical-system <i><logical-system-name></i>	(Optional) Display ARP information for the specified logical system or all logical systems.

Required Privilege Level

view

Output Fields

[Table 68 on page 1816](#) lists the output fields for the `show evpn arp-table` command. Output fields are listed in the approximate order in which they appear.

Table 68: show evpn arp-table Output Fields

Field Name	Field Description	Level of Output
INET address	The INET address related to the INET entries that are added to the ARP table.	All levels
MAC address	MAC addresses learned through ARP.	brief, detail, extensive, instance, mac-address,,
Logical Interface	Logical interface associated with the routing instance in which the ARP INET address is learned.	brief, instance, mac-address,,
Routing instance	Routing instance in which the ARP INET address is learned.	all levels
Bridging domain	Bridging domain in which the ARP INET address is learned.	all levels
Learning interface	Interface on which the ARP INET address is learned.	detail, extensive

Table 68: show evpn arp-table Output Fields (*Continued*)

Field Name	Field Description	Level of Output
Count	Indicates the number of ARP INET addresses learned in a routing instance in a bridge domain.	count

Sample Output

show evpn arp-table

```
user@host> show evpn arp-table
INET          MAC          Logical    Routing    Bridging
address       address      interface  instance   domain
203.0.113.2   00:05:86:a0:dc:f0  irb.0      evpn1      __evpn1__
```

show evpn arp-table 00:05:86:a0:dc:f0 (MAC address)

```
user@host> show evpn arp-table 00:05:86:a0:dc:f0
INET          MAC          Logical    Routing    Bridging
address       address      interface  instance   domain
203.0.113.2   00:05:86:a0:dc:f0  irb.0      evpn1      __evpn1__
```

show evpn arp-table brief

```
user@host> show evpn arp-table brief
INET          MAC          Logical    Routing    Bridging
address       address      interface  instance   domain
203.0.113.2   00:05:86:a0:dc:f0  irb.0      evpn1      __evpn1__
```

show evpn arp-table detail

```
user@host> show evpn arp-table detail

INET address: 203.0.113.2
```

```
MAC address: 00:05:86:a0:dc:f0
Routing instance: evpn1
Bridging domain: __evpn1__
Learning interface: irb.0
```

show evpn arp-table count

```
user@switch> show evpn arp-table count
1 ARP INET addresses learned in routing instance evpn1 bridge domain __evpn1__
```

show evpn arp-table extensive

```
user@host> show evpn arp-table extensive

INET address: 203.0.113.2
MAC address: 00:05:86:a0:dc:f0
Routing instance: evpn1
Bridging domain: __evpn1__
Learning interface: irb.0
```

show evpn arp-table instance evpn1

```
user@host> show evpn arp-table instance evpn1
```

INET address	MAC address	Logical interface	Routing instance	Bridging domain
203.0.113.2	00:05:86:a0:dc:f0	irb.0	evpn1	__evpn1__

Release Information

Command introduced in Junos OS Release 15.1.

logical system option introduced in Junos OS Release 18.1R1.

show evpn database

IN THIS SECTION

- [Syntax | 1819](#)
- [Syntax for mac-vrf routing | 1819](#)
- [Description | 1820](#)
- [Options | 1820](#)
- [Required Privilege Level | 1820](#)
- [Output Fields | 1821](#)
- [Sample Output | 1823](#)
- [Release Information | 1825](#)

Syntax

```
show evpn database
<extensive>
<instance instance-name>
<interface interface-name>
<logical-system logical-system-name>
<mac-address address>
<neighbor neighbor-name>
<origin origin-name>
<state state-name>
<vlan-id vlan-id>
```

Syntax for mac-vrf routing

```
show mac-vrf routing database
<extensive>
<instance instance-name>
<interface interface-name>
<logical-system logical-system-name>
<mac-address address>
```

```
<neighbor neighbor-name>
<origin origin-name>
<state state-name>
<vlan-id vlan-id>
```

Description

Show Ethernet VPN (EVPN) database information.

When up MEP is configured,, the **show evpn instance** displays the CFM MAC local interface with a naming convention as .local..*number* (for example, .local..8).

Options

none	Display brief information about the EVPN database.
extensive	(Optional) Display detailed information about the EVPN database.
instance <i>instance-name</i>	(Optional) Display MAC addresses from the specified routing instance.
interface <i>interface-name</i>	(Optional) Display the MAC address learned from the specified interface.
logical-system <<i>logical-system-name</i>>	(Optional) Display database information for the specified logical system or all logical systems.
mac-address <i>address</i>	(Optional) Display the specified MAC address.
neighbor <i>neighbor-name</i>	(Optional) Display the MAC address learned from the specified neighbor.
origin <i>origin-name</i>	(Optional) Display the MAC address with the specified origin.
state <i>state-name</i>	(Optional) Display the MAC address with the specified state.
vlan-id <i>vlan-id</i>	(Optional) Display the MAC address with the specified VLAN.

Required Privilege Level

view

Output Fields

[Table 69 on page 1821](#) lists the output fields for the `show evpn database` command. Output fields are listed in the approximate order in which they appear.

Table 69: show evpn database Output Fields

Field Name	Field Description	Level of Output
Instance	Name of the routing instances.	All Levels
VNI	VXLAN network identifier.	All Levels
MAC address	MAC address of the routing instance.	All Levels
Origin	Specific origin of the MAC address.	All Levels
Timestamp	Month, day and time of generated command output.	All Levels
IP address	IP address.	All Levels
Prefix	Prefix address of the VXLAN network identifier.	extensive
Source	Source address of the VXLAN network identifier.	extensive
Rank	Level number of the VXLAN network identifier.	extensive
Status	Status of the of the VXLAN network identifier.	extensive
Remote origin	Specific remote origin of the VXLAN network identifier.	extensive

Table 69: show evpn database Output Fields (Continued)

Field Name	Field Description	Level of Output
Mobility History	Information on a MAC address mobility history: <ul style="list-style-type: none"> • Mobility event time—Date and time of the MAC mobility event • Type—Origin of the learned MAC address (Local or Remote). • Source—Source of the learned MAC address (Local Interface, Remote PE IP address, or ESI) • Seq num—Sequence number associated with the mobility event 	extensive
Mac label	Label advertised by the remote PE device when forwarding unicast traffic. In EVPN-MPLS, it is the MPLS label and in EVPN-VXLAN, it is the VNI .	extensive
Mobility Sequence number	Current sequence number associated with the MAC address. The minimum origin address is the IP address of the PE device with the lowest IP address that is sending a MAC advertisement.	extensive
State	Information collected in the EVPN database from different flag states including Duplicate-Detected, Local-Pinned, and Remote-Pinned.	extensive
IP address	IPv4 or IPv6 address for the MAC address.	extensive
L3 route	Route installed on the EVPN IRB interface.	extensive
L3 context	Name of the routing instance that has the Layer 3 routes installed for an EVPN IRB interface, typically a virtual routing and forward (VRF) routing instance.	extensive

Sample Output

show evpn database

```

user@host> show evpn database
Instance: bd_red
VNI  MAC address      Origin      Timestamp      IP address
50    1a:1b:1b:1b:1b:1b  192.0.2.3   Jun 12 21:57:17
50    1a:1c:1c:1c:1c:1c  ge-2/3/0.0  Jun 12 22:05:37
50    b0:c6:9a:e9:cd:d8  192.0.2.3   Jun 12 21:57:17
50    b0:c6:9a:ea:87:42  ge-2/3/0.0  Jun 12 22:05:38

Instance: evpn-0
VNI  MAC address      Origin      Timestamp      IP address
4000  1a:1a:1a:1a:01:01  ge-2/3/0.10 Jun 12 21:53:01
4000  1a:1a:1a:1a:01:02  ge-2/3/0.10 Jun 12 21:53:01

```

show evpn database extensive

```

user@host> show evpn database extensive
Instance: ALPHA

VN Identifier: 9100, Prefix: 10.0.0.0/24
  Source: 10.255.0.2, Rank: 1, Status: Active
  Timestamp: May 22 17:16:12 (0x555fc6cc)

VN Identifier: 9100, Prefix: 198.51.100.0/24
  Source: 05:00:00:00:64:00:00:23:8c:00, Rank: 1, Status: Active
  Remote origin: 10.255.0.2
  Remote origin: 10.255.0.3
  Timestamp: May 22 17:14:20 (0x555fc65c)

VN Identifier: 9100, Prefix: 192.0.2.0/24
  Source: irb.0, Rank: 1, Status: Active
  Timestamp: May 22 17:16:12 (0x555fc6cc)

VN Identifier: 9100, Prefix: 203.0.113.0/24
  Source: 10.255.0.3, Rank: 1, Status: Active
  Timestamp: May 22 17:16:12 (0x555fc6cc)

```

show evpn database extensive instance

```

user@PE1> show evpn database instance ALPHA mac-address 00:00:00:00:00:01
extensive
Instance: ALPHA

VLAN ID: 100, MAC address: 00:00:00:00:00:01
Nexthop ID: 1048575
Mobility history
  Mobility event time      Type      Source                               Seq num
  Jul 10 16:26:14.920136 Remote  10.255.0.3                          13
  Jul 10 16:26:16.174769 Local   ge-0/0/2.0                         14
  Jul 10 16:26:17.868187 Remote  10.255.0.3                          15
  Jul 10 16:26:19.129879 Local   ge-0/0/2.0                         16
  Jul 10 16:26:25.972747 Remote  10.255.0.3                          17
Source: 10.255.0.3, Rank: 1, Status: Active
MAC label: 299776
Mobility sequence number: 17 (minimum origin address 10.255.0.3)
Timestamp: Jul 10 16:26:25 (0x59640d21)
State: <Remote-To-Local-Adv-Done>
IP address: 10.0.0.3
  L3 route: 10.0.0.3/32, L3 context: DELTA (irb.0)

```

show evpn database extensive (Duplicate MAC Address)

```

user@PE1> show evpn database instance ALPHA mac-address 00:00:00:00:00:012extensive
Instance: ALPHA

VLAN ID: 100, MAC address: 00:00:00:00:00:02
State: 0x1 <Duplicate-Detected>
Mobility history
  Mobility event time      Type      Source                               Seq num
  Aug 03 17:22:28.585619 Local   ge-0/0/2.0                         31
  Aug 03 17:22:30.307198 Remote  10.255.0.3                          32
  Aug 03 17:22:37.611786 Local   ge-0/0/2.0                         33
  Aug 03 17:22:39.289357 Remote  10.255.0.3                          34
  Aug 03 17:22:45.609449 Local   ge-0/0/2.0                         35
Source: ge-0/0/2.0, Rank: 1, Status: Active
Mobility sequence number: 35 (minimum origin address 10.255.0.2)

```

```

Timestamp: Aug 03 17:22:44 (0x5983be54)
State: <Local-MAC-Only Local-To-Remote-Adv-Allowed>
MAC advertisement route status: Not created (duplicate MAC suppression)
IP address: 10.0.0.2
Source: 10.255.0.3, Rank: 2, Status: Inactive
MAC label: 300176
Mobility sequence number: 34 (minimum origin address 10.255.0.3)
Timestamp: Aug 03 17:22:39 (0x5983be4f)
State: <>
MAC advertisement route status: Not created (inactive source)
IP address: 10.0.0.3

```

Release Information

Command introduced in Junos OS Release 14.2.

RELATED DOCUMENTATION

[Example: Configuring an EVPN with IRB Solution on EX9200 Switches](#) | 995

show evpn flood

IN THIS SECTION

- [Syntax](#) | 1826
- [Description](#) | 1826
- [Options](#) | 1826
- [Required Privilege Level](#) | 1826
- [Release Information](#) | 1826

Syntax

```
show evpn flood
event-queue
<instance instance-name>
<logical-system logical-system-name>
<route route-name>
```

Description

Show Ethernet VPN (EVPN) flooding information.

Options

<code>none</code>	Display brief information about EVPN flooding.
<code>brief detail extensive summary</code>	(Optional) Display the specified level of output.
<code>event-queue</code>	(Optional) Display the queue of pending EVPN flood events.
<code>instance <i>instance-name</i></code>	(Optional) Display flooding information for the specified routing instance.
<code>logical-system <<i>logical-system-name</i>></code>	(Optional) Display flooding information for the specified logical system or all logical systems.
<code>route <i>route-name</i></code>	(Optional) Display flooding information for the specified route.

Required Privilege Level

view

Release Information

Command introduced in Junos OS Release 14.2.

RELATED DOCUMENTATION

| [Example: Configuring an EVPN with IRB Solution on EX9200 Switches](#) | 995

show evpn igmp-snooping database

IN THIS SECTION

- [Syntax | 1827](#)
- [Syntax for mac-vrf routing | 1827](#)
- [Description | 1828](#)
- [Options | 1828](#)
- [Required Privilege Level | 1828](#)
- [Output Fields | 1828](#)
- [Sample Output | 1829](#)
- [Release Information | 1830](#)

Syntax

```
show evpn igmp-snooping database
<extensive>
<esi id>
<group group-address>
<instance id>
<interface name>
<l2-domain-id id>
<logical-system name | all>
```

Syntax for mac-vrf routing

```
show mac-vrf routing igmp-snooping database
<extensive>
<esi id>
<group group-address>
<instance id>
<interface name>
<l2-domain-id id>
```

Description

Display IGMP snooping information in an EVPN network.

Options

none	Display detailed information.
extensive	(Optional) Display more extensive output.
esi <i>esi-id</i>	(Optional) Display output only for a specified Ethernet segment ID (ESI).
group <i>group-address</i>	(Optional) Display output only for the specified multicast group address range.
instance <i>instance-name</i>	(Optional) Display information for the specified EVPN instance (EVI).
interface <i>name</i>	(Optional) Display information about the specified interface.
l2domain <i>l2-domain-id</i>	(Optional) Display output for the specified bridging domain, VLAN, or virtual network identifier (VNI).

Required Privilege Level

view

Output Fields

[Table 70 on page 1828](#) lists the output fields for the `show evpn igmp-snooping database` command. Output fields are listed in the approximate order in which they appear.

Table 70: show evpn igmp-snooping database Output Fields

Field Name	Field Description	Level of Output
Instance	Name of the EVPN routing instance for which information is displayed.	All levels
VN Identifier	Tunnel network identifier.	All levels
Group IP	Multicast IP address of the multicast group.	All levels

Table 70: show evpn igmp-snooping database Output Fields (Continued)

Field Name	Field Description	Level of Output
Source IP	IP address of the multicast data source for the group.	All levels
Access OIF Count	Number of outgoing interface (OIF) list entries on the access side.	All levels
Access OIF Count (more information)	More OIF Entry Information: <ul style="list-style-type: none"> Interface—Interface name ESI—Associated EVPN Ethernet Segment ID for the interface Local—Local receiver. Remote—Remote receiver. 	extensive
Remote OIF	Information about outgoing remote interfaces: <ul style="list-style-type: none"> Count—Number of OIF list entries Core NH—Next hop ID corresponding to the next hop interface. OIF Entry Information: <ul style="list-style-type: none"> Interface—Interface name Nbr—Neighbor ID. 	extensive

Sample Output

show evpn igmp-snooping database extensive (default switch instance)

```
user@device> show evpn igmp-snooping database extensive
```

```
Instance: default-switch
```

```
VLAN ID: 7000
```

```
Group IP: 233.252.0.2
```

```
Access OIF Count: 1
```

Interface	ESI	Local	Remote
xe-0/0/1.0	00:01:02:03:04:05:06:07:08:09 0		1

```
Remote OIF Count: 4, Core NH: 1048676
```


Interface	Nbr
vtep.32768	9.255.255.2
vtep.32768	9.255.255.3
vtep.32768	9.255.255.4
vtep.32768	9.255.255.5

show evpn igmp-snooping database instance instance-name extensive

```

user@device> show evpn igmp-snooping database instance evpn-B extensive
Instance: evpn-B
  VN Identifier: 477
    Group IP: 233.252.0.1, Source IP: 0.0.0.0
      Access OIF Count: 1
        Interface      ESI              Local  Remote
        ae477.477      00:22:44:66:88:00:22:44:66:99 1      0
    Group IP: 233.252.0.2, Source IP: 0.0.0.0
      Access OIF Count: 1
        Interface      ESI              Local  Remote
        ae477.477      00:22:44:66:88:00:22:44:66:99 1      0
    Group IP: 233.252.0.3, Source IP: 0.0.0.0
      Access OIF Count: 1
        Interface      ESI              Local  Remote
        ae477.477      00:22:44:66:88:00:22:44:66:99 1      0
    Group IP: 233.252.0.4, Source IP: 0.0.0.0
      Access OIF Count: 1
        Interface      ESI              Local  Remote
        ae477.477      00:22:44:66:88:00:22:44:66:99 1      0

```

show evpn igmp-snooping database instance l2-domain-id domain-ID

```

user@device> show evpn igmp-snooping database l2-domain-id 100201
Instance: default-switch
  VN Identifier: 100201
    Group IP: 233.252.0.1, Source IP: 0.0.0.0, Access OIF Count: 2
    Group IP: 233.252.0.2, Source IP: 0.0.0.0, Access OIF Count: 2

```

Release Information

Introduced in Junos OS Release 17.2R1 on QFX10000 switches.

Introduced in Junos OS Release 18.2R1 on MX Series routers and EX9200 switches.

Introduced in Junos OS Release 19.4R2 on ACX Series routers.

RELATED DOCUMENTATION

[Configure Multicast Forwarding with IGMP Snooping in an EVPN-MPLS Environment | 1028](#)

[Example: Preserving Bandwidth with IGMP Snooping in an EVPN-VXLAN Environment | 694](#)

[Overview of Multicast Forwarding with IGMP or MLD Snooping in an EVPN-MPLS Environment | 1013](#)

[Overview of Multicast Forwarding with IGMP Snooping or MLD Snooping in an EVPN-VXLAN Environment | 675](#)

show evpn igmp-snooping proxy

IN THIS SECTION

- [Syntax | 1831](#)
- [Syntax for mac-vrf routing | 1832](#)
- [Description | 1832](#)
- [Options | 1832](#)
- [Required Privilege Level | 1832](#)
- [Output Fields | 1832](#)
- [Sample Output | 1833](#)
- [Release Information | 1833](#)

Syntax

```
show evpn igmp-snooping proxy
<extensive>
<group group-address>
<instance instance-name>
```

```
<l2-domain-id id>
<logical-system name | all>
```

Syntax for mac-vrf routing

```
show mac-vrf routing igmp-snooping proxy
<extensive>
<group group-address>
<instance instance-name>
<l2-domain-id id>
```

Description

Display EVPN IGMP snooping proxy information.

Options

- extensive** (Optional) Display more extensive output.
- group *group-address*** (Optional) Display output only for the specified multicast group address range.
- instance *instance-name*** (Optional) Display information for the specified EVPN instance.
- l2-domain-id *id*** (Optional) Display output for the specified bridging domain, VLAN, or virtual network identifier (VNI).

Required Privilege Level

view

Output Fields

[Table 71 on page 1833](#) lists the output fields for the show evpn instance command. Output fields are listed in the approximate order in which they appear.

Table 71: show evpn igmp snooping output fields

Field Name	Field Description
VN Identifier	VXLAN network identifier.
Group	Multicast group IP address.
Source	Multicast source IP address.
Local	Indicates whether the access interface has a listener. A value of 1 indicates there is an interested listener.
Remote	Number of remote interested listeners.
Corenh	Core next hop identifier.
Flood	Indicates whether flooding is used for this multicast group. A value of 0 indicates that selective multicast forwarding is used in the traffic flow. A value of 1 indicates that inclusive multicast forwarding (flooding) is being used in the traffic flow.

Sample Output

show evpn igmp-snooping proxy extensive

```

user@pe1> show evpn igmp-snooping proxy extensive
Instance: default-switch
  VN Identifier: 100
    Group      Source   Local  Remote  Corenh  Flood
    ---      -
    235.1.1.2  0.0.0.0   0      1       131083   0
    235.1.1.4  0.0.0.0   0      1       131080   0

```

Release Information

Introduced in Junos OS Release 18.3R1 on QFX Series Switches.

RELATED DOCUMENTATION

[Overview of Multicast Forwarding with IGMP or MLD Snooping in an EVPN-MPLS Environment | 1013](#)

[Overview of Multicast Forwarding with IGMP Snooping or MLD Snooping in an EVPN-VXLAN Environment | 675](#)

[Overview of Selective Multicast Forwarding | 747](#)

show evpn instance

IN THIS SECTION

- [Syntax | 1834](#)
- [Syntax for mac-vrf routing | 1835](#)
- [Description | 1835](#)
- [Options | 1835](#)
- [Required Privilege Level | 1836](#)
- [Output Fields | 1836](#)
- [Sample Output | 1841](#)
- [Release Information | 1851](#)

Syntax

```
show evpn instance
<brief | extensive>
<backup-forwarder>
<dci>
<designated-forwarder>
<esi esi>
<esi-info esi>
<instance-name>
<neighbor neighbor-address>
<neighbor-info neighbor-address>
```

Syntax for mac-vrf routing

```

show mac-vrf routing instance
<brief | extensive>
<backup-forwarder>
<dci>
<designated-forwarder>
<esi esi>
<esi-info esi>
<instance-name>
<neighbor neighbor-address>
<neighbor-info neighbor-address>

```

Description

Show Ethernet VPN (EVPN) routing instance information.

When up MEP is configured for an EVPN instance (EVI), the **show evpn instance** displays a default interface without any configuration with a naming convention as .local..*number* (for example, .local..8).

Options

none	Display brief information about the EVPN routing instance.
brief extensive	(Optional) Display the specified level of output.
backup-forwarder	(Optional) Display IP addresses of all the backup designated forwarders for the Ethernet segment.
dci	(Optional) Display data center interconnect information.
designated-forwarder	(Optional) Display the IP address of the designated forwarder for the Ethernet segment.
esi esi	(Optional) Display brief information about the routing instance associated with the specified Ethernet segment identifier (ESI) value, including information on the route distinguisher, bridge domain, IRB, and other information associated with the ESI.
esi-info esi	(Optional) Display only the ESI information about the routing instance associated with the specified ESI value.

<i>instance-name</i>	(Optional) Display information about the specified routing instance.
<i>neighbor neighbor-address</i>	(Optional) Display the IP address of the EVPN neighbor along with information on the connected bridge domain, route distinguisher, and IRB.
<i>neighbor-info neighbor-address</i>	(Optional) Display only the IP address of the EVPN neighbor for a routing instance.

Required Privilege Level

view

Output Fields

Table 72 on page 1836 lists the output fields for the `show evpn instance` command. Output fields are listed in the approximate order in which they appear.

Table 72: show evpn instance Output Fields

Field Name	Field Description	Level of Output
Instance	Names of the routing instances.	All levels
Intfs	Total number of interfaces participating in each routing instance, and number of interfaces that are up.	brief
IRB intfs	Statistics on the number of integrated routing and bridging (IRB) interfaces for each routing instance: <ul style="list-style-type: none"> Total—Total number of IRB interfaces. Up—Number of active IRB interfaces. Nbrs—Number of neighbor IRB interfaces. 	brief
MH ESIs	Number of Ethernet segments per routing instance that connect to a multihomed customer site.	brief
MAC addresses	Number of local and remote MAC addresses for each routing instance.	brief

Table 72: show evpn instance Output Fields (Continued)

Field Name	Field Description	Level of Output
Route Distinguisher	Unique route distinguisher associated with this routing instance.	none
VLAN ID	VLAN identifier.	none
Label allocation mode	Label allocation policy for the routing instance.	none
Encapsulation type	Encapsulation type for VXLAN EVPN instances (EVIs).	extensive
Per-instance MAC route label	Label of MAC route for each routing instance.	none
Duplicate MAC detection threshold	Number of MAC mobility events detected for a given MAC address before it is identified as a duplicate MAC address.	extensive
Duplicate MAC detection window	The time interval used in detecting a duplicate MAC address.	extensive
Duplicate MAC auto-recovery time	Length of time a device suppresses a duplicate MAC address. At the end of this duration, MAC address updates will resume.	extensive
DF Election preference	<p>Preference value used for the designated forwarder (DF) election:</p> <ul style="list-style-type: none"> • Highest preference—Default DF election preference. • Lowest preference—Based on the configuration of the designated-forwarder-preference-least statement at the [edit routing-instance <i>routing-instance-name</i> protocols evpn] hierarchy level. 	extensive
Per-instance multicast route label	Label of the multicast route for each routing instance.	none

Table 72: show evpn instance Output Fields (Continued)

Field Name	Field Description	Level of Output
Total MAC addresses	Total number of local and remote MAC addresses received for each routing instance.	extensive
Default gateway MAC addresses	Number of local and remote MAC addresses serving as a default gateway in this routing instance.	extensive
Number of local interfaces	Number of local interfaces belonging to this routing instance.	extensive
Number of local interfaces up	Number of active local interfaces belonging to this routing instance.	extensive
Interface name	Name of interfaces that belong to this routing instance.	extensive
ESI	Ethernet segment identifier (ESI) value of the interfaces belonging to this routing instance.	extensive
DF Election Algorithm	DF election type: <ul style="list-style-type: none"> • MOD based—Default DF election algorithm based on the modulo operation. • Preference based—DF election based on the configured preference values for an ESI. 	extensive
ESI granularity	Type: Per ESI.	extensive
LACP OOS on NDF	Enabled or Disabled	extensive
Preference	Preference value for EVPN Multihoming DF election.	extensive

Table 72: show evpn instance Output Fields (Continued)

Field Name	Field Description	Level of Output
Mode	<p>Mode of operation for each routing instance:</p> <ul style="list-style-type: none"> • single-homed—Default mode and does not require Ethernet segment values to be configured. • single-active—EVPN active-standby multihoming mode of operation. 	extensive
SH label	Split horizon label used for the active-standby multihoming mode of operation.	extensive
Number of IRB interfaces	Number of IRB interfaces that belong to this routing instance.	extensive
Number of protect interfaces	Number of protect interfaces that belong to this routing instance.	extensive
Interface name	Name of the primary interface.	extensive
Protect Interface name	Name of the protect interface.	extensive
Status	Protected status of the primary interface. The status is either Protect-inactive or Protect-active.	extensive
L3 context	Names of routing instances that have the Layer 3 routes installed for an EVPN IRB interface, typically a virtual routing and forwarding (VRF) routing instance.	extensive
Number of bridge domains	Number of bridge domain for each EVPN instance.	extensive
MAC label	Locally allocated label advertised per bridge domain for forwarding unicast traffic.	extensive

Table 72: show evpn instance Output Fields *(Continued)*

Field Name	Field Description	Level of Output
Number of neighbors	Number of neighbors connected to this routing instance and their IP addresses.	extensive
MAC address advertisement	Number of MAC address advertisements received from the neighbor.	extensive
MAC+IP address advertisement	Number of MAC and IP address advertisements received from the neighbor.	extensive
Inclusive multicast	Number of inclusive multicast routes received from the neighbor.	extensive
Ethernet auto-discovery	Number of autodiscovery routes per Ethernet segment received from the neighbor.	extensive
Number of ethernet segments	Total number of Ethernet segments for the routing instance.	extensive
Designated forwarder	IP address of the designated forwarder (DF) for the Ethernet segment.	extensive
Backup forwarder	<p>IP address of all the backup designated forwarders or routers that are not designated forwarders for the Ethernet segment.</p> <p>NOTE: Immediately after an EVPN interface Ethernet segment identifier value is changed and the new configuration is committed, the Designated forwarder information changes to DF not elected yet and the backup forwarder information is not displayed until after the election is complete.</p>	extensive
SMET Forwarding	Identifies whether a device is using selective mulicast forwarding.	extensive
Nexthop Limit	The number of SMET next hop that is supported.	extensive

Table 72: show evpn instance Output Fields (Continued)

Field Name	Field Description	Level of Output
Nexthop Usage	The number of SMET next hop currently in use.	extensive

Sample Output

show evpn instance brief

```
user@host> show evpn instance brief
```

Instance	Intfs		IRB intfs		Nbrs	MH	MAC addresses	
	Total	Up	Total	Up		ESIs	Local	Remote
ALPHA	2	2	1	1	2	1	3	4
BETA	2	2	1	1	2	1	2	4
__default_evpn__	0	0	0	0	1	0	0	0

show evpn instance

```
user@host> show evpn instance
Instance: black
Route Distinguisher: 101:101
VLAN ID: 100
Label allocation mode: Per-instance
Per-instance MAC route label: 299776
Per-instance multicast route label: 299792
Number of local interfaces: 1
Number of local interfaces up: 1
  Interface name   Static MACs   ESI
  cbp-0.0          0            0
Number of neighbors: 1
192.0.2.1
  Received routes
    MAC address advertisement:      1
    Ethernet auto-discovery:        0
    Inclusive multicast:             1
```

show evpn instance extensive (In Junos OS Release 16.1 and earlier)

```

user@host> show evpn instance extensive
Instance: ALPHA
  Route Distinguisher: 10.255.0.1:100
  Encapsulation type: VXLAN
  Per-instance MAC route label: 300144
  Per-instance multicast route label: 300160
  MAC database status
    Local Remote
  Total MAC addresses:          3      4
  Default gateway MAC addresses: 1      2
  Number of local interfaces: 2 (2 up)
    Interface name  ESI                               Mode          SH label
  ae0.0            00:11:22:33:44:55:66:77:88:99 single-active
  ge-0/0/2.0       00:00:00:00:00:00:00:00:00:00 single-homed
  Number of IRB interfaces: 1 (1 up)
    Interface name  L3 context
  irb.0            DELTA
  Number of neighbors: 2
    10.255.0.2
      Received routes
        MAC address advertisement:          2
        MAC+IP address advertisement:       3
        Inclusive multicast:                1
        Ethernet auto-discovery:            1
    10.255.0.3
      Received routes
        MAC address advertisement:          2
        MAC+IP address advertisement:       2
        Inclusive multicast:                1
        Ethernet auto-discovery:            0
  Number of ethernet segments: 1
    ESI: 00:11:22:33:44:55:66:77:88:99
    Designated forwarder: 10.255.0.1
    Backup forwarder: 10.255.0.2

Instance: BETA
  Route Distinguisher: 10.255.0.1:300
  Encapsulation type: VXLAN
  VLAN ID: 300
  Per-instance MAC route label: 300176
  Per-instance multicast route label: 300192

```

```

MAC database status          Local Remote
Total MAC addresses:        3      4
Default gateway MAC addresses: 1      2
Number of local interfaces: 2 (2 up)
Interface name  ESI                      Mode      SH label
ae1.0          00:00:00:00:00:00:00:00:00 single-homed
ge-0/0/4.0     00:22:44:66:88:00:22:44:66:88 single-active
Number of IRB interfaces: 1 (1 up)
Interface name  L3 context
irb.1          DELTA
Number of neighbors: 2
10.255.0.2
Received routes
MAC address advertisement:      2
MAC+IP address advertisement:   3
Inclusive multicast:            1
Ethernet auto-discovery:        1
10.255.0.3
Received routes
MAC address advertisement:      2
MAC+IP address advertisement:   2
Inclusive multicast:            1
Ethernet auto-discovery:        0
Number of ethernet segments: 1
ESI: 00:22:44:66:88:00:22:44:66:88
Designated forwarder: 10.255.0.1
Backup forwarder: 10.255.0.2

Instance: __default_evpn__
Route Distinguisher: 10.255.0.1:0
Encapsulation type: VXLAN
VLAN ID: 0
Per-instance MAC route label: 300208
Per-instance multicast route label: 300224
MAC database status          Local Remote
Total MAC addresses:        0      0
Default gateway MAC addresses: 0      0
Number of local interfaces: 0 (0 up)
Number of IRB interfaces: 0 (0 up)
Number of neighbors: 1
10.255.0.2
Received routes
Ethernet auto-discovery:      0

```

```

Ethernet Segment:                2
Number of ethernet segments: 0

```

show evpn instance extensive (In Junos OS Release 16.2 and later)

```
user@host> show evpn instance extensive
```

```
user@host> show evpn instance extensive
```

Instance: ALPHA

Route Distinguisher: 10.255.0.1:100

Encapsulation type: VXLAN

Per-instance MAC route label: 300144

Per-instance multicast route label: 300160

MAC database status	Local	Remote
---------------------	-------	--------

Total MAC addresses:	3	4
----------------------	---	---

Default gateway MAC addresses:	1	2
--------------------------------	---	---

Number of local interfaces: 2 (2 up)

Interface name	ESI	Mode	SH label
ae0.0	00:11:22:33:44:55:66:77:88:99	single-active	
ge-0/0/2.0	00:00:00:00:00:00:00:00:00:00	single-homed	

Number of IRB interfaces: 1 (1 up)

Interface name	L3 context
----------------	------------

irb.0	DELTA
-------	-------

Number of neighbors: 2

Address	MAC	MAC+IP	AD	IM	ES
10.255.0.2	2	3	1	1	0
10.255.0.3	2	2	0	1	0

Number of ethernet segments: 1

ESI: 00:11:22:33:44:55:66:77:88:99

Designated forwarder: 10.255.0.1

Backup forwarder: 10.255.0.2

Instance: BETA

Route Distinguisher: 10.255.0.1:300

Encapsulation type: VXLAN

VLAN ID: 300

Per-instance MAC route label: 300176

Per-instance multicast route label: 300192

MAC database status	Local	Remote
---------------------	-------	--------

Total MAC addresses:	3	4
----------------------	---	---

```

    Default gateway MAC addresses:      1      2
Number of local interfaces: 2 (2 up)
  Interface name  ESI                               Mode      SH label
  ae1.0          00:00:00:00:00:00:00:00:00:00 single-homed
  ge-0/0/4.0     00:22:44:66:88:00:22:44:66:88 single-active
Number of IRB interfaces: 1 (1 up)
  Interface name  L3 context
  irb.1          DELTA
Number of neighbors: 2
  Address          MAC      MAC+IP      AD      IM      ES
  10.255.0.2       2        3          1       1       0
  10.255.0.3       2        2          0       1       0
Number of ethernet segments: 1
  ESI: 00:22:44:66:88:00:22:44:66:88
  Designated forwarder: 10.255.0.1
  Backup forwarder: 10.255.0.2

Instance: __default_evpn__
  Route Distinguisher: 10.255.0.1:0
  Encapsulation type: VXLAN
  VLAN ID: 0
  Per-instance MAC route label: 300208
  Per-instance multicast route label: 300224
  MAC database status      Local  Remote
  Total MAC addresses:      0      0
  Default gateway MAC addresses: 0      0
Number of local interfaces: 0 (0 up)
Number of IRB interfaces: 0 (0 up)
Number of neighbors: 1
  Address          MAC      MAC+IP      AD      IM      ES
  10.255.0.2       0        0          0       0       2
Number of ethernet segments: 0

```

show evpn instance extensive (Preference-based DF Election)

```

user@host> show evpn instance EVPN_1 extensive
Instance: EVPN_1
  Route Distinguisher: 1:101
  Per-instance MAC route label: 299792
+ DF Election preference: Lowest preference
  MAC database status      Local  Remote

```



```

MAC advertisements:                7      15
MAC+IP advertisements:             7      12
Default gateway MAC advertisements: 4      8
Number of local interfaces: 2 (2 up)
  Interface name  ESI                                Mode      Status
  ae101.0         00:11:11:11:11:11:11:11:01  all-active  Up
  ae102.0         00:11:11:11:11:11:11:11:02  all-active  Up
Number of IRB interfaces: 4 (4 up)
  Interface name  VLAN   VNI    Status  L3 context
  irb.101         101           Up      master
  irb.102         102           Up      master
  irb.103         103           Up      master
  irb.104         104           Up      master
Number of bridge domains: 4
  VLAN  Domain ID  Intfs / up  IRB intf  Mode      MAC sync  IM route label
  101           1    1    irb.101  Extended  Enabled   299776
  102           1    1    irb.102  Extended  Enabled   299776
  103           1    1    irb.103  Extended  Enabled   299776
  104           1    1    irb.104  Extended  Enabled   299776
Number of neighbors: 2
  100.0.0.2
    Received routes
      MAC address advertisement:      7
      MAC+IP address advertisement:   4
      Inclusive multicast:            4
      Ethernet auto-discovery:        4
  100.0.0.4
    Received routes
      MAC address advertisement:      8
      MAC+IP address advertisement:   8
      Inclusive multicast:            4
      Ethernet auto-discovery:        0
Number of ethernet segments: 2
  ESI: 00:11:11:11:11:11:11:11:11
  Status: Resolved by IFL ae0.110
  Local interface: ae0.110, Status: Up/Forwarding
  Number of remote PEs connected: 2
    Remote PE      MAC label  Aliasing label  Mode
    100.100.100.3   0          0              single-active
    100.100.100.2   0          0              single-active
  DF Election Algorithm: Preference based
  Designated forwarder: 100.100.100.3, Preference: 200
  Backup forwarder: 100.100.100.1, Preference: 800

```

```

Backup forwarder: 100.100.100.2, Preference: 400
ESI: 00:11:11:11:11:11:11:11:02
Status: Resolved by IFL ae102.0
Local interface: ae102.0, Status: Up/Forwarding
Number of remote PEs connected: 1
  Remote PE      MAC label  Aliasing label  Mode
  100.0.0.2      299792      299792          all-active
DF Election Algorithm: MOD based
Designated forwarder: 100.0.0.2
Backup forwarder: 100.0.0.1
Last designated forwarder update: Feb 21 22:23:32
Advertised split horizon label: 300800

```

show evpn instance extensive (Duplicate MAC Address)

```

user@host> show evpn instance ALPHA extensive
Instance: ALPHA
Route Distinguisher: 10.255.0.2:100
Per-instance MAC route label: 304192
Duplicate MAC detection threshold: 5
Duplicate MAC detection window: 180
Duplicate MAC auto-recovery time: 10
...

```

show evpn instance extensive (Protected Interface)

```

user@host> show evpn instance blue extensive
Instance: blue
Route Distinguisher: 10.255.255.1:100
Per-instance MAC route label: 299776
MAC database status
  Local  Remote
MAC advertisements:          0      0
MAC+IP advertisements:      0      0
Default gateway MAC advertisements: 0      0
Number of local interfaces: 5 (5 up)
  Interface name  ESI                               Mode          Status  AC-Role
  ae0.0           00:11:22:33:44:55:66:77:88:99    all-active    Up      Root
  ge-0/0/3.0      00:00:00:00:00:00:00:00:00:00    single-homed  Up      Root
  ge-0/0/4.0      00:11:11:11:44:55:66:77:88:99    all-active    Up      Root

```

```

    ge-0/0/4.1      00:22:22:22:44:55:66:77:88:99 all-active    Up      Root
    ge-0/0/4.50     00:00:00:00:00:00:00:00:00 single-homed  Up      Root
Number of IRB interfaces: 1 (0 up)
  Interface name  VLAN   VNI    Status  L3 context
  irb.1           25                Down    vrf
Number of protect interfaces: 1
  Interface name  protect-interface  active-interface
  ge-0/0/3.1      ge-0/0/4.50        ge-0/0/3.1

```

show evpn instance extensive

The following output shows a partial output snippet with information on a bridge domain. The output shows the MAC labels that have been allocated for each bridge domain.

```

user@host> show evpn instance extensive
Instance: evpn-A
.
.
.
Number of bridge domains: 1
  VLAN  Domain-ID Intfs/up  IRB-intf  Mode          MAC-sync IM-label  MAC-Label  v4-SG-sync
IM-core-NH v6-SG-sync IM-core-NH Trans-ID
    601                1 1    irb.0          Extended  Enabled    300880
300656    Disabled                Disabled
    602                1 1          Extended  Enabled    300896
300672    Disabled                Disabled
Number of neighbors: 3
  Address          MAC    MAC+IP    AD    IM    ES Leaf-label Remote-DCI-Peer
  81.3.3.3          0      0         2     1     0
  81.4.4.4          0      0         2     1     0
  81.5.5.5          1      1         2     1     0
Number of ethernet segments: 1
ESI: 00:01:02:03:04:05:06:07:08:09
Status: Resolved by NH 47084
Number of remote PEs connected: 3
  Remote-PE      MAC-label  Aliasing-label  Mode
  81.4.4.4        0          18              all-active
  81.3.3.3        0          18              all-active
  81.5.5.5        299920     299920          all-active
SMET Forwarding: Disabled

```

show evpn instance esi-info

```

user@host> show evpn instance esi-info esi 00:11:22:33:44:55:66:77:88:99
Instance: ALPHA
  Number of ethernet segments: 1
    ESI: 00:11:22:33:44:55:66:77:88:99
      Status: Resolved by IFL ae0.0
      Local interface: ae0.0, Status: Up/Forwarding
      Number of remote PEs connected: 1
        Remote PE      MAC label  Aliasing label  Mode
        10.255.0.2      0          299856          all-active
      DF Election Algorithm: MOD based
      Designated forwarder: 10.255.0.1
      Backup forwarder: 10.255.0.2
      Last designated forwarder update: Feb 04 11:52:47
      Advertised MAC label: 299856
      Advertised aliasing label: 299856
      Advertised split horizon label: 299872

```

show evpn instance esi backup-forwarder (Instance Name with Ethernet Segment Identifier)

```

user@host> show evpn instance ALPHA esi 00:11:22:33:44:55:66:77:88:99
backup-forwarder
Instance: ALPHA
  Number of ethernet segments: 1
    ESI: 00:11:22:33:44:55:66:77:88:99
      Backup forwarder: 10.255.0.2

```

show evpn instance esi designated-forwarder (Instance Name with Ethernet Segment Identifier)

```

user@host> show evpn instance ALPHA esi 00:11:22:33:44:55:66:77:88:99
designated-forwarder
Instance: ALPHA
  Number of ethernet segments: 1
    ESI: 00:11:22:33:44:55:66:77:88:99
      Designated forwarder: 10.255.0.1

```

show evpn instance (CFM up)

The following shows the partial output listing the interface for show evpn instance command.

```
user@host > show evpn instance evpn_1
Number of local interfaces: 3 (3 up)
  Interface name  ESI                               Mode      Status   AC-Role
  .local..8      00:00:00:00:00:00:00:00:00:00    single-homed  Up       Root
  xe-0/0/11:0.0  00:00:00:00:00:00:00:00:00:00    single-homed  Up       Root
```

show evpn instance (SMET)

```
user@host > show evpn instance evpn_1
Instance: default-switch
Route Distinguisher: 10.255.0.3:100
Encapsulation type: VXLAN
Duplicate MAC detection threshold: 5
Duplicate MAC detection window: 180
MAC database status
  Local  Remote
  MAC advertisements:      4      1
  MAC+IP advertisements:   4      1
  Default gateway MAC advertisements: 2      0
Number of local interfaces: 3 (3 up)
  Interface name  ESI                               Mode      Status   AC-Role
  .local..5      00:00:00:00:00:00:00:00:00:00    single-homed  Up       Root
  xe-0/0/1.0     00:00:00:00:00:00:00:00:00:00    single-homed  Up       Root
  xe-0/0/2.0     00:00:00:00:00:00:00:00:00:00    single-homed  Up       Root
Number of IRB interfaces: 2 (2 up)
  Interface name  VLAN  VNI  Status  L3 context
  irb.0          100   Up   DELTA
  irb.1          200   Up   DELTA
Number of protect interfaces: 0
Number of bridge domains: 2
  VLAN  Domain ID  Intfs / up  IRB intf  Mode      MAC sync  IM route label  IPv4 SG sync
IPv4 IM core nexthop  IPv6 SG sync  IPv6 IM core nexthop
  100   100      1 1  irb.0  Extended  Enabled  100
Enabled 131074 Disabled
  200   200      1 1  irb.1  Extended  Enabled  200
Enabled 131077 Disabled
Number of neighbors: 3
```

Address	MAC	MAC+IP	AD	IM	ES Leaf-label
10.255.0.2	0	0	0	2	0
10.255.0.4	1	1	0	2	0
10.255.0.5	0	0	0	2	0

Number of ethernet segments: 2

ESI: 05:00:00:00:64:00:00:00:64:00

Local interface: irb.0, Status: Up/Forwarding

ESI: 05:00:00:00:64:00:00:00:c8:00

Local interface: irb.1, Status: Up/Forwarding

Router-ID: 10.255.0.3

SMET Forwarding: Enabled: Nexthop Limit: 10000 Nexthop Usage: 7

show evpn instance (Per ESI)

```
user@host > show evpn instance ALPHA esi 00:11:11:11:11:11:11:11:11
```

Instance: ALPHA

Number of ethernet segments: 1

ESI: 00:11:11:11:11:11:11:11:11

Status: Unresolved

Local interface: ae0.210, Status: Down

Number of remote PEs connected: 1

Remote PE	MAC label	Aliasing label	Mode
10.10.10.2	0	300256	single-active

DF Election Algorithm: MOD based

ESI granularity: Per ESI

LACP OOS on NDF: Enabled

Designated forwarder: 10.255.0.1

Last designated forwarder update: Apr 16 17:08:33

Advertised MAC label: 301248

Advertised aliasing label: 301248

Advertised split horizon label: 300032

Release Information

Command introduced in Junos OS Release 14.1.

show evpn ip-prefix-database

IN THIS SECTION

- [Syntax | 1852](#)
- [Description | 1852](#)
- [Options | 1852](#)
- [Required Privilege Level | 1853](#)
- [Output Fields | 1853](#)
- [Sample Output | 1855](#)
- [Release Information | 1862](#)

Syntax

```
show evpn ip-prefix-database
<extensive>
<direction (imported | exported)>
<esi number>
<ethernet-tag number>
<family (inet | inet6)>
<gateway ip-address>
<l3-context routing-instance-name>
<logical-system logical-system-name>
<nexthop ip-address>
<prefix ip-prefix>
```

Description

Display Ethernet VPN (EVPN) database information for imported and exported IP prefixes.

Options

none	Display standard information for all EVPN imported and exported IP prefixes.
-------------	--

extensive	Display the specified level of output.
direction (imported exported)	(Optional) Display the specified subset of IP prefixes.
esi <i>number</i>	(Optional) Display the IP prefix associated with the specified Ethernet segment identifier.
ethernet-tag <i>number</i>	(Optional) Display the IP prefix associated with the specified Ethernet tag.
family (inet inet6)	(Optional) Display IP prefixes for the specified protocol family.
gateway <i>ip-address</i>	(Optional) Display the IP Prefix associated with the overlay gateway IP address.
l3-context <i>routing-instance-name</i>	(Optional) Display EVPN IP prefix information for the specified Layer 3 virtual routing and forwarding (VRF) instance.
logical-system <<i>logical-system-name</i>>	(Optional) Display EVPN IP prefix information for the specified logical system or all logical systems.
nexthop <i>ip-address</i>	(Optional) Display EVPN IP prefix information for the specified underlay next-hop IP address.
prefix <i>ip-prefix</i>	(Optional) Display EVPN database information for the specified IP prefix.

Required Privilege Level

view

Output Fields

[Table 73 on page 1853](#) lists the output fields for the `show evpn ip-prefix-database` command. Output fields are listed in the approximate order in which they appear.

Table 73: show evpn ip-prefix-database Output Fields

Field Name	Field Description	Level of Output
L3 context	Name of virtual routing and forwarding (VRF) instance.	All levels

Table 73: show evpn ip-prefix-database Output Fields (Continued)

Field Name	Field Description	Level of Output
EVPN Exported Prefixes Prefix	List of exported IP prefixes.	All levels
EVPN route status	For exported prefixes only, status of route: Created.	level-of-output none
EVPN imported Prefixes	List of imported EVPN IP prefixes.	All levels
Etag	Ethernet tag	level-of-output none
Route distinguisher	IP address identifier for the IP prefix route.	All levels
VNI	Virtual network identifier for the Layer 3 virtual and routing forwarding (VRF) for the customer or tenant domain.	All levels
Router MAC	MAC address associated with the IP prefix.	All levels
Nexthop/Overlay GW/ESI	For imported IP prefixes, next-hop IP address.	level-of-output none
Change flags	Trace flags.	level-of-output extensive
Advertisement mode	For exported IP prefixes, type of next-hop address.	level-of-output extensive
Encapsulation	For exported IP prefixes, type of encapsulation	level-of-output extensive

Table 73: show evpn ip-prefix-database Output Fields (Continued)

Field Name	Field Description	Level of Output
Remote Advertisements	For imported IP prefixes, route distinguisher identifier, MAC address, virtual network identifier, and BGP next-hop address to remote destination.	level-of-output extensive

Sample Output

show evpn ip-prefix-database

```
user@host > show evpn ip-prefix-database
```

```
L3 context: VRF-100
```

IPv4->EVPN Exported Prefixes

Prefix	EVPN route status
100.1.0.0/22	Created
100.1.4.0/22	Created
100.1.8.0/22	Created
100.1.12.0/22	Created
100.1.16.0/22	Created

IPv6->EVPN Exported Prefixes

Prefix	EVPN route status
1234:100:1::/64	Created
1234:100:1:4::/64	Created
1234:100:1:8::/64	Created
1234:100:1:12::/64	Created
1234:100:1:16::/64	Created

EVPN->IPv4 Imported Prefixes

Prefix	Etag	IP route status	
100.2.0.0/22	0	Created	
Route distinguisher	VNI/Label	Router MAC	Nexthop/Overlay GW/ESI
10.255.2.1:100	9101	00:05:86:e1:03:f0	10.255.2.1
10.255.2.2:100	9101	00:05:86:d0:a6:f0	10.255.2.2
100.2.4.0/22	0	Created	
Route distinguisher	VNI/Label	Router MAC	Nexthop/Overlay GW/ESI
10.255.2.1:100	9101	00:05:86:e1:03:f0	10.255.2.1

```

10.255.2.2:100      9101      00:05:86:d0:a6:f0 10.255.2.2
100.2.8.0/22        0          Created
Route distinguisher  VNI/Label  Router MAC      Nexthop/Overlay GW/ESI
10.255.2.1:100      9101      00:05:86:e1:03:f0 10.255.2.1
10.255.2.2:100      9101      00:05:86:d0:a6:f0 10.255.2.2
100.2.12.0/22       0          Created
Route distinguisher  VNI/Label  Router MAC      Nexthop/Overlay GW/ESI
10.255.2.1:100      9101      00:05:86:e1:03:f0 10.255.2.1
10.255.2.2:100      9101      00:05:86:d0:a6:f0 10.255.2.2
100.2.16.0/22       0          Created
Route distinguisher  VNI/Label  Router MAC      Nexthop/Overlay GW/ESI
10.255.2.1:100      9101      00:05:86:e1:03:f0 10.255.2.1
10.255.2.2:100      9101      00:05:86:d0:a6:f0 10.255.2.2

```

EVPN->IPv6 Imported Prefixes

```

Prefix                                Etag      IP route status
1234:100:2::/64                      0          Created
Route distinguisher  VNI/Label  Router MAC      Nexthop/Overlay GW/ESI
10.255.2.1:100      9101      00:05:86:e1:03:f0 10.255.2.1
10.255.2.2:100      9101      00:05:86:d0:a6:f0 10.255.2.2
1234:100:2:4::/64                    0          Created
Route distinguisher  VNI/Label  Router MAC      Nexthop/Overlay GW/ESI
10.255.2.1:100      9101      00:05:86:e1:03:f0 10.255.2.1
10.255.2.2:100      9101      00:05:86:d0:a6:f0 10.255.2.2
1234:100:2:8::/64                    0          Created
Route distinguisher  VNI/Label  Router MAC      Nexthop/Overlay GW/ESI
10.255.2.1:100      9101      00:05:86:e1:03:f0 10.255.2.1
10.255.2.2:100      9101      00:05:86:d0:a6:f0 10.255.2.2
1234:100:2:12::/64                   0          Created
Route distinguisher  VNI/Label  Router MAC      Nexthop/Overlay GW/ESI
10.255.2.1:100      9101      00:05:86:e1:03:f0 10.255.2.1
10.255.2.2:100      9101      00:05:86:d0:a6:f0 10.255.2.2
1234:100:2:16::/64                   0          Created
Route distinguisher  VNI/Label  Router MAC      Nexthop/Overlay GW/ESI
10.255.2.1:100      9101      00:05:86:e1:03:f0 10.255.2.1
10.255.2.2:100      9101      00:05:86:d0:a6:f0 10.255.2.2

```

show evpn ip-prefix-database extensive

```

user@host> show evpn ip-prefix-database extensive
L3 context: VRF-100

```

IPv4->EVPN Exported Prefixes

Prefix: 100.1.0.0/22

EVPN route status: Created

Change flags: 0x0

Advertisement mode: Direct nexthop

Encapsulation: VXLAN

VNI: 9100

Router MAC: 00:05:86:28:90:f0

Prefix: 100.1.4.0/22

EVPN route status: Created

Change flags: 0x0

Advertisement mode: Direct nexthop

Encapsulation: VXLAN

VNI: 9100

Router MAC: 00:05:86:28:90:f0

Prefix: 100.1.8.0/22

EVPN route status: Created

Change flags: 0x0

Advertisement mode: Direct nexthop

Encapsulation: VXLAN

VNI: 9100

Router MAC: 00:05:86:28:90:f0

Prefix: 100.1.12.0/22

EVPN route status: Created

Change flags: 0x0

Advertisement mode: Direct nexthop

Encapsulation: VXLAN

VNI: 9100

Router MAC: 00:05:86:28:90:f0

Prefix: 100.1.16.0/22

EVPN route status: Created

Change flags: 0x0

Advertisement mode: Direct nexthop

Encapsulation: VXLAN

VNI: 9100

Router MAC: 00:05:86:28:90:f0

IPv6->EVPN Exported Prefixes

Prefix: 1234:100:1::/64

EVPN route status: Created

Change flags: 0x0

Advertisement mode: Direct nexthop

Encapsulation: VXLAN

VNI: 9100

Router MAC: 00:05:86:28:90:f0

Prefix: 1234:100:1:4::/64

EVPN route status: Created

Change flags: 0x0

Advertisement mode: Direct nexthop

Encapsulation: VXLAN

VNI: 9100

Router MAC: 00:05:86:28:90:f0

Prefix: 1234:100:1:8::/64

EVPN route status: Created

Change flags: 0x0

Advertisement mode: Direct nexthop

Encapsulation: VXLAN

VNI: 9100

Router MAC: 00:05:86:28:90:f0

Prefix: 1234:100:1:12::/64

EVPN route status: Created

Change flags: 0x0

Advertisement mode: Direct nexthop

Encapsulation: VXLAN

VNI: 9100

Router MAC: 00:05:86:28:90:f0

Prefix: 1234:100:1:16::/64

EVPN route status: Created

Change flags: 0x0

Advertisement mode: Direct nexthop

Encapsulation: VXLAN

VNI: 9100

Router MAC: 00:05:86:28:90:f0

EVPN->IPv4 Imported Prefixes

Prefix: 100.2.0.0/22, Ethernet tag: 0
IP route status: Created
Change flags: 0x0
Remote advertisements:
Route Distinguisher: 10.255.2.1:100
VNI: 9101
Router MAC: 00:05:86:e1:03:f0
BGP nexthop address: 10.255.2.1
Route Distinguisher: 10.255.2.2:100
VNI: 9101
Router MAC: 00:05:86:d0:a6:f0
BGP nexthop address: 10.255.2.2

Prefix: 100.2.4.0/22, Ethernet tag: 0
IP route status: Created
Change flags: 0x0
Remote advertisements:
Route Distinguisher: 10.255.2.1:100
VNI: 9101
Router MAC: 00:05:86:e1:03:f0
BGP nexthop address: 10.255.2.1
Route Distinguisher: 10.255.2.2:100
VNI: 9101
Router MAC: 00:05:86:d0:a6:f0
BGP nexthop address: 10.255.2.2

Prefix: 100.2.8.0/22, Ethernet tag: 0
IP route status: Created
Change flags: 0x0
Remote advertisements:
Route Distinguisher: 10.255.2.1:100
VNI: 9101
Router MAC: 00:05:86:e1:03:f0
BGP nexthop address: 10.255.2.1
Route Distinguisher: 10.255.2.2:100
VNI: 9101
Router MAC: 00:05:86:d0:a6:f0
BGP nexthop address: 10.255.2.2

Prefix: 100.2.12.0/22, Ethernet tag: 0
IP route status: Created
Change flags: 0x0

Remote advertisements:

Route Distinguisher: 10.255.2.1:100

VNI: 9101

Router MAC: 00:05:86:e1:03:f0

BGP nexthop address: 10.255.2.1

Route Distinguisher: 10.255.2.2:100

VNI: 9101

Router MAC: 00:05:86:d0:a6:f0

BGP nexthop address: 10.255.2.2

Prefix: 100.2.16.0/22, Ethernet tag: 0

IP route status: Created

Change flags: 0x0

Remote advertisements:

Route Distinguisher: 10.255.2.1:100

VNI: 9101

Router MAC: 00:05:86:e1:03:f0

BGP nexthop address: 10.255.2.1

Route Distinguisher: 10.255.2.2:100

VNI: 9101

Router MAC: 00:05:86:d0:a6:f0

BGP nexthop address: 10.255.2.2

EVPN->IPv6 Imported Prefixes

Prefix: 1234:100:2::/64, Ethernet tag: 0

IP route status: Created

Change flags: 0x0

Remote advertisements:

Route Distinguisher: 10.255.2.1:100

VNI: 9101

Router MAC: 00:05:86:e1:03:f0

BGP nexthop address: 10.255.2.1

Route Distinguisher: 10.255.2.2:100

VNI: 9101

Router MAC: 00:05:86:d0:a6:f0

BGP nexthop address: 10.255.2.2

Prefix: 1234:100:2:4::/64, Ethernet tag: 0

IP route status: Created

Change flags: 0x0

Remote advertisements:

Route Distinguisher: 10.255.2.1:100

```

VNI: 9101
Router MAC: 00:05:86:e1:03:f0
BGP nexthop address: 10.255.2.1
Route Distinguisher: 10.255.2.2:100
VNI: 9101
Router MAC: 00:05:86:d0:a6:f0
BGP nexthop address: 10.255.2.2

```

Prefix: 1234:100:2:8::/64, Ethernet tag: 0

IP route status: Created

Change flags: 0x0

Remote advertisements:

```

Route Distinguisher: 10.255.2.1:100
VNI: 9101
Router MAC: 00:05:86:e1:03:f0
BGP nexthop address: 10.255.2.1
Route Distinguisher: 10.255.2.2:100
VNI: 9101
Router MAC: 00:05:86:d0:a6:f0
BGP nexthop address: 10.255.2.2

```

Prefix: 1234:100:2:12::/64, Ethernet tag: 0

IP route status: Created

Change flags: 0x0

Remote advertisements:

```

Route Distinguisher: 10.255.2.1:100
VNI: 9101
Router MAC: 00:05:86:e1:03:f0
BGP nexthop address: 10.255.2.1
Route Distinguisher: 10.255.2.2:100
VNI: 9101
Router MAC: 00:05:86:d0:a6:f0
BGP nexthop address: 10.255.2.2

```

Prefix: 1234:100:2:16::/64, Ethernet tag: 0

IP route status: Created

Change flags: 0x0

Remote advertisements:

```

Route Distinguisher: 10.255.2.1:100
VNI: 9101
Router MAC: 00:05:86:e1:03:f0
BGP nexthop address: 10.255.2.1
Route Distinguisher: 10.255.2.2:100

```



```
VNI: 9101
Router MAC: 00:05:86:d0:a6:f0
BGP nexthop address: 10.255.2.2
```

Release Information

Command introduced in Junos OS Release 15.1X53-D30.

RELATED DOCUMENTATION

[show evpn l3-context](#) | [1862](#)

show route table

show evpn l3-context

IN THIS SECTION

- [Syntax](#) | [1862](#)
- [Description](#) | [1863](#)
- [Options](#) | [1863](#)
- [Additional Information](#) | [1863](#)
- [Required Privilege Level](#) | [1863](#)
- [Output Fields](#) | [1863](#)
- [Sample Output](#) | [1865](#)
- [Release Information](#) | [1866](#)

Syntax

```
show evpn l3-context
<brief | extensive>
<logical-system logical-system-name>
<routing-instance-name>
```

Description

Display EVPN database information for Layer 3 virtual routing and forwarding (VRF) instances.

Options

- none** Display standard EVPN database information for all Layer 3 VRF instances.
- brief | extensive** (Optional) Display the specified level of output.
- logical-system <logical-system-name>** (Optional) Display Layer 3 information for the specified logical system or all logical systems.
- routing-instance-name** (Optional) Display EVPN database information for the specified Layer 3 VRF instance.

Additional Information

Required Privilege Level

view

Output Fields

Table 74 on page 1863 lists the output fields for the show evpn l3-context command. Output fields are listed in the approximate order in which they appear.

Table 74: show evpn l3-context Output Fields

Field Name	Field Description	Level of Output
L3 context	Name of VRF instance.	All levels
Type	Status of VRF instance	All levels
Fwd	Type of forwarding route	level-of-output none

Table 74: show evpn l3-context Output Fields (Continued)

Field Name	Field Description	Level of Output
Encap	Type of encapsulation.	level-of-output none
VNI	Virtual network identifier for VRF instance.	All levels
Router MAC/GW intf	For a pure type-5 route, the router MAC provides next-hop reachability information and the MAC address of the sending device For a standard type-5 route, the GW interface provides the next-hop route.	All levels
Advertisement mode	Type of forwarding route.	level-of-output Extensive
IPv4 source VTEP address	IPv4 source address for the Virtual Extensible LAN (VXLAN) tunnel.	level-of-output Extensive
IPv6 source VTEP address	IPv6 source address for the Virtual Extensible LAN (VXLAN) tunnel.	level-of-output Extensive
IP->EVPN export policy	Name of export routing policy applied to forwarded IP routes.	level-of-output extensive
Flags	Enabled trace flags.	level-of-output extensive
Change flags	Changed trace flags.	level-of-output extensive
Composite nexthop	Status of next-hop route: Enabled or Disabled.	level-of-output extensive
Route distinguisher	Route distinguisher identifier of the VRF instance	level-of-output extensive

Table 74: show evpn l3-context Output Fields (Continued)

Field Name	Field Description	Level of Output
Reference count		level-of-output extensive

Sample Output

show evpn l3-context

```

user@DC1_SPINE1_RE> show evpn l3-context
L3 context          Type Fwd  Encap VNI/Label Router MAC/GW intf
VRF-100             Cfg  Sym  VXLAN 9100      00:05:86:28:90:f0
VRF-200             Cfg  Sym  VXLAN 9200      00:05:86:28:90:f0

```

show evpn l3-context extensive

```

user@DC1_SPINE1_RE> show evpn l3-context extensive
L3 context: VRF-100
Type: Configured
  Advertisement mode: Direct nexthop, Router MAC: 00:05:86:28:90:f0
  Encapsulation: VXLAN, VNI: 9100
  IPv4 source VTEP address: 10.255.1.1
  IPv6 source VTEP address: abcd::128:102:242:145
  IP->EVPN export policy: EVPN-TYPE5-EXPORT-VRF-100
  Flags: 0x19 <Configured IRB-MAC New>
  Change flags: 0xe <Export-Policy Fwd-Mode Encap>
  Composite nexthop support: Enabled
  Route Distinguisher: 10.255.1.1:100
  Reference count: 21

L3 context: VRF-200
Type: Configured
  Advertisement mode: Direct nexthop, Router MAC: 00:05:86:28:90:f0
  Encapsulation: VXLAN, VNI: 9200
  IPv4 source VTEP address: 10.255.1.1

```

```
IPv6 source VTEP address: abcd::128:102:242:145
IP->EVPN export policy: EVPN-TYPE5-EXPORT-VRF-200
Flags: 0x19 <Configured IRB-MAC New>
Change flags: 0xe <Export-Policy Fwd-Mode Encap>
Composite nexthop support: Enabled
Route Distinguisher: 10.255.1.1:200
Reference count: 21
```

Release Information

Statement introduced in Junos OS Release 15.1X53-D30.

RELATED DOCUMENTATION

[show evpn ip-prefix-database](#) | [1852](#)

show evpn mac-ip-table

IN THIS SECTION

- [Syntax](#) | [1866](#)
- [Description](#) | [1867](#)
- [Options](#) | [1867](#)
- [Required Privilege Level](#) | [1867](#)
- [Output Fields](#) | [1867](#)
- [Sample Output](#) | [1869](#)
- [Release Information](#) | [1870](#)

Syntax

```
show evpn mac-ip-table
<brief | count | detail | detail | extensive>
```

```
<instance>
<kernel >
<logical-system (all | logical-system-name)>
```

Description

Displays the MAC-IP address for all IPv4 (ARP) and IPv6 (ND) bindings for routing instances where the instance-type is evpn.

Options

- brief | detail | extensive | summary** (Optional) Display the specified level of output.
- instance *instance-name*** (Optional) Display the MAC-IP address information for the specified routing instance.
- kernel** (Optional) Display the MAC-IP address entries in the kernel
- logical-system *<logical-system-name>*** (Optional) Display the MAC-IP address information for the specified logical system or all logical systems.

Required Privilege Level

maintenance

Output Fields

[Table 75 on page 1867](#) lists the output fields for the show evpn mac-ip-table command. Output fields are listed in the approximate order in which they appear.

Table 75: show evpn mac-ip-table Output Fields

Field Name	Field Description	Level of Output
IP address	IP address associated with the EVPN routing instance.	All levels
MAC address	MAC address	All levels

Table 75: show evpn mac-ip-table Output Fields *(Continued)*

Field Name	Field Description	Level of Output
Flags	<p>MAC IP flags: Identifies : statically installed MAC-IP entries.</p> <ul style="list-style-type: none"> • S—Statically installed MAC-IP entries. • D—Dynamic installed MAC-IP address entries. • L—Locally learned MAC-IP address entries • R— MAC-IP address entries that are learnt from a remote device via the control plane. • K—MAC-IP address entries that are installed in the kernel. • RT—Destination Route that corresponds to an installed entry. • AD—Advertised to remote device. • RE—ARP/ND entry has expired and another arp or network solicitation request has been sent. • R0—Router • OV—Override 	Brief, detail
Logical Interface	The logical interface associated with the routing instance.	Brief, detail
Active Source	The source of the learned MAC-IP address entry. Displays the ESI or router ID.	All levels

Table 75: show evpn mac-ip-table Output Fields *(Continued)*

Field Name	Field Description	Level of Output
Routing Instance	Information on the Routing instance where the MAC-IP address entry was learned. <ul style="list-style-type: none"> • Bridging domain • MAC-IP local mask • MAC-IP remote mask • MAC-IP RPD flags • MAC-IP flags 	Extensive
BD ID	Bridge Domain ID	Brief

Sample Output

show evpn mac-ip-table

```

user@host> show evpn mac-ip-table
MAC IP flags (S - Static, D - Dynamic, L - Local , R - Remote, K - Kernel, RT - Dest Route,
              AD - Advt to remote, RE - Re-ARP/ND, RO - Router, OV - Override)
Routing instance : evpn
Bridging domain : bd1
  IP          MAC          Flags          Logical          Active
  address     address                Interface        source
  10.1.1.1    00:11:00:00:00:01    DR,K
81.1.1.1
  10.1.1.2    00:22:00:00:00:01    DR,K
81.1.2.2
  10.1.1.3    00:33:00:00:00:01    DL,K,AD        ge-0/0/2.1
  10.1.1.4    00:44:00:00:00:01    DR,K
81.4.4.4

```


show evpn mac-ip-table extensive

```
user@host> show evpn mac-ip-table extensive
IP address: 10.1.1.1
MAC address: 00:11:00:00:00:01
Routing instance: evpn
Bridging domain: bd1
MAC-IP local mask: 0x0000000000000000
MAC-IP remote mask: 0x8000000000000000
MAC-IP RPD flags: 0x00000081
MAC-IP flags: remote, kernel
```

show evpn mac-ip-table kernel

```
user@host> show evpn mac-ip-table kernel
```

BD	IP	MAC	Logical	Flags
id	address	address	Interface	
2	192.168.1.1	00:21:59:aa:77:f0	irb.1	0x00000209
2	192.168.1.100	00:00:5e:00:01:01	irb.1	0x00000609
3	192.168.2.1	00:21:59:aa:77:f0	irb.2	0x00000209
3	192.168.2.100	00:00:5e:00:01:01	irb.2	0x00000609
4	192.168.3.1	00:21:59:aa:77:f0	irb.3	0x00000209
4	192.168.3.100	00:00:5e:00:01:01	irb.3	0x00000609
5	192.168.4.1	00:21:59:aa:77:f0	irb.4	0x00000209
5	192.168.4.100	00:00:5e:00:01:01	irb.4	0x00000609

Release Information

Command introduced in Junos OS Release 17.4R2.

show evpn mac-table

IN THIS SECTION

- Syntax | 1871

- [Description | 1871](#)
- [Options | 1871](#)
- [Required Privilege Level | 1872](#)
- [Release Information | 1872](#)

Syntax

```
show evpn mac-table
<age>
<address>
<brief | count | detail | extensive | summary>
<instance instance-name>
<interface interface-name>
<isid <isid>>
<logical-system <logical-system-name>>
<vlan-id vlan-id>
```

Description

Show Ethernet VPN (EVPN) MAC table information.

Options

none	Display brief information about the EVPN MAC table.
age	(Optional) Display age of a single mac-address.
address	(Optional) Display MAC table information for the specified MAC address.
brief count detail extensive summary	(Optional) Display the specified level of output.
instance <i>instance-name</i>	(Optional) Display MAC table information for a specific routing instance.
isid <i><isid></i>	(Optional) Display MAC table information for the specified ISID or all ISIDs.

<code>logical-system <logical-system-name></code>	(Optional) Display MAC table information for the specified logical system or all logical systems.
<code>vlan-id <vlan-id></code>	(Optional) Display MAC table information for the specified VLAN.

Required Privilege Level

view

Release Information

Command introduced in Junos OS Release 14.2.

RELATED DOCUMENTATION

[Example: Configuring an EVPN with IRB Solution on EX9200 Switches](#) | 995

show evpn mld-snooping database

IN THIS SECTION

- [Syntax](#) | 1873
- [Syntax for mac-vrf routing](#) | 1873
- [Description](#) | 1873
- [Options](#) | 1873
- [Required Privilege Level](#) | 1874
- [Output Fields](#) | 1874
- [Sample Output](#) | 1875
- [Release Information](#) | 1876

Syntax

```
show evpn mld-snooping database
<extensive>
<esi id>
<group group-address>
<instance id>
<interface name>
<l2-domain-id id>
```

Syntax for mac-vrf routing

```
show mac-vrf routing mld-snooping database
<extensive>
<esi id>
<group group-address>
<instance id>
<interface name>
<l2-domain-id id>
```

Description

Display MLD snooping information for IPv6 multicast forwarding in an EVPN network.

Options

none	Display detailed information.
extensive	(Optional) Display more extensive output.
esi	(Optional) Display output only for a specified EVPN Ethernet segment ID (ESI).
group <i>group-address</i>	(Optional) Display output only for the specified IPv6 multicast group address range.
instance <i>instance-name</i>	(Optional) Display information for the specified EVPN instance (EVI).
interface <i>name</i>	(Optional) Display information about the specified interface.

I2-domain-id *id* (Optional) Display output for the specified bridging domain, VLAN, or virtual network identifier (VNI).

Required Privilege Level

view

Output Fields

Table 76 on page 1874 lists the output fields for the show evpn mld-snooping database command. Output fields are listed in the approximate order in which they appear.

Table 76: show evpn mld-snooping database Output Fields

Field Name	Field Description	Level of Output
Instance	Name of the EVPN routing instance for which information is displayed.	All levels
VN Identifier	Tunnel network identifier.	All levels
Group IP	Multicast IP address of the multicast group.	All levels
Source IP	IP address of the multicast data source for the group.	All levels
Access OIF Count	Number of outgoing interface (OIF) list entries on the access side.	All levels
Access OIF Count (more information)	More OIF Entry Information: <ul style="list-style-type: none"> Interface—Interface name ESI—Associated EVPN Ethernet Segment ID for the interface Local—Local receiver. Remote—Remote receiver. 	extensive

Sample Output

show evpn mld-snooping database

```

user@device> show evpn mld-snooping database
Instance: EVPN-2
  VN Identifier: 100
    Group IP: ff1e::25:25:1, Source IP: ::, Access OIF Count: 1
    Group IP: ff1e::33:33:1, Source IP: ::, Access OIF Count: 1

Instance: EVPN-4
  VN Identifier: 200
    Group IP: ff1e::25:25:1, Source IP: ::, Access OIF Count: 1
    Group IP: ff1e::33:33:1, Source IP: ::, Access OIF Count: 1

```

show evpn mld-snooping database instance instance-name

```

user@device> show evpn mld-snooping database
EVPN-4
Instance: EVPN-4
  VN Identifier: 200
    Group IP: ff1e::25:25:1, Source IP: ::, Access OIF Count: 1
    Group IP: ff1e::33:33:1, Source IP: ::, Access OIF Count: 1

```

show evpn mld-snooping database extensive

```

user@device> show evpn mld-snooping database
extensive
Instance: EVPN-2
  VN Identifier: 100
    Group IP: ff1e::25:25:1, Source IP: ::
      Access OIF Count: 1
        Interface          ESI                      Local  Remote
        ae3.0              00:22:22:22:22:22:22:22 0          1
    Group IP: ff1e::33:33:1, Source IP: ::
      Access OIF Count: 1

```

```

      Interface          ESI          Local Remote
      ae3.0             00:22:22:22:22:22:22:22:22 1      0

Instance: EVPN-4
  VN Identifier: 200
    Group IP: ff1e::25:25:1, Source IP: ::
      Access 0IF Count: 1
        Interface          ESI          Local Remote
        ae4.0             00:44:44:44:44:44:44:44:44 0      1
    Group IP: ff1e::33:33:1, Source IP: ::
      Access 0IF Count: 1
        Interface          ESI          Local Remote
        ae4.0             00:44:44:44:44:44:44:44:44 1      0

```

Release Information

Command introduced in Junos OS Release 18.4R1.

Introduced in Junos OS Release 19.4R2 on ACX Series routers.

RELATED DOCUMENTATION

[Configure Multicast Forwarding with MLD Snooping in an EVPN-MPLS Environment | 1038](#)

[Overview of Multicast Forwarding with IGMP or MLD Snooping in an EVPN-MPLS Environment | 1013](#)

[Overview of Multicast Forwarding with IGMP Snooping or MLD Snooping in an EVPN-VXLAN Environment | 675](#)

show evpn multicast-snooping assisted-replication multihomed-peers

IN THIS SECTION

- [Syntax | 1877](#)
- [Syntax for mac-vrf routing | 1877](#)
- [Description | 1877](#)

- Options | 1878
- Required Privilege Level | 1878
- Output Fields | 1878
- Sample Output | 1879
- Release Information | 1879

Syntax

```
show evpn multicast-snooping assisted-replication multihomed-peers
<extensive>
<instance evpn-instance-name>
```

Syntax for mac-vrf routing

```
show mac-vrf routing multicast-snooping assisted-replication multihomed-peers
<extensive>
<instance mac-vrf-routing-instance-name>
```

Description

Display multihoming peer neighbor addresses and corresponding multihomed Ethernet segment information for an assisted replication (AR) leaf device.

When devices with AR enabled operate in extended AR mode (see ["Extended AR Mode for Multihomed Ethernet Segments" on page 758](#)), an ingress AR leaf device uses the information this command shows to replicate and forward multicast traffic to all other AR leaf devices that are its multihoming peers (have attached access devices with the same multihomed Ethernet segment identifier [ESI]). The ingress AR leaf device also sends the traffic to an AR replicator device that replicates it to other devices in the EVPN network that are not multihoming peers with the AR leaf device.

NOTE: QFX Series devices must use extended AR mode to support AR in an EVPN network with a Virtual Extensible LAN (VXLAN) overlay and multihomed Ethernet segments.

Output for this command is only available when IGMP snooping is enabled.

Options

- extensive** Display more details about multihoming peers.
- instance *evpn-instance-name*** (Optional) Display information only for the specified EVPN instance.

NOTE: You can only enable IGMP snooping for the default-switch EVPN instance on QFX Series switches in an EVPN-VXLAN environment, so you don't need to use the `instance` option because this command only displays information for the default-switch instance.

Required Privilege Level

view

Output Fields

Table 77 on page 1878 describes the output fields for the `show evpn multicast-snooping assisted-replication multihomed-peers` command. Output fields are listed in the approximate order in which they appear.

Table 77: show evpn multicast-snooping assisted-replication multihomed-peers Output Fields

Field Name	Field Description
Instance	EVPN instance name.
Neighbor Address	Multihoming peer IP address.
Nexthop Index	Ingress replication next hop for the multihoming peer AR leaf device in the Neighbor Address output field.
Interface	Interface name to reach the multihoming peer device in the Neighbor Address output field.
Local Multi-homed ESIs	(Displayed with extensive option only) Multihomed Ethernet segments shared with the multihoming peer in the Neighbor Address output field.

Sample Output

show evpn multicast-snooping assisted-replication multihomed-peers

```
user@device> show evpn multicast-snooping assisted-replication
multihomed-peers
Instance: default-switch
  Neighbor Address  Nexthop Index  Interface
  10.0.1.2          1841           vtep.32782
```

show evpn multicast-snooping assisted-replication multihomed-peers (extensive)

```
user@device> show evpn multicast-snooping assisted-replication
multihomed-peers extensive
Instance: default-switch

  Neighbor address: 10.0.1.2
  Nexthop Index: 1841
  Interface: vtep.32782
  Local Multi-homed ESIs
    00:11:22:33:44:55:66:77:88:99
    00:11:22:33:44:55:66:77:88:aa
```

Release Information

Command introduced in Junos OS Release 18.4R2 and 19.4R1.

RELATED DOCUMENTATION

[Assisted Replication Multicast Optimization in EVPN Networks | 752](#)

[Extended AR Mode for Multihomed Ethernet Segments | 758](#)

show evpn multicast-snooping assisted-replication next-hops

IN THIS SECTION

- [Syntax | 1880](#)
- [Description | 1880](#)
- [Options | 1881](#)
- [Required Privilege Level | 1881](#)
- [Output Fields | 1881](#)
- [Sample Output | 1882](#)
- [Release Information | 1883](#)

Syntax

```
show evpn multicast-snooping assisted-replication next-hops  
<index next-hop-ID>  
<instance evpn-instance-name>  
<l2-domain ID>
```

Description

Display next-hop information on an assisted replication (AR) leaf device that the AR leaf uses for load-balancing among available AR replicators.

- AR leaf devices in the QFX5000 line of switches perform load-balancing among the available AR replicators by designating a particular AR replicator device for a VLAN or VXLAN network identifier (VNI).
- AR leaf devices in the QFX10000 line of switches load-balance among the available AR replicators actively based on traffic flow levels within a VNI.

An AR device will have an AR VXLAN encapsulation next hop and an ingress replication VXLAN encapsulation next hop towards a particular AR replicator. Load-balancing next hops are AR VXLAN encapsulation next hops towards the AR replicator.

Output for this command is only available when IGMP snooping is enabled.

Options

<code>index <i>nh-index</i></code>	(Optional) Display information only for the specified next-hop.
<code>instance <i>evpn-instance-name</i></code>	<div>(Optional) Display information only for the specified EVPN instance.<div>NOTE: You can only enable IGMP snooping for the default-switch EVPN instance on QFX Series switches in an EVPN-VXLAN environment, so you don't need to use the <code>instance</code> option because this command only displays information for the default-switch instance.</div></div>
<code>l2-domain-id <i>ID</i></code>	<div>(Optional) Display information only for the specified Layer 2 domain ID, which can be a VLAN ID, a VXLAN network identifier (VNI), or a service identifier (I-SID) within the range from 1 through 16777214.<div>NOTE: For QFX Series switches in an EVPN-VXLAN environment, you must specify the Layer 2 domain ID as a VNI.</div></div>

Required Privilege Level

view

Output Fields

Table 78 on page 1881 describes the output fields for the `show evpn multicast-snooping assisted-replication next-hops` command. Output fields are listed in the approximate order in which they appear.

Table 78: `show evpn multicast-snooping assisted-replication next-hops` Output Fields

Field Name	Field Description
Instance	EVPN instance name.
VN Identifier	Lists AR load-balancing next-hop information about this VNI.

Table 78: show evpn multicast-snooping assisted-replication next-hops Output Fields (Continued)

Field Name	Field Description
Load Balance Nexthop Index	Index of the load-balancing next hop (top level).
Load balance to	Lists next-hop information for available AR replicators.
Nexthop Index	AR next hop for the AR replicator in the AR IP output field.
Interface	AR tunnel interface name to this AR replicator.
AR IP	IP address of this AR replicator (the configured secondary loopback interface address for AR).
(Designated Node)	<p>(AR leaf devices in the QFX5000 line of switches only) Designated AR replicator for a VLAN or VNI for load-balancing AR operation among the available AR replicators.</p> <p>NOTE: AR leaf devices in the QFX10000 line of switches load-balance among the available AR replicators based on traffic flow levels within a VNI, and don't show a designated replicator in the output of this command.</p>

Sample Output

show evpn multicast-snooping assisted-replication next-hops

```

user@device> show evpn multicast-snooping assisted-replication
next-hops
Instance: default-switch

VN Identifier: 100
Load Balance Nexthop Index: 131091
Load balance to:
Nexthop Index    Interface    AR IP
1797             vtep.32772  192.168.6.6
1772             vtep.32770  192.168.5.5 (Designated Node)

```

VN Identifier: 101		
Load Balance Nexthop Index: 131092		
Load balance to:		
Nexthop Index	Interface	AR IP
1797	vtep.32772	192.168.6.6 (Designated Node)
1772	vtep.32770	192.168.5.5

Release Information

Command introduced in Junos OS Release 18.4R2 and 19.4R1.

RELATED DOCUMENTATION

- [Assisted Replication Multicast Optimization in EVPN Networks | 752](#)
- [AR Leaf Device Load Balancing With Multiple Replicators | 759](#)

show evpn multicast-snooping assisted-replication replicators

IN THIS SECTION

- Syntax | 1884
- Description | 1884
- Options | 1884
- Required Privilege Level | 1885
- Output Fields | 1885
- Sample Output | 1886
- Release Information | 1887

Syntax

```
show evpn multicast-snooping assisted-replication replicators
<instance evpn-instance-name>
<l2-domain-id ID>
```

Description

Displays information about the assisted replication (AR) replicator devices advertised in an EVPN network and the mode in which they operate.

Extended AR mode enables AR to work with AR replicator devices that can't retain the ingress AR leaf device IP address as the multicast source when replicating and forwarding the traffic into the EVPN core, usually due to restrictions with the overlay tunnel type. QFX series AR devices require this mode to support AR in an EVPN network with a Virtual Extensible LAN (VXLAN) overlay that might have multihomed Ethernet segments.

Output for this command is only available when IGMP snooping is enabled.

Options

instance
evpn-
instance-
name

(Optional) Display information only for the specified EVPN instance.

NOTE: You can only enable IGMP snooping for the default-switch EVPN instance on QFX Series switches in an EVPN-VXLAN environment, so you don't need to use the *instance* option because this command only displays information for the default-switch instance.

l2-domain-id
ID

(Optional) Display information only for the specified Layer 2 domain ID, which can be a VLAN ID, a VXLAN network identifier (VNI), or a service identifier (I-SID) within the range from 1 through 16777214.

NOTE: For QFX Series switches in an EVPN-VXLAN environment, you must specify the Layer 2 domain ID as a VNI.

Required Privilege Level

view

Output Fields

Table 79 on page 1885 describes the output fields for the `show evpn multicast-snooping assisted-replication replicators` command. Output fields are listed in the approximate order in which they appear.

Table 79: show evpn multicast-snooping assisted-replication replicators Output Fields

Field Name	Field Description
Instance	EVPN instance name.
AR Role	AR role of the device where you run this command: <ul style="list-style-type: none"> • AR Leaf • AR Replicator
VN Identifier	Lists information about AR replicators available to this VNI.
Operational Mode	AR mode in which this AR device is operating based on conditions in the network for AR that the device detects: <ul style="list-style-type: none"> • Misconfiguration/Ingress Replication—The AR device detected a misconfiguration in which AR won't operate correctly, such as when an AR replicator and AR leaf share a multihomed Ethernet segment. The AR leaf device will not use AR, and defaults to using regular ingress replication. • No replicators/Ingress Replication—The AR leaf device has not learned about any AR replicator devices, and defaults to using regular ingress replication instead of AR. • Extended AR—Ingress AR leaf devices should replicate multicast traffic to all other AR leaf devices that are its multihoming peers (have attached access devices in the same multihomed Ethernet segment). This AR replicator device will not duplicate traffic to those multihoming peers. • Regular AR—When ingress AR leaf devices send multicast traffic to this AR replicator device, they will not send the traffic to their multihoming peers. This AR replicator device will do the replication and forwarding to the other devices in the EVPN network, including the ingress AR leaf device's multihoming peers.

Table 79: show evpn multicast-snooping assisted-replication replicators Output Fields (Continued)

Field Name	Field Description
Replicator IP	IP address of an available AR replicator (the configured loopback address for AR).
Nexthop Index	Next hops index for the AR replicator in the Replicator IP output field.
Interface	<p>Tunnel interface name for the AR replicator in the Replicator IP output field.</p> <p>NOTE: An AR device will have an AR VXLAN encapsulation next hop and an ingress replication VXLAN encapsulation next hop towards a particular AR replicator. This field displays the AR VXLAN encapsulation next hop towards the AR replicator.</p>
Mode	<p>AR mode advertised by the AR replicator in the Replicator IP output field:</p> <ul style="list-style-type: none"> Extended AR—Ingress AR leaf devices should replicate multicast traffic to all other AR leaf devices that are its multihoming peers (have attached access devices in the same multihomed Ethernet segment). This AR replicator device will not duplicate traffic to those multihoming peers. Regular AR—When ingress AR leaf devices send multicast traffic to this AR replicator device, they will not send the traffic to their multihoming peers. This AR replicator device will do the replication and forwarding to the other devices in the EVPN network, including the ingress AR leaf device's multihoming peers.

Sample Output

show evpn multicast-snooping assisted-replication replicators (on AR replicator device)

```

user@device> show evpn multicast-snooping assisted-replication
replicators
Instance: default-switch
AR Role: AR Replicator

```

```
VN Identifier: 100
```

```
Operational Mode: Extended AR
```

Replicator IP	Nexthop Index	Interface	Mode
192.168.6.6	0	Self	Extended AR
192.168.5.5	1875	vtep.32780	Extended AR

```

VN Identifier: 101
Operational Mode: Extended AR
  Replicator IP  Nexthop Index  Interface  Mode
  192.168.6.6    0              Self       Extended AR
  192.168.5.5    1875           vtep.32780 Extended AR

```

show evpn multicast-snooping assisted-replication replicators (on AR leaf device)

```

user@device> show evpn multicast-snooping assisted-replication
replicators
Instance: default-switch
AR Role: AR Leaf

VN Identifier: 100
Operational Mode: Extended AR
  Replicator IP  Nexthop Index  Interface  Mode
  192.168.5.5    1772           vtep.32770 Extended AR
  192.168.6.6    1797           vtep.32772 Extended AR

VN Identifier: 101
Operational Mode: Extended AR
  Replicator IP  Nexthop Index  Interface  Mode
  192.168.5.5    1772           vtep.32770 Extended AR
  192.168.6.6    1797           vtep.32772 Extended AR

```

Release Information

Command introduced in Junos OS Release 18.4R2 and 19.4R1.

RELATED DOCUMENTATION

[Assisted Replication Multicast Optimization in EVPN Networks | 752](#)

[Extended AR Mode for Multihomed Ethernet Segments | 758](#)

[show evpn multicast-snooping assisted-replication next-hops | 1880](#)

show evpn multicast-snooping status

IN THIS SECTION

- [Syntax | 1888](#)
- [Description | 1888](#)
- [Options | 1888](#)
- [Required Privilege Level | 1889](#)
- [Output Fields | 1889](#)
- [Sample Output | 1890](#)
- [Release Information | 1890](#)

Syntax

```
show evpn multicast-snooping status  
<extensive>
```

Description

Display IGMP snooping (for IPv4 multicast traffic) status or MLD snooping (for IPv6 multicast traffic) status in the default-switch instance on a device in an EVPN fabric.

You can enable IGMP snooping (or MLD snooping) in an EVPN network to help optimize multicast traffic flow in the fabric. The devices in the fabric use the following EVPN route types to synchronize the multicast state between multihoming peer provider edge (PE) devices:

- EVPN Type 7 Join Sync routes
- EVPN Type 8 Leave Sync routes

Options

extensive Display more detailed output.

Required Privilege Level

view

Output Fields

Table 1 lists the output fields for the show evpn multicast-snooping status command.

Table 80: show evpn multicast-snooping status Output Fields

Field Name	Field Description
Instance	Layer 2 routing instance. The instance corresponds to the default-switch instance on QFX Series devices in EVPN-VXLAN fabrics.
VLAN ID	VLAN ID.
Multicast Address Family	Show enabled or disabled multicast snooping status for: <ul style="list-style-type: none">• INET—IGMP snooping (IPv4 multicast traffic).• INET6—MLD snooping (IPv6 multicast traffic).
SG Sync	<p>EVPN multicast snooping status for the indicated multicast traffic family—Enabled or Disabled.</p> <p>Extensive output shows the following additional statistics with multicast snooping status Enabled:</p> <ul style="list-style-type: none">• Number of local join-sync SGs• Number of remote join-sync SGs• Number of local proxy SGs• Number of remote proxy SGs

Sample Output

show evpn multicast-snooping status

```
user@host> show evpn multicast-snooping status
Instance: __default_evpn__

Instance: default-switch
  VLAN ID: 1
    Multicast Address Family: INET
      SG Sync: Enabled
    Multicast Address Family: INET6
      SG Sync: Disabled
```

Release Information

RELATED DOCUMENTATION

[Overview of Multicast Forwarding with IGMP or MLD Snooping in an EVPN-MPLS Environment | 1013](#)

[Overview of Multicast Forwarding with IGMP Snooping or MLD Snooping in an EVPN-VXLAN Environment | 675](#)

show evpn nd-table

IN THIS SECTION

- [Syntax | 1891](#)
- [Description | 1891](#)
- [Options | 1891](#)
- [Required Privilege Level | 1891](#)
- [Output Fields | 1891](#)
- [Sample Output | 1892](#)

Syntax

```
show evpn nd-table
<address>
<brief |count | detail | extensive>
<instance instance-name>
```

Description

Show Ethernet VPN (EVPN) Network Discovery Protocol (NDP) entries associated with learned MAC addresses.

Options

- none** Display brief information about the EVPN NDP table.
- address** (Optional) Display NDP information for the specified MAC address.
- brief | count | detail |extensive** (Optional) Display the specified level of output.
- instance <*instance-name*>** (Optional) Display NDP information for the specified routing instance .

Required Privilege Level

view

Output Fields

[Table 81 on page 1892](#) lists the output fields for the `show evpn nd-table` command. Output fields are listed in the approximate order in which they appear.

Table 81: show evpn nd-table Output Fields

Field Name	Field Description	Level of Output
INET address	The INET address related to the INET entries that are added to the NDP table.	All levels
MAC address	MAC addresses learned through NDP.	brief, detail, extensive, instance, mac-address,,
Logical Interface	Logical interface associated with the routing instance in which the NDP INET address is learned.	brief, instance, mac-address,,
Routing instance	Routing instance in which the NDP INET address is learned.	all levels
Bridging domain	Bridging domain in which the NDP INET address is learned.	all levels
Learning interface	Interface on which the NDP INET address is learned.	detail, extensive
Count	Indicates the number of NDP INET addresses learned in a routing instance in a bridge domain.	count

Sample Output

show evpn nd-table

```

user@host> show evpn nd-table
INET          MAC          Logical      Routing      Bridging
address       address      interface    instance     domain
8001::2       00:05:86:a0:dc:f0  irb.0        evpn_1       __evpn_1__

```

show evpn nd-table 00:05:86:a0:dc:f0 (MAC address)

```

user@host> show evpn nd-table 00:05:86:a0:dc:f0
INET          MAC          Logical    Routing    Bridging
address       address       interface  instance   domain
8001::2       00:05:86:a0:dc:f0  irb.0      evpn1      __evpn1__

```

show evpn nd-table brief

```

user@host> show evpn nd-table brief
INET          MAC          Logical    Routing    Bridging
address       address       interface  instance   domain
8001::2       00:05:86:a0:dc:f0  irb.0      evpn1      __evpn1__

```

show evpn nd-table detail

```

user@host> show evpn nd-table detail

INET address: 8001::2
MAC address: 00:05:86:a0:dc:f0
  Routing instance: evpn1
  Bridging domain: __evpn1__
  Learning interface: irb.0

```

show evpn nd-table count

```

user@switch> show evpn nd-table count
1 NDP INET addresses learned in routing instance evpn1 bridge domain __evpn1__

```

show evpn and-table extensive

```

user@host> show evpn nd-table extensive

INET address: 8001::2
MAC address: 00:05:86:a0:dc:f0
  Routing instance: evpn1

```



```
Bridging domain: __evpn1__
Learning interface: irb.0
```

show evpn ndp-table instance evpn1

```
user@host> show evpn arp-table instance evpn1
```

INET	MAC	Logical	Routing	Bridging
address	address	interface	instance	domain
8001::2	00:05:86:a0:dc:f0	irb.0	evpn1	__evpn1__

Release Information

Command introduced in Junos OS Release 16.2.

show evpn oism

IN THIS SECTION

- [Syntax | 1894](#)
- [Description | 1895](#)
- [Options | 1895](#)
- [Required Privilege Level | 1895](#)
- [Output Fields | 1895](#)
- [Sample Output | 1896](#)
- [Release Information | 1896](#)

Syntax

```
show evpn oism
<extensive>
<l3-context l3-vrf-name>
```

Description

Display information about optimized inter-subnet multicast (OISM) configured on the devices in an EVPN-VXLAN fabric.

Options

extensive (Optional) Display more detailed output. Without this option, the command displays brief output by default.

l3-context l3-vrf-name (Optional) Display information only for the specified Layer 3 virtual routing and forwarding (VRF) routing instance.

Without this option, this command displays information for all Layer 3 VRF routing instances where you enabled OISM.

Required Privilege Level

view

Output Fields

Table 1 lists the output fields for the show evpn oism command.

Table 82: show evpn oism Output Fields

Field Name	Field Description
L3 context (default) or EVPN L3 context (extensive)	VRF routing instance name.
SBD (default) or OISM SBD interface (extensive)	IRB interface of the supplemental bridge domain (SBD) that you configure in the corresponding VRF routing instance when you enable OISM.
	If you don't enable OISM in the VRF routing instance, this command displays (null) in this field.

Sample Output

show evpn oism

```
user@host> show evpn oism
L3 context          SBD
DELTA-VRF           irb.901
GAMMA-VRF           irb.902
```

show evpn oism l3-context extensive

```
user@host> show evpn oism l3-context DELTA-VRF extensive
EVPN L3 context: DELTA
OISM SBD interface: irb.901
```

Release Information

Command introduced in Junos OS Release 21.2R1.

RELATED DOCUMENTATION

[Optimized Inter-Subnet Multicast in EVPN Networks | 773](#)

show evpn p2mp

IN THIS SECTION

- [Syntax | 1897](#)
- [Description | 1897](#)
- [Options | 1897](#)
- [Required Privilege Level | 1897](#)
- [Output Fields | 1897](#)

- Sample Output | 1899
- Release Information | 1899

Syntax

```
show evpn p2mp
<brief | extensive>
<instance instance-name>
<logical-system (all | logical-system-name)>
```

Description

Displays EVPN P2MP information.

Options

brief detail extensive	(Optional) Display the specified level of output.
instance <i>instance-name</i>	(Optional) Display routing instance information for the specified instance only.
logical-system (all <i>logical-system-name</i>)	(Optional) Display the EVPN P2MP information of all logical systems or a particular logical system.

Required Privilege Level

View

Output Fields

[Table 83 on page 1898](#) lists the output fields for the show evpn instance command. Output fields are listed in the approximate order in which they appear.

Table 83: show evpn p2mp Output Fields

Field Name	Field Description
Instance	Names of the routing instances.
VLAN	<p>VLAN identifier.</p> <p>The following information is displayed for each VLAN when the extensive option is used.</p> <ul style="list-style-type: none"> • Multicast Nexthop—The Inclusive Multicast (IM) nexthop that is used to send BUM traffic for a VLAN. • P2MP Tunnel—The name of the P2MP tunnel. • PMSI—The provider multicast service interface attributes as advertised in EVPN IM route. This is described in RFC 7432. • ESI—Ethernet segment identifier (ESI) value of the interfaces. For each ESI, the SH nexthop identifier that is used.
Neighbor Address	<p>IP address of the EVPN neighbor.</p> <p>The following information is displayed for each neighbor when the extensive option is used.</p> <ul style="list-style-type: none"> • Transport Labels—The LDP/RSVP transport label for the P2MP LSP rooted at this neighbor • PMSI—The provider multicast service interface attributes as advertised in EVPN IM route. This is described in RFC 7432.
Multicast Nexthop	Multicast nexthop identifier. This is the Inclusive Multicast (IM) nexthop that is used to send BUM traffic for a VLAN
E-tree Leaf Nexthop	E-tree leaf nexthop identifier.
Remote PE	IP address of the remote provider edge router.
Transport Labels	The LDP/RSVP transport label for the P2MP LSP for the remote PE.

Sample Output

show evpn p2mp

```
user@host> show evpn p2mp
regress@PE1_re> show evpn p2mp
Instance: ALPHA
  VLAN      Multicast Nexthop  E-tree Leaf Nexthop
  100       1048576
  Remote PE      VLAN      Transport Labels
  10.255.0.2     100       300448
  10.255.0.3     100       300560
  10.255.0.4     100       300608
```

show evpn p2mp extensive

```
user@host> show evpn p2mp extensive
Instance: ALPHA
  VLAN ID: 100
    Multicast Nexthop: 1048576
      P2MP Tunnel: 10.255.0.1:100:ldp-p2mp:evpn:100:ALPHA
      PMSI: Flags 0x0: Label 0: LDP-P2MP: Root 10.255.0.1 , lsp-id 16777217
      ESI: 00:11:22:33:44:55:66:77:88:99 SH Nexthop: 1048577
    Neighbor Address: 10.255.0.2, VLAN ID: 100
      Transport Labels: 300448
        PMSI: Flags 0x0: Label 0: LDP-P2MP: Root 10.255.0.2 , lsp-id 16777217
    Neighbor Address: 10.255.0.3, VLAN ID: 100
      Transport Labels: 300560
        PMSI: Flags 0x0: Label 0: LDP-P2MP: Root 10.255.0.3 , lsp-id 16777217
    Neighbor Address: 10.255.0.4, VLAN ID: 100
      Transport Labels: 300608
        PMSI: Flags 0x0: Label 0: LDP-P2MP: Root 10.255.0.4 , lsp-id 16777217
```

Release Information

Command introduced in Junos OS Release 18.2.

show evpn peer-gateway-macs

IN THIS SECTION

- [Syntax | 1900](#)
- [Description | 1900](#)
- [Options | 1900](#)
- [Required Privilege Level | 1900](#)
- [Release Information | 1901](#)

Syntax

```
show evpn peer-gateway-macs
<address>
<instance instance-name>
```

Description

Show Ethernet VPN (EVPN) peer gateway MAC information.

Options

<code>none</code>	Display brief information about the EVPN peer gateway MAC.
<code>address</code>	(Optional) Display peer gateway information for the specified MAC address.
<code>instance <i>instance-name</i></code>	(Optional) Display peer gateway MAC information for the specified routing instance.

Required Privilege Level

view

Release Information

Command introduced in Junos OS Release 14.2.

RELATED DOCUMENTATION

[Example: Configuring an EVPN with IRB Solution on EX9200 Switches](#) | 995

show evpn prefix

IN THIS SECTION

- [Syntax](#) | 1901
- [Description](#) | 1901
- [Required Privilege Level](#) | 1901
- [Output Fields](#) | 1902
- [Sample Output](#) | 1902
- [Release Information](#) | 1902

Syntax

```
show evpn prefix
```

Description

Display Ethernet VPN (EVPN) information for imported and exported prefixes.

Required Privilege Level

view

Output Fields

The table 2 lists the output fields for the `show evpn ip-prefix-database` command. Output fields are listed in the approximate order in which they appear.

Table 84: show evpn prefix Output Fields

Field Name	Field Description
VLAN	VLAN identifier of the Layer 2 circuit.
VNI	VXLAN network identifier for the Layer 3 virtual and routing forwarding (VRF) for the customer or tenant domain.
Prefix	MAC address associated with the IP prefix.
Active Source	The IP address or the ESI value or the IRB interface name of the source.
Timestamp	Month, day and time of generated command output.

Sample Output

show evpn prefixes

```
user@host > show evpn prefixes
```

```
Instance: ALPHA
```

VLAN	VNI	Prefix	Active source	Timestamp
	9100	10.0.0.0/24	10.255.0.2	May 22 17:16:12
	9100	20.0.0.0/24	00:11:22:33:44:55:66:77:88:99	May 22 22:33:35
	9100	30.0.0.0/24	irb.0	May 22 17:13:31
	9100	40.0.0.0/24	10.255.0.3	May 22 17:14:20

Release Information

Command introduced in Junos OS Release 17.1.

show evpn vpws-instance

IN THIS SECTION

- [Syntax | 1903](#)
- [Description | 1903](#)
- [Options | 1903](#)
- [Required Privilege Level | 1903](#)
- [Output Fields | 1903](#)
- [Sample Output | 1908](#)
- [Release Information | 1911](#)

Syntax

```
show evpn vpws-instance  
evpn-vpws-instance-name
```

Description

Display the details of the VPWS instance of the EVPN.

Options

evpn-vpws-instance-name (Optional) Name of the VPWS instance of the EVPN.

Required Privilege Level

view

Output Fields

[Table 85 on page 1904](#) describes the output fields for the `show evpn vpws-instance` command. Output fields are listed in the approximate order in which they appear.

Table 85: show evpn vpws-instance Output Fields

Field Name	Field Description
Instance	Name of the routing instance.
Route Distinguisher	Route distinguisher as configured for the routing instance on the node.
Number of local interfaces	Number of interfaces configured to be part of the routing instance.
Interface name	Name of the interface.
ESI	Ethernet segment identifier (ESI) configured for the interface.
Mode	Single-homed, single-active, or all-active form of multihoming.
Role	<p>Primary or backup, depending on the role. The role, depending on the mode, is as follows:</p> <ul style="list-style-type: none"> • Single-active multihoming–Role is primary if the interface is the designated forwarder(DF) and backup if the interface is the non-DF. • All-active multihoming–Role is always primary. • Single-homed–Role is always primary.
Status	Status of the interface is either Up or Down. Interface is available if the status is Up and interface is not available if the interface is Down.
Control-Word	Whether the device has negotiated to use the control word for MPLS pseudowire traffic (Yes) or not (No, the default).
Flow-Label-Tx	Flow-aware transport of pseudowires (FAT) flow labels is enabled (Yes) or is not enabled (No, the default) in the routing instance to push flow labels in pseudowire packets sent to a remote provider edge (PE) router.
Flow-Label-Rx	Flow-aware transport of pseudowires (FAT) flow labels is enabled (Yes) or is not enabled (No, the default) in the routing instance to pop flow labels from pseudowire packets from a remote PE router.

Table 85: show evpn vpws-instance Output Fields (Continued)

Field Name	Field Description
Local SID	Local service identifier configured on the interface.
Advertised Label	Label advertised for the service identifier as part of the autodiscovery route per the EVPN instance (EVI).
PE routers hosting the local service identifier as configured on the interface	
PE addr	IP address of the PE router advertising the autodiscovered route per the EVI.
ESI	ESI encoded in the autodiscovered route per the EVI.
Label	Label as encoded in the autodiscovered route per the EVI.
Mode	Single-homed, single-active, or all-active form of multihoming.
Role	<p>Primary or backup depending on the role. The role depending on the mode is as follows:</p> <ul style="list-style-type: none"> • Single-active multihoming–Role is primary if the interface is designated forwarder(DF) and backup if the interface is non-DF. • All-active multihoming–Role is always primary. • Single-homed–Role is always primary.
TS	Timestamp of the receipt of the autodiscovered route per the EVI. This is used as tie breaker in case there are two PE routers advertising the autodiscovered route per the EVI with the P bit set.

Table 85: show evpn vpws-instance Output Fields (*Continued*)

Field Name	Field Description
Status	<p>The status can be as follows:</p> <ul style="list-style-type: none"> Resolved—The autodiscovered route per the ESI was received for the ESI from the PE router. Unresolved—The autodiscovered route per the ESI was not received for the ESI from the PE router.
PE routers hosting the remote service identifier as configured on the interface	
Remote SID	Remote service identifier configured on the interface.
PE addr	IP address of the PE router advertising the autodiscovered route per the EVI.
ESI	ESI encoded in the autodiscovered route per the EVI.
Label	Label as encoded in the autodiscovery route per the EVI.
Mode	Single-homed, single-active, or all-active form of multihoming.
Role	<p>Primary or backup depending on the role. The role depending on the mode is as follows:</p> <ul style="list-style-type: none"> Single-active multihoming—Role is primary if the interface is designated forwarder(DF) and backup if the interface is non-DF. All-active multihoming—Role is always primary. Single-homed—Role is always primary.
TS	Timestamp of the receipt of the autodiscovered route per the EVI. This is used as tie breaker in case there are two PE routers advertising the autodiscovered route per the EVI with the P bit set.

Table 85: show evpn vpws-instance Output Fields (Continued)

Field Name	Field Description
Status	<p>The status can be as follows:</p> <ul style="list-style-type: none"> Resolved—The autodiscovered route per the ESI was received for the ESI from the PE router. Unresolved—The autodiscovered route per the ESI was not received for the ESI from the PE router.
Fast Convergence Information	Details of the ESI, the PE router, and the advertised service identifier used during fast convergence.
ESI	ESI as advertised in the autodiscovered route per the ESI.
Number of PE nodes	Number of PE nodes available.
PE	IP address of the PE router advertising the autodiscovered route per the ESI.
Advertised SID	List of service identifiers that get impacted if the PE router withdraws the autodiscovered route per the ESI.
DF Election Information for Single-Active ESI	Details of the DF election information for the single-active ESI.
ESI	ESI as advertised in the Ethernet segment route.
SID used for DF Election	Minimum service identifier configured for the ESI in the EVI.
ESI granularity	Type: Per ESI.
LACP OOS on NDF	Enabled or Disabled
Primary PE	PE router that is the DF for the ESI.

Table 85: show evpn vpws-instance Output Fields (Continued)

Field Name	Field Description
Backup PE	PE router that is the backup DF for the ESI.
Last DF Election	Last time when the DF election algorithm was executed on the PE router.

Sample Output**show evpn vpws-instance *instance-name***

```
user@host> show evpn vpws-instance vpws1004
```

```
Instance: vpws1004
```

```
Route Distinguisher: 100.100.100.4:1004
```

```
Number of local interfaces: 1 (1 up)
```

Interface name	ESI	Mode	Role	Status
ge-0/0/1.1004	00:00:00:00:00:00:00:00:00:00	single-homed	Primary	Up
Local SID: 1004 Advertised Label: 301360				
Remote SID: 2004				
PE addr	ESI	Label	Mode	Role
TS	Status			
100.100.100.2	00:11:11:11:11:11:11:11:11:11	301888	all-active	Primary
2017-01-11 21:57:31.916	Resolved			
100.100.100.1	00:11:11:11:11:11:11:11:11:11	301952	all-active	Primary
2017-01-11 21:59:28.491	Resolved			
100.100.100.3	00:11:11:11:11:11:11:11:11:11	301984	all-active	Primary
2017-01-11 21:59:28.522	Resolved			

Fast Convergence Information

```
ESI: 00:11:11:11:11:11:11:11:11:11 Number of PE nodes: 3
```

```
PE: 100.100.100.2
```

```
Advertised SID: 2004
```

```
PE: 100.100.100.1
```

```
Advertised SID: 2004
```

```

PE: 100.100.100.3
  Advertised SID: 2004

```

show evpn vpws-instance *instance-name*

```
user@host>show evpn vpws-instance vpws1003
```

```
Instance: vpws1003
```

```
Route Distinguisher: 100.100.100.4:1003
```

```
Number of local interfaces: 2 (2 up)
```

Interface name	ESI	Mode	Role	Status
ae10.2003	00:44:44:44:44:44:44:44:44	all-active	Primary	Up
Local SID: 2003 Advertised Label: 301312				
PE addr	ESI	Label	Mode	Role
TS	Status			
100.100.100.3	00:44:44:44:44:44:44:44:44	301792	all-active	Primary
2017-01-11 21:59:28.498 Resolved				
Remote SID: 1003				
Local Interface: ge-0/0/1.1003 (Up)				

Interface name	ESI	Mode	Role	Status
ge-0/0/1.1003	00:00:00:00:00:00:00:00:00	single-homed	Primary	Up
Local SID: 1003 Advertised Label: 301296				
Remote SID: 2003				
Local Interface: ae10.2003 (Up)				
PE addr	ESI	Label	Mode	Role
TS	Status			
100.100.100.3	00:44:44:44:44:44:44:44:44	301792	all-active	Primary
2017-01-11 21:59:28.498 Resolved				

```
Fast Convergence Information
```

```
ESI: 00:44:44:44:44:44:44:44:44 Number of PE nodes: 1
```

```
PE: 100.100.100.3
```

```
Advertised SID: 2003
```

show evpn vpws-instance *instance-name*(Per ESI)

```
user@host>show evpn vpws-instance vpws1001
```

```
DF Election Information for Single-Active ESI
```

```
ESI: 00:11:11:11:11:11:11:11:11
```



```

DF Election Algorithm: MOD based
SID used for DF Election: 1001
ESI granularity: Per ESI
LACP OOS on NDF: Enabled
Primary PE: 10.10.10.2
Backup PE: 10.10.10.1
Last DF Election: 2020-04-16 16:04:32

```

show evpn vpws-instance *instance-name* (with FAT flow label load balancing enabled)

```

user@host> show evpn vpws-instance
Instance: VPWS-MH-SA, Instance type: EVPN VPWS
Route Distinguisher: 10.255.0.1:200
Number of local interfaces: 1 (1 up)

Interface name  ESI                               Mode      Role      Status  Control-Word
Flow-Label-Tx  Flow-Label-Rx
ge-0/0/2.200    00:00:00:00:00:00:00:22:22:22 single-active Primary  Up       No
Yes             Yes
Local SID: 102 Advertised Label: 299808
PE addr        ESI                               Label  Mode      Role
TS              Status
10.255.0.2     00:00:00:00:00:00:00:22:22:22 299808 single-active Backup   2021-03-08
17:19:02.107 Resolved
Remote SID: 202
PE addr        ESI                               Label  Mode      Role
TS              Status
10.255.0.3     00:00:00:00:00:00:00:00:00:00 299808 single-homed Primary  2021-03-08
17:17:53.121 Resolved
Number of protect interfaces: 0

Fast Convergence Information
ESI: 00:00:00:00:00:00:00:22:22:22 Number of PE nodes: 1
PE: 10.255.0.2
Advertised SID: 102

DF Election Information for Single-Active ESI
ESI: 00:00:00:00:00:00:00:22:22:22
DF Election Algorithm: MOD based
SID used for DF Election: 102
Primary PE: 10.255.0.1

```

Backup PE: 10.255.0.2
Last DF Election: 2021-03-08 17:19:02

Release Information

Statement introduced in Junos OS Release 17.1.

RELATED DOCUMENTATION

[Overview of VPWS with EVPN Signaling Mechanisms](#) | 1046

show loop-detect enhanced interface

IN THIS SECTION

- [Syntax](#) | 1911
- [Description](#) | 1911
- [Options](#) | 1912
- [Required Privilege Level](#) | 1912
- [Output Fields](#) | 1912
- [Sample Output](#) | 1914
- [Release Information](#) | 1916

Syntax

```
show loop-detect enhanced interface  
<interface-name>
```

Description

Display status information about leaf device interfaces with loop detection enabled in an EVPN-VXLAN fabric.

In an EVPN-VXLAN fabric, you can configure loop detection on leaf devices for server-facing Layer 2 interfaces, and specify an action that the leaf device performs on the interface when it detects a loop, such as bringing the interface down. You can also configure an interval after which the device will automatically revert the interface to its state prior to the loop detection action following repair of the loop condition.

After you've enabled loop detection, the interfaces periodically send multicast loop-detection protocol data units (PDUs). If the interface receives a loop-detection PDU, which indicates a loop exists, the interface triggers the configured action to break the loop. The network administrator might then need to repair the conditions that caused the loop before reverting the interface to its state prior to loop detection. This command shows the configured loop detection parameters, current interface status, and detail about the remote interface that formed the loop.

Options

- interface-name* Display loop detection status for the specified interface.

If you don't specify an interface name, this command displays loop detection status for all interfaces that have this feature enabled.

Required Privilege Level

view

Output Fields

Table 86: show loop-detect enhanced interface Output Fields

Field Name	Field Description
Interface	Local interface name with loop detection configured.
Vlan-id	Local interface VLAN ID for which loop detection is configured.
ESI	Ethernet Segment ID (ESI) associated with the local interface.

Table 86: show loop-detect enhanced interface Output Fields *(Continued)*

Field Name	Field Description
Current status	<p>Current status of the interface—Loop-detected or Normal [Link Up] (no loop detected)</p> <p>This output section displays the following additional fields if the interface detected a loop. These fields describe the interface that routed loop detection PDUs back to the local interface:</p> <ul style="list-style-type: none"> • Remote Host: Remote host name. • Remote Chassis: Remote host chassis ID. • Remote Interface: Remote interface name that is looping PDUs. • Remote ESI: ESI associated with the remote interface. <p>NOTE: The interface causing the loop might be on the same local device or on a remote device.</p>
Last loop-detect time	Timestamp when the device initially detected the loop and changed the interface status to Loop-detected.
Receive statistics	Number of loop-detection PDUs received since the current loop detection period timestamp.
Action configured	The configured action that the device triggers upon loop detection.
Action count	Cumulative count of loop detection actions triggered on the local interface.
Transmit Interval	Default or configured interval at which the interface sends loop detection PDUs.
Revert Interval	Configured interval after the loop condition is repaired in which the interface state reverts to the previous state before the loop detection action occurred.

Sample Output

show loop-detect enhanced interface *interface-name* (no loop detected)

```
user@host> show loop-detect enhanced interface xe-0/0/12.0
```

```
Interface           :xe-0/0/12.0
Vlan-id             :100
ESI                 :00:00:00:00:00:00:00:00:00:00
Current status      :Normal[Link Up]
Last loop-detect time :-
Receive statistics   :0
Action configured    :Interface-down
Action count         :0
Transmit Interval    :1s
Revert Interval      :60s
```

show loop-detect enhanced interface *interface-name* (loop detected)

```
user@host> show loop-detect enhanced interface xe-0/0/12.0
```

```
Interface           :xe-0/0/12.0
Vlan-id             :100
ESI                 :00:00:00:00:00:00:00:00:00:00
Current status      :Loop-detected
                    Remote Host      :leaf04
                    Remote Chassis    :ab:cd:ef:12:34:56
                    Remote Interface  :xe-0/0/12.0
                    Remote ESI        :00:00:00:00:00:00:00:00:00:00
Last loop-detect time :Tue Oct 20 04:36:37 2020
Receive statistics   :4
Action configured    :Interface-down
Action count         :1
Transmit Interval    :1s
Revert Interval      :60s
```

show loop-detect enhanced interface (EX4300-48MP, all interfaces configured with loop detection)

```
user@host> show loop-detect enhanced interface
```

Interface	:ae60.0
Vlan-id	:104
ESI	:00:00:00:00:e1:06:06:06:06:00
Current status	:Loop-detected
Remote Host	:host-A
Remote Chassis	:aa:bb:cc:cc:ee:fa
Remote Interface	:ae65.0
Remote ESI	:00:00:00:00:e1:06:06:06:06:00
Last loop-detect time	:Wed Dec 23 23:20:26 2020
Receive statistics	:1624
Action configured	:Interface-down
Action count	:1
Transmit Interval	:1s
Revert Interval	:220s
Interface	:mge-0/0/31.0
Vlan-id	:101
ESI	:00:00:00:00:00:00:00:00:00:00
Current status	:Loop-detected
Remote Host	:host-B
Remote Chassis	:aa:bb:cc:cc:ee:fb
Remote Interface	:mge-0/0/32.0
Remote ESI	:00:00:00:00:00:00:00:00:00:00
Last loop-detect time	:Wed Dec 23 23:20:25 2020
Receive statistics	:1625
Action configured	:Interface-down
Action count	:1
Transmit Interval	:1s
Revert Interval	:300s
Interface	:mge-0/0/32.0
Vlan-id	:101
ESI	:00:00:00:00:00:00:00:00:00:00
Current status	:Loop-detected
Remote Host	:host-B
Remote Chassis	:aa:bb:cc:cc:ee:fb

```

Remote Interface :mge-0/0/31.0
Remote ESI      :00:00:00:00:00:00:00:00:00:00
Last loop-detect time :Wed Dec 23 23:20:25 2020
Receive statistics   :1625
Action configured    :Interface-down
Action count         :1
Transmit Interval    :1s
Revert Interval      :270s

Interface           :mge-0/0/41.0
Vlan-id             :105
ESI                 :00:00:00:00:00:00:00:00:00:00
Current status       :Loop-detected

Remote Host         :host-A
Remote Chassis      :aa:bb:cc:cc:ee:fa
Remote Interface    :xe-0/0/6.0
Remote ESI          :00:00:00:00:00:00:00:00:00:00
Last loop-detect time :Wed Dec 23 23:20:24 2020
Receive statistics   :1626
Action configured    :Interface-down
Action count         :1
Transmit Interval    :1s
Revert Interval      :240s

```

Release Information

Command introduced in Junos OS Release 20.4R1.

RELATED DOCUMENTATION

[loop-detect \(EVPN\)](#) | 1615

show mac-vrf forwarding flood

IN THIS SECTION

- [Syntax | 1917](#)
- [Description | 1917](#)
- [Options | 1918](#)
- [Additional Information | 1918](#)
- [Required Privilege Level | 1919](#)
- [Output Fields | 1919](#)
- [Sample Output | 1920](#)
- [Release Information | 1921](#)

Syntax

```
show mac-vrf forwarding flood
<brief | detail | extensive>
<event-queue>
<instance instance-name>
<route (all-ce-flood | all ve-flood | alt-root-flood | bd-flood | mlp-flood | re-flood)>
<vlan-name vlan-name>
```

Description

Show Ethernet-switching flooding information for the bridging or switching instances.

NOTE: We've implemented the show mac-vrf commands on all supported platforms. If a particular platform doesn't support a specific sub-command or option, when you run the command, the output is empty. For example, show mac-vrf forwarding age is available only on MX Series platforms. If you run that command on any other supported platform, the output is empty.

Options

none	Display all Ethernet-switching flooding information for all VLANs.
brief detail extensive	(Optional) Display the specified level of output.
event-queue	(Optional) Display the queue of pending Ethernet-switching flood events.
instance <i>instance-name</i>	(Optional) Display Ethernet-switching flooding information for the specified routing instance.
route (all-ce-flood all-ve-flood alt-root-flood bd-flood mlp-flood re-flood)	<p>(Optional) Display the route for the following traffic types::</p> <ul style="list-style-type: none"> • all-ce-flood—Display the route for flooding traffic to all customer edge routers or switches if you have configured the no-local-switching option. • all-ve-flood—Display the route for flooding traffic to all VPLS edge routers or switches if you have configured the no-local-switching option. • alt-root-flood—Display the Spanning Tree Protocol (STP) alt-root flooding route used for the interface. The alt-root flooding route is used in spanning tree protocol (STP) when the root port is in discarding state. • bd-flood—Display the route for flooding traffic of a VLAN if you have not configured the no-local-switching option. • mlp-flood—Display the route for flooding traffic to MAC learning chips.. • re-flood—Display the route that the Routing Engine (RE) uses when flooding to all interfaces.
vlan-name <i>vlan-name</i>	(Optional) Display Ethernet-switching flooding information for the specified VLAN.

Additional Information

On MX Series routers and the EX9200 line of switches, this command hierarchy is an alias for the `show bridge flood` command hierarchy.

On QFX Series switches, this command hierarchy is an alias for the `show ethernet-switching flood` command hierarchy.

Required Privilege Level

view

Output Fields

"[show mac-vrf forwarding flood](#)" on [page 1917](#) lists the output fields for the `show mac-vrf forwarding flood` command.

Table 87: show mac-vrf forwarding flood Output Fields

Field Name	Field Description
Name	Name of the switching instance
CEs	Number of customer edge routers or switches
VEs	Number of VPLS edge routers or switches
VLAN Name	Name of the associated VLAN in the switching instance (if any)
Flood Routes	Routes that get flooded to other destinations
Prefix	Prefix of the flooded route
Owner	Owner of the flood route. The displayed values could be one of: <ul style="list-style-type: none">• __all_ces__• __all_ves__• __alt_root__• __bd_flood__• __mlp_flood__• __re_flood__
Type	Type of routes flooded

Table 87: show mac-vrf forwarding flood Output Fields (*Continued*)

Field Name	Field Description
NhType	Type of next-hop receiver. The displayed values could be one of: <ul style="list-style-type: none"> • comp • ucast
NhIndex	Index number of next hop entry

Sample Output

show mac-vrf forwarding flood

```
user@host> show mac-vrf forwarding flood
```

```
Name: __juniper_private1__
```

```
CEs: 0
```

```
VEs: 0
```

```
Name: default-switch
```

```
CEs: 9
```

```
VEs: 0
```

```
VLAN Name: VLAN101
```

```
Flood Routes:
```

Prefix	Type	Owner	NhType	NhIndex
0x3057b/51	FLOOD_GRP_COMP_NH	__all_ces__	comp	12866
0x30004/51	FLOOD_GRP_COMP_NH	__re_flood__	comp	12863

```
VLAN Name: VLAN102
```

```
Flood Routes:
```

Prefix	Type	Owner	NhType	NhIndex
0x3057c/51	FLOOD_GRP_COMP_NH	__all_ces__	comp	12875
0x30005/51	FLOOD_GRP_COMP_NH	__re_flood__	comp	12872

```
VLAN Name: VLAN103
```

```
Flood Routes:
```

Prefix	Type	Owner	NhType	NhIndex
--------	------	-------	--------	---------

0x3057d/51	FLOOD_GRP_COMP_NH __all_ces__	comp	12884
0x30006/51	FLOOD_GRP_COMP_NH __re_flood__	comp	12881

Release Information

Command introduced in Junos OS Release 20.4R1.

RELATED DOCUMENTATION

[show mac-vrf forwarding flood-group](#) | [1921](#)

show mac-vrf forwarding flood-group

IN THIS SECTION

- [Syntax](#) | [1921](#)
- [Description](#) | [1922](#)
- [Options](#) | [1922](#)
- [Additional Information](#) | [1922](#)
- [Required Privilege Level](#) | [1922](#)
- [Output Fields](#) | [1922](#)
- [Sample Output](#) | [1923](#)
- [Release Information](#) | [1925](#)

Syntax

```
show mac-vrf forwarding flood-group
<eamnh>
<instance instance name>
```

Description

Display flood-group information for all active MAC-VRF instances.

NOTE: We've implemented the show mac-vrf commands on all supported platforms. If a particular platform doesn't support a specific sub-command or option, when you run the command, the output is empty. For example, show mac-vrf forwarding age is available only on MX Series platforms. If you run that command on any other supported platform, the output is empty.

Options

none Display flood-group information for the MAC-VRF instance.

eamnh (Optional)

instance <instance-name> | bd-index <bridge-domain-index-id> (Optional) Display flood-group information for the specified routing instance or for a specific bridge domain index ID.

Additional Information

On MX Series routers, this command is an alias for the show l2 learning flood-group command.
On QFX Series switches, this command is an alias for the show ethernet-switching flood-group command.

Required Privilege Level

view

Output Fields

Table 88: show mac-vrf forwarding flood-group Output Fields

Field Name	Field Description
Bd Name	Bridge domain name
Eamnh	Enhanced mesh group and next hop information
Fgrp Flag	Flood group flag

Table 88: show mac-vrf forwarding flood-group Output Fields (*Continued*)

Field Name	Field Description
Flags	Number of flags
FldGrp Name	Flood group name
Flood Route	Route identification in hexadecimal
Flood Token	Token identification in hexadecimal
Fnh id	Forwarding next-hop id.
id	EAMNH id
MeshGrpName	Name of the mesh group

Sample Output

show mac-vrf forwarding flood-group

```
user@host> show mac-vrf forwarding flood-group
```

```
FldGrp Name : __ves__+++2 MeshGrp Name: __ves__ Bd Name: v150+150 Fgrp Flag: 0xc
Eamnh: 0x94aae88 id: 1801 Flags: 0
Flood Route : 0x94aae08 Flood Token : 0x30002/51 Fnh id : 1803

FldGrp Name : __all_ces__+++2 MeshGrp Name: __all_ces__ Bd Name: v150+150 Fgrp Flag: 0xa
Eamnh: 0x94aaf08 id: 1781 Flags: 0
Flood Route : 0x94aaec8 Flood Token : 0x30003/51 Fnh id : 1783

FldGrp Name : __mlp_flood__+++2 MeshGrp Name: __mlp_flood__ Bd Name: v150+150 Fgrp Flag: 0x40c
Eamnh: 0x956f348 id: 0 Flags: 1
Flood Route : 0x94aaf48 Flood Token : 0x20002/51 Fnh id : 0

FldGrp Name : __re_flood__+++2 MeshGrp Name: __re_flood__ Bd Name: v150+150 Fgrp Flag: 0x4c
Eamnh: 0x956f608 id: 0 Flags: 1
Flood Route : 0x956f5c8 Flood Token : 0x30000/51 Fnh id : 1788
```

```
FldGrp Name : __ves__+++3 MeshGrp Name: __ves__ Bd Name: v250+250 Fgrp Flag: 0xc
Eamnh: 0x956f808 id: 1800 Flags: 0
Flood Route : 0x956f7c8 Flood Token : 0x30004/51 Fnh id : 1802

FldGrp Name : __all_ces__+++3 MeshGrp Name: __all_ces__ Bd Name: v250+250 Fgrp Flag: 0xa
Eamnh: 0x956f888 id: 1782 Flags: 0
Flood Route : 0x956f848 Flood Token : 0x30005/51 Fnh id : 1789

FldGrp Name : __mlp_flood__+++3 MeshGrp Name: __mlp_flood__ Bd Name: v250+250 Fgrp Flag: 0x40c
Eamnh: 0x956f908 id: 0 Flags: 1
Flood Route : 0x956f8c8 Flood Token : 0x20003/51 Fnh id : 0

FldGrp Name : __re_flood__+++3 MeshGrp Name: __re_flood__ Bd Name: v250+250 Fgrp Flag: 0x4c
Eamnh: 0x956f988 id: 0 Flags: 1
Flood Route : 0x956f948 Flood Token : 0x30001/51 Fnh id : 1790
```

```
user@host> show mac-vrf forwarding flood-group eamnh
```

```
Routing Instance Name : macvrf_v150
Bridge Domain Name : v150+150
Mesh Group Name : __ves__
NH Index : 1801
IF NH tree:

Component      Comp      Index      Gen      NH/Ucst      Flood
Name           Type              Number     INDEX      Ready
vtep.32771     L2_IFF      576        180        1793        True
Routing Instance Name : macvrf_v150
Bridge Domain Name : v150+150
Mesh Group Name : __all_ces__
NH Index : 1781
IF NH tree:

Component      Comp      Index      Gen      NH/Ucst      Flood
Name           Type              Number     INDEX      Ready
ae0.0          L2_IFF      575        176        1787        True
Routing Instance Name : macvrf_v150
Bridge Domain Name : v150+150
Mesh Group Name : __mlp_flood__
```

```

NH Index    : 0
Routing Instance Name : macvrf_v150
Bridge Domain Name   : v150+150
Mesh Group Name    : __re_flood__
NH Index    : 0
Routing Instance Name : macvrf_v150
Bridge Domain Name   : v250+250
Mesh Group Name    : __ves__
NH Index    : 1800
IF NH tree:

```

Component Name	Comp Type	Index	Gen Number	NH/Ucst INDEX	Flood Ready
vtep.32771	L2_IFF	576	180	1793	True

```

Routing Instance Name : macvrf_v150
Bridge Domain Name   : v250+250
Mesh Group Name    : __all_ces__
NH Index    : 1782
IF NH tree:

```

Component Name	Comp Type	Index	Gen Number	NH/Ucst INDEX	Flood Ready
ae0.0	L2_IFF	575	176	1787	True

```

Routing Instance Name : macvrf_v150
Bridge Domain Name   : v250+250
Mesh Group Name    : __mlp_flood__
NH Index    : 0
Routing Instance Name : macvrf_v150
Bridge Domain Name   : v250+250
Mesh Group Name    : __re_flood__
NH Index    : 0

```

Release Information

Command introduced in Junos OS Release 20.4R1.

RELATED DOCUMENTATION

[show mac-vrf forwarding flood | 1917](#)

[clear mac-vrf forwarding mac-table | 1739](#)

show mac-vrf forwarding global-information

IN THIS SECTION

- [Syntax | 1926](#)
- [Description | 1926](#)
- [Options | 1926](#)
- [Additional Information | 1927](#)
- [Required Privilege Level | 1927](#)
- [Output Fields | 1927](#)
- [Sample Output | 1928](#)
- [Release Information | 1928](#)

Syntax

```
show mac-vrf forwarding global-information
```

Description

Display global process and configuration information about the MAC-VRF subsystems.

NOTE: We've implemented the show mac-vrf commands on all supported platforms. If a particular platform doesn't support a specific sub-command or option, when you run the command, the output is empty. For example, show mac-vrf forwarding age is available only on MX Series platforms. If you run that command on any other supported platform, the output is empty.

Options

This command has no options.

Additional Information

On MX Series routers and the EX9200 line of switches, this command is an alias for the `show l2-learning global-information` command.

On QFX Series switches, this command is an alias for the `show ethernet-switching global-information` command.

Required Privilege Level

view

Output Fields

Table 89: show mac-vrf forwarding global-information Output Fields

Field Name	Field Description
MAC aging interval	How fast learned MAC address entries age out of the table (in seconds)
MAC learning	Allow MAC learning—enabled or disabled
MAC statistics	Keep MAC statistics—enabled or disabled
MAC limit count	Limit MAC entries to this number
MAC limit hit	Warn when MAC limit is hit—enabled or disabled
MAC packet action drop	Drop MAC packets—enabled or disabled
MAC+IP aging interval	Time to keep old MAC+IP entries—in seconds. One timer for IPv4 entries and one timer for IPv6 entries.
MAC+IP limit count	Total MAC+IP addresses supported on the platform.
MAC+IP limit reached	Has your system reached the MAC+IP limit?—Yes or No
LE aging time	Time to keep LE entries.
LE VLAN aging time	Time to keep LE VLAN entries
RE state	State of the Routing Engine—primary or backup

Table 89: show mac-vrf forwarding global-information Output Fields (*Continued*)

Field Name	Field Description
VXLAN Overlay load balance	VXLAN overlay load balancing—enabled or disabled
VXLAN ECMP	VXLAN ECMP—enabled or disabled

Sample Output

show mac-vrf forwarding global-information

```
user@host> show mac-vrf forwarding global-information
```

Global Configuration:

```
MAC aging interval      : 300
MAC learning           : Enabled
MAC statistics          : Disabled
MAC limit Count         : 393215
MAC limit hit           : Disabled
MAC packet action drop : Disabled
MAC+IP aging interval  : IPv4 - 1200 seconds
                       : IPv6 - 1200 seconds
MAC+IP limit Count      : 393215
MAC+IP limit reached    : No
LE aging time           : 1200
LE VLAN aging time      : 1200
RE state                : Master
VXLAN Overlay load bal : Disabled
VXLAN ECMP              : Disabled
```

Release Information

Command introduced in Junos OS Release 20.4R1.

RELATED DOCUMENTATION

[show mac-vrf forwarding global-mac-count | 1929](#)

[show mac-vrf forwarding global-mac-ip-count | 1931](#)

show mac-vrf forwarding global-mac-count

IN THIS SECTION

- [Syntax | 1929](#)
- [Description | 1929](#)
- [Options | 1930](#)
- [Additional Information | 1930](#)
- [Required Privilege Level | 1930](#)
- [Sample Output | 1930](#)
- [Release Information | 1930](#)

Syntax

```
show mac-vrf forwarding global-mac-count
```

Description

Display the number of globally-learned Layer 2 MAC addresses.

NOTE: We've implemented the show mac-vrf commands on all supported platforms. If a particular platform doesn't support a specific sub-command or option, when you run the command, the output is empty. For example, show mac-vrf forwarding age is available only on MX Series platforms. If you run that command on any other supported platform, the output is empty.

Options

No options exist for this command.

Additional Information

On MX Series routers, this command is an alias for `show l2-learning global-mac-count`.

On QFX Series switches, this command is an alias for `show ethernet-switching global-mac-count`.

Required Privilege Level

view

Sample Output

show mac-vrf forwarding global-mac-count

```
user@host> show mac-vrf forwarding global-mac-count
```

```
1032 dynamic and static MAC addresses learned globally
```

Release Information

Command introduced in Junos OS Release 20.4R1.

RELATED DOCUMENTATION

[show mac-vrf forwarding flood | 1917](#)

[show mac-vrf forwarding global-information | 1926](#)

[show mac-vrf forwarding global-mac-ip-count | 1931](#)

show mac-vrf forwarding global-mac-ip-count

IN THIS SECTION

- [Syntax | 1931](#)
- [Description | 1931](#)
- [Options | 1931](#)
- [Additional Information | 1931](#)
- [Required Privilege Level | 1931](#)
- [Sample Output | 1932](#)
- [Release Information | 1932](#)

Syntax

```
show mac-vrf forwarding global-mac-ip-count
```

Description

Display the number of globally-learned MAC address to IP address binding count. The first column of output shows the number of MAC addresses that the device can learn. The second column shows the number of IP addresses that are bound to learned MAC addresses.

Options

No options exist for this command.

Additional Information

On MX Series routers, this command is an alias for `show l2-learning global-mac-ip-count`.

On QFX Series switches, this command is an alias for `show ethernet-switching global-mac-ip-count`.

Required Privilege Level

view

Sample Output

show mac-vrf forwarding global-mac-ip-count

```
user@r1_dc1_spine1> show mac-vrf forwarding global-mac-ip-count
```

```
735 MAC+IP bindings learned globally
```

Release Information

Command introduced in Junos OS Release 20.4R1.

RELATED DOCUMENTATION

[show mac-vrf forwarding global-information | 1926](#)

[show mac-vrf forwarding global-mac-count | 1929](#)

show mac-vrf forwarding instance

IN THIS SECTION

- [Syntax | 1933](#)
- [Description | 1933](#)
- [Options | 1933](#)
- [Additional Information | 1933](#)
- [Required Privilege Level | 1933](#)
- [Output Fields | 1934](#)
- [Sample Output | 1934](#)
- [Release Information | 1938](#)

Syntax

```
show mac-vrf forwarding instance
<brief|detail|extensive>
<vlan-name vlan name>
```

Description

Display the MAC-IP addresses for all IPv4 (ARP) and IPv6 (ND) bindings for VLANs in routing instances where the instance type is MAC-VRF.

NOTE: We've implemented the show mac-vrf commands on all supported platforms. If a particular platform doesn't support a specific sub-command or option, when you run the command, the output is empty. For example, show mac-vrf forwarding age is available only on MX Series platforms. If you run that command on any other supported platform, the output is empty.

Options

- | | |
|-----------------------------------|--|
| none | Display the brief listing of MAC-IP addresses for all IPv4 (ARP) and IPv6 (ND) bindings for VLANs in routing instances where the instance type is MAC-VRF. |
| brief detail extensive | (Optional) Display the specified level of output. |
| vlan-name <i>vlan-name</i> | (Optional) Display information about the specified VLAN only. |

Additional Information

On MX Series routers and the EX9200 line of switches, this command hierarchy is an alias for the show 12-learning instance command hierarchy.

On QFX Series switches, this command hierarchy is an alias for the show ethernet-switching instance command hierarchy.

Required Privilege Level

view

Output Fields

Table 90: show mac-vrf forwarding instance Output Fields

Field Name	Field Description
Instance type	Routing instance type
Logical system	Name of the logical system associated with the routing instance
Routing instance	Name of the routing instance
VLAN name	Name of the associated VLAN
Index	Routing instance index number
IRB index	Integrated routing and bridging (IRB) index number (if any)
Flags	Flags for the routing instance: <div> <div>DL</div> <div>disable learning</div> <div>SE</div> <div>statistics enabled</div> <div>AD</div> <div>packet action drop</div> <div>LH</div> <div>MAC limit hit</div> <div>MI</div> <div>MAC+IP limit hit</div> </div>
Tag	VLAN tags (if any)

Sample Output

show mac-vrf forwarding instance

```

user@host> show mac-vrf forwarding instance
Information for routing instance and VLAN:

Flags (DL - disable learning, SE - stats enabled,
      AD - packet action drop, LH - MAC limit hit,
      MI - mac+ip limit hit)

Inst Logical   Routing      VLAN name      Index IRB   Flags Tag

```

type	system	instance	index		
RTT	Default	__juniper_private1__	1		
RTT	Default	default-switch	4		
RTT	Default	default_internal_6	5		
RTT	Default	macvrf_v150	6		
vlan	Default	macvrf_v150	2	553	150
vlan	Default	macvrf_v150	3	554	250

show mac-vrf forwarding instance detail

user@host **show mac-vrf forwarding instance detail**Information for routing instance and VLAN:

Routing instance : __juniper_private1__

RTB index: 1

MAC limit: 5000 MACs learned: 0 Local Macs learned: 0

Static MACs learned: 0 Non config static MACs learned: 0

MAC+IP limit: 0 MAC+IP bindings learned: 0 Local MAC+IP learned:0

Sequence number: 0 Handle: 0x931a008

VLAN id:

Flags: Bridge instance,RTT is present in all PFE.

Config service id: 0 Active service id: 0

Config VLAN Id : Config operation: none

Config params: mac tbl sz: 0, mac age: 300000, intf mac limit: 8192,

Config MAC+IP params: mac+ip tbl sz: 0, intf mac+ip limit: 8192,

Config static MAC count :

Counters:

Kernel write errors : 0

Information for routing instance and VLAN:

Routing instance : default-switch

RTB index: 4

MAC limit: 0 MACs learned: 0 Local Macs learned: 0

Static MACs learned: 0 Non config static MACs learned: 0

MAC+IP limit: 0 MAC+IP bindings learned: 0 Local MAC+IP learned:0

Sequence number: 0 Handle: 0x9317008

VLAN id:

Flags: Bridge instance,RTT defined,Default RTT,RTT is present in all PFE.

Config service id: 0 Active service id: 0

Config VLAN Id : Config operation: none

Config params: mac tbl sz: 0, mac age: 300000, intf mac limit: 8192,

Config MAC+IP params: mac+ip tbl sz: 0, intf mac+ip limit: 8192,

Config static MAC count :

Counters:

Kernel write errors : 0

Information for routing instance and VLAN:

Routing instance : default_internal_6

RTB index: 5

MAC limit: 5000 MACs learned: 0 Local Macs learned: 0

Static MACs learned: 0 Non config static MACs learned: 0

MAC+IP limit: 0 MAC+IP bindings learned: 0 Local MAC+IP learned: 0

Sequence number: 0 Handle: 0x9570008

VLAN id:

Flags: Bridge instance, RTT defined, RTT is present in all PFE.

Config service id: 0 Active service id: 0

Config VLAN Id : Config operation: none

Config params: mac tbl sz: 0, mac age: 300000, intf mac limit: 8192,

Config MAC+IP params: mac+ip tbl sz: 0, intf mac+ip limit: 8192,

Config flags: mac-vrf fwd-inst

Config static MAC count :

Counters:

Kernel write errors : 0

Information for routing instance and VLAN:

Routing instance : macvrf_v150

RTB index: 6

MAC limit: 0 MACs learned: 6 Local Macs learned: 0

Static MACs learned: 0 Non config static MACs learned: 0

MAC+IP limit: 0 MAC+IP bindings learned: 7 Local MAC+IP learned: 10

Sequence number: 0 Handle: 0x956d008

VLAN id:

Flags: Bridge instance, RTT defined, vpls RTT, RTT is present in all PFE., EVPN VXLAN

Forwarding instance: default_internal_6 (Index: 5)

Config service id: 0 Active service id: 0

Config VLAN Id : Config operation: none

Config params: mac tbl sz: 0, mac age: 300000, intf mac limit: 8192,

Config MAC+IP params: mac+ip tbl sz: 0, intf mac+ip limit: 8192,

Config flags: vpls, irb ifl, svtep ifl, evpn vxlan, mac-vrf

Service type: vlan-aware

Config static MAC count :

Counters:

Kernel write errors : 0

Information for routing instance and VLAN:

Routing instance : macvrf_v150

```

VLAN Name : v150
  RTB index: 6                VLAN index: 2
  MAC limit: 5120             MACs learned: 3      Local Macs learned: 0
  MAC+IP limit: 5120          MAC+IP bindings learned: 3
  Sequence number: 0          Handle: 0x957e008
  VLAN id: 150
  Flags: BD defined,VxLAN,Ageless CPMAC,Recd MAC SYNC,Sent MAC SYNC,ARP/NDP flood suppression
  Config VLAN Id      : 150      Config operation: none
  Config params: mac tbl sz: 5120, mac age: 300000, intf mac limit: 1024,
  Config MAC+IP params: mac+ip tbl sz: 5120, intf mac+ip limit: 1024,
  Config flags: vlan,VxLAN,Ageless CPMAC,no st mac cfg
  Config static MAC count : 0
  Config ownership flags: config
  Config RG Id: 0             Active RG Id: 0
  Config service id: 0         Active service id: 0
  Config VXLAN Id: 150        Active VXLAN Id: 150
  Config VXLAN MCIP: 0.0.0.0   Active VXLAN MCIP: 0.0.0.0
  Config VXLAN Decapsulate accept inner vlan: 0
  Config VXLAN Encapsulate inner vlan      : 0
  Config VXLAN Unreachable RVTEP age timer : 300
  Config VXLAN OVSDB Enabled: No
  SVTEP L3-RTT ID: 0           Source-VTEP IP: 10.255.1.1
  Kernel ownership flags: config
  MVRP ref count: 0
Counters:
  Kernel write errors      : 0

```

Information for routing instance and VLAN:

Routing instance : macvrf_v150

```

VLAN Name : v250
  RTB index: 6                VLAN index: 3
  MAC limit: 5120             MACs learned: 3      Local Macs learned: 0
  MAC+IP limit: 5120          MAC+IP bindings learned: 4
  Sequence number: 0          Handle: 0x9580008
  VLAN id: 250
  Flags: BD defined,VxLAN,Ageless CPMAC,Recd MAC SYNC,Sent MAC SYNC,ARP/NDP flood suppression
  Config VLAN Id      : 250      Config operation: none
  Config params: mac tbl sz: 5120, mac age: 300000, intf mac limit: 1024,
  Config MAC+IP params: mac+ip tbl sz: 5120, intf mac+ip limit: 1024,
  Config flags: vlan,VxLAN,Ageless CPMAC,no st mac cfg
  Config static MAC count : 0
  Config ownership flags: config
  Config RG Id: 0             Active RG Id: 0

```

```

Config service id: 0          Active service id: 0
Config VXLAN   Id: 250          Active VXLAN   Id: 250
Config VXLAN MCIP: 0.0.0.0      Active VXLAN MCIP: 0.0.0.0
Config VXLAN Decapsulate accept inner vlan: 0
Config VXLAN Encapsulate inner vlan      : 0
Config VXLAN Unreachable RVTEP age timer : 300
Config VXLAN OVSDB Enabled: No
SVTEP L3-RTT ID:  0            Source-VTEP IP: 10.255.1.1
Kernel ownership flags: config
MVRP ref count: 0
Counters:
Kernel write errors      : 0

```

Release Information

Command introduced in Junos OS Release 20.4R1.

RELATED DOCUMENTATION

[show mac-vrf forwarding global-information](#) | [1926](#)

show mac-vrf forwarding instance-mapping

IN THIS SECTION

- [Syntax](#) | [1939](#)
- [Description](#) | [1939](#)
- [Options](#) | [1939](#)
- [Required Privilege Level](#) | [1939](#)
- [Output Fields](#) | [1939](#)
- [Sample Output](#) | [1940](#)
- [Release Information](#) | [1940](#)

Syntax

```
show mac-vrf forwarding instance-mapping
<brief|detail>
<forwarding-instance-id forwarding instance id number>
```

Description

Display information about the mapping of configured MAC-VRF instances to forwarding instances.

Options

- none** Display brief information about the mapping of the global MAC-VRF instance to the forwarding instance.
- brief | detail** (Optional) Display the specified level of detail.
- forwarding-instance-id forwarding instance id number** (Optional) Display the instances mapped to a specific forwarding instance id.

Required Privilege Level

view

Output Fields

Table 91: show mac-vrf forwarding instance-mapping Output Fields

Field Name	Field Description
Forwarding-instance-id	Configuration ID for the MAC-VRF routing instance—Id 0 is reserved for the default-switch instance. The output displays other ID numbers only if you've configured the IDs using the set routing-instances <i>mac-vrf-instance-name</i> forwarding-instance identifier <i>id-number</i> command.
Forwarding-instance	Forwarding-instance name—The instance name can be either default-switch (if you have not configured a forwarding-instance identifier) or the name of the configured forwarding instance.

Table 91: show mac-vrf forwarding instance-mapping Output Fields *(Continued)*

Field Name	Field Description
Table-id	Table ID for the forwarding instance—Unique, kernel-allocated table ID for the forwarding instance.
Mac-vrf instance	Name of the MAC-VRF instance that is mapped to the forwarding instance.
Table-id	Unique, kernel-allocated table ID for the MAC-VRF instance.

Sample Output

show mac-vrf forwarding instance-mapping

```
user@host> show mac-vrf forwarding instance-mapping
```

Forwarding-instance-id	Forwarding instance	Table-id	Mac-vrf instance	Table-id
0	default-switch	6	macvrf_v500	13
6	default_internal_6	7	macvrf_v100	8
			macvrf_v200	9
60	default_internal_60	10	macvrf_v300	11
	vs400	14		

Release Information

Command introduced in Junos OS Release 20.4R1.

RELATED DOCUMENTATION

[show mac-vrf forwarding instance](#) | 1932

[show mac-vrf forwarding mac-table](#) | [1951](#)

show mac-vrf forwarding interface

IN THIS SECTION

- [Syntax](#) | [1941](#)
- [Description](#) | [1941](#)
- [Options](#) | [1941](#)
- [Additional Information](#) | [1942](#)
- [Required Privilege Level](#) | [1942](#)
- [Output Fields](#) | [1942](#)
- [Sample Output](#) | [1943](#)
- [Release Information](#) | [1945](#)

Syntax

```
show mac-vrf forwarding interface  
<brief | detail | extensive>  
<interface-name>
```

Description

Show Ethernet switching information about all the interfaces within all active MAC-VRF routing instances.

Options

none Show brief Ethernet switching information about all the interfaces within all MAC-VRF routing instances.

brief | detail | extensive (Optional) Show the specified level of output.

`interface-name` (Optional) Show the Ethernet switching information for the specified interface only.

NOTE: We've implemented the `show mac-vrf` commands on all supported platforms. If a particular platform doesn't support a specific sub-command or option, when you run the command, the output is empty. For example, `show mac-vrf forwarding age` is available only on MX Series platforms. If you run that command on any other supported platform, the output is empty.

Additional Information

On MX Series routers and the EX9200 line of switches, this command hierarchy is an alias for the `show l2-learning instance` command hierarchy.

On QFX Series switches, this command hierarchy is an alias for the `show ethernet-switching interface` command hierarchy.

Required Privilege Level

view

Output Fields

Table 92: show mac-vrf forwarding interface Output Fields

Field Name	Field Description
Routing instance name	Name of the mac-vrf routing instance.
Logical interface	Logical interface name associated with the mac-vrf routing instance
VLAN members	For aggregated Ethernet (ae) interfaces, this lists the member VLANs for the displayed interface.
TAG	VLAN tag associated with VLAN member interfaces.
MAC limit	Maximum number of MAC address entries allowed for each interface. The MAC limit for each member interface must add up to less than the total MAC limit for the aggregated Ethernet interface.

Table 92: show mac-vrf forwarding interface Output Fields *(Continued)*

Field Name	Field Description
MAC+IP limit	Maximum number of MAC+IP address entries allowed for each interface. The MAC+IP limit for each member interface must add up to less than the total MAC+IP limit for the aggregated Ethernet interface.
STP state	The spanning tree protocol state for the displayed VLAN member interfaces.
Logical interface flags	DL disable learning AD packet action drop LH MAC limit hit DN interface down MMAS MAC move action shutdown AS Autostate-exclude enabled SCTL shutdown by storm control MI MAC+IP limit hit
Tagging	Display shows tagged, or blank depending on whether the interface is tagged or not.

Sample Output

show mac-vrf forwarding interface

```
user@host> show mac-vrf forwarding interface
```

```
Routing Instance Name : macvrf_v150
Logical Interface flags (DL - disable learning, AD - packet action drop,
                        LH - MAC limit hit, DN - interface down,
                        MMAS - Mac-move action shutdown, AS - Autostate-exclude enabled,
                        SCTL - shutdown by Storm-control, MI - MAC+IP limit hit)

Logical      Vlan      TAG  MAC   MAC+IP  STP      Logical      Tagging
interface    members  limit limit state   interface flags
vtep.32770   0        0      0      0
Routing Instance Name : macvrf_v150
```

Logical Interface flags (DL - disable learning, AD - packet action drop,
 LH - MAC limit hit, DN - interface down,
 MMAS - Mac-move action shutdown, AS - Autostate-exclude enabled,
 SCTL - shutdown by Storm-control, MI - MAC+IP limit hit)

Logical interface	Vlan members	TAG	MAC limit	MAC+IP limit	STP state	Logical interface flags	Tagging
vtep.32769			0	0			tagged

Routing Instance Name : macvrf_v150

Logical Interface flags (DL - disable learning, AD - packet action drop,
 LH - MAC limit hit, DN - interface down,
 MMAS - Mac-move action shutdown, AS - Autostate-exclude enabled,
 SCTL - shutdown by Storm-control, MI - MAC+IP limit hit)

Logical interface	Vlan members	TAG	MAC limit	MAC+IP limit	STP state	Logical interface flags	Tagging
ae0.0			8192	8192			tagged
	v150	150	1024	1024	Forwarding		tagged
	v250	250	1024	1024	Forwarding		tagged

Routing Instance Name : macvrf_v150

Logical Interface flags (DL - disable learning, AD - packet action drop,
 LH - MAC limit hit, DN - interface down,
 MMAS - Mac-move action shutdown, AS - Autostate-exclude enabled,
 SCTL - shutdown by Storm-control, MI - MAC+IP limit hit)

Logical interface	Vlan members	TAG	MAC limit	MAC+IP limit	STP state	Logical interface flags	Tagging
vtep.32771			0	0			tagged
	v250	250	0	0	Forwarding		tagged
	v150	150	0	0	Forwarding		tagged

Routing Instance Name : macvrf_v150

Logical Interface flags (DL - disable learning, AD - packet action drop,
 LH - MAC limit hit, DN - interface down,
 MMAS - Mac-move action shutdown, AS - Autostate-exclude enabled,
 SCTL - shutdown by Storm-control, MI - MAC+IP limit hit)

Logical interface	Vlan members	TAG	MAC limit	MAC+IP limit	STP state	Logical interface flags	Tagging
esi.1786			0	0			tagged
	v150	150	0	0	Forwarding		tagged

Routing Instance Name : macvrf_v150

Logical Interface flags (DL - disable learning, AD - packet action drop,
 LH - MAC limit hit, DN - interface down,

```
MMAS - Mac-move action shutdown, AS - Autostate-exclude enabled,
SCTL - shutdown by Storm-control, MI - MAC+IP limit hit)

Logical      Vlan      TAG  MAC  MAC+IP STP      Logical      Tagging
interface    members                                interface flags
esi.1794                                0      0                                tagged
                                v250  250  0      0      Forwarding                                tagged
Routing Instance Name : macvrf_v150
Logical Interface flags (DL - disable learning, AD - packet action drop,
                        LH - MAC limit hit, DN - interface down,
                        MMAS - Mac-move action shutdown, AS - Autostate-exclude enabled,
                        SCTL - shutdown by Storm-control, MI - MAC+IP limit hit)

Logical      Vlan      TAG  MAC  MAC+IP STP      Logical      Tagging
interface    members                                interface flags
esi.1804                                0      0                                tagged
Routing Instance Name : macvrf_v150
Logical Interface flags (DL - disable learning, AD - packet action drop,
                        LH - MAC limit hit, DN - interface down,
                        MMAS - Mac-move action shutdown, AS - Autostate-exclude enabled,
                        SCTL - shutdown by Storm-control, MI - MAC+IP limit hit)

Logical      Vlan      TAG  MAC  MAC+IP STP      Logical      Tagging
interface    members                                interface flags
esi.1805                                0      0                                tagged
Paste device command output here
```

Release Information

Command introduced in Junos OS Release 20.4R1.

RELATED DOCUMENTATION

- [show mac-vrf forwarding instance | 1932](#)
- [show ethernet-switching mac-ip-table | 1748](#)

show mac-vrf forwarding mac-ip-table

IN THIS SECTION

- [Syntax | 1946](#)
- [Description | 1946](#)
- [Options | 1947](#)
- [Additional Information | 1947](#)
- [Required Privilege Level | 1947](#)
- [Output Fields | 1947](#)
- [Sample Output | 1949](#)
- [Release Information | 1951](#)

Syntax

```
show mac-vrf forwarding mac-ip-table  
<brief | count | detail | extensive>  
<age>  
<instance>  
<ipv4>  
<ipv6>  
<kernel >  
<vlan-name (all | vlan-name)>
```

Description

Display the MAC-IP address for all IPv4 (ARP) and IPv6 (ND) bindings for VLANs in MAC-VRF type routing instances.

NOTE: We've implemented the show mac-vrf commands on all supported platforms. If a particular platform doesn't support a specific subcommand or option, when you run the command, the output is empty. For example, show mac-vrf forwarding age is available only on MX Series platforms. If you run that command on any other supported platform, the output is empty.

Options

none	Display brief information about the MAC-IP table.
brief count detail extensive	(Optional) Display the specified level of output.
age	(Optional) (For MX Series routers and the EX9200 line of switches only) Display the age of a single MAC address.
instance <i>instance-name</i>	(Optional) Display MAC-IP table information for the specified routing instance.
ipv4	(Optional) Display IPv4 address entries associated with MAC addresses learned by ARP.
ipv6	(Optional) Display IPv6 address entries associated with MAC addresses learned by NDP.
vlan-name (all <i>vlan name</i>)	(Optional) Display MAC-IP table information for the specified VLAN.

Additional Information

On MX Series routers and the EX9200 line of switches, this command hierarchy is an alias for the `show bridge mac-ip-table` command hierarchy.

On QFX Series switches, this command hierarchy is an alias for the `show ethernet-switching mac-ip-table` command hierarchy.

Required Privilege Level

view

Output Fields

Table 1 lists the output fields for the `show mac-vrf forwarding mac-ip table` command.

Table 93: show mac-vrf forwarding mac-ip-table Output Fields

Field Name	Field Description	Level of Output
IP address	IP address associated with the EVPN routing instance.	All levels

Table 93: show mac-vrf forwarding mac-ip-table Output Fields (Continued)

Field Name	Field Description	Level of Output
MAC address	MAC address	All levels
Flags	<p>MAC IP flags—Identifies statically installed MAC-IP entries.</p> <ul style="list-style-type: none"> • S—Statically installed MAC-IP address entries. • D—Dynamic installed MAC-IP address entries. • L—Locally learned MAC-IP address entries • R— MAC-IP address entries that are learned from a remote device through the control plane. • K—MAC-IP address entries that are installed in the kernel. • RT—Destination route that corresponds to an installed entry. • AD—Advertised to remote device. • RE—ARP/ND entry has expired and another ARP or network solicitation request has been sent. • R0—Router • OV—Override 	Brief, detail
Logical Interface	The logical interface associated with the routing instance.	Brief, detail

Table 93: show mac-vrf forwarding mac-ip-table Output Fields (Continued)

Field Name	Field Description	Level of Output
Active source	The source of the learned MAC-IP address entry. Displays the ESI or router ID.	All levels
Routing instance	Information about the routing instance where the MAC-IP address entry was learned. <ul style="list-style-type: none"> • Bridging domain • MAC-IP local mask • MAC-IP remote mask • MAC-IP RPD flags • MAC-IP flags 	Extensive
BD ID	Bridge domain ID	Brief

Sample Output

show mac-vrf forwarding mac-ip-table vlan-name v100

```
user@host> show mac-vrf forwarding mac-ip-table vlan-name v100
```

MAC IP flags (S - Static, D - Dynamic, L - Local , R - Remote, K - Kernel, RT - Dest Route,
AD - Advt to remote, RE - Re-ARP/ND, RO - Router, OV - Override)

Routing instance : default-

Bridging domain : v100

IP address	MAC address	Flags	Logical Interface	Active source
10.1.1.1	00:11:00:00:00:01	DR,K,RT	vtep.32769	
101.1.1.1				
2001::192:100:1:1	00:00:10:00:11:00	DR,K,RT	vtep.32769	

101.1.1.1	fe80::205:8600:6471:e600	00:00:10:00:11:00	DR,K,RT	vtep.32769
101.1.1.1	10.0.1.2	00:00:10:00:22:00	S,K	
	2001::192:100:1:2	00:00:10:00:22:00	S,K	
	fe80::205:8600:6471:d900	00:00:10:00:22:00	S,K	
	10.0.1.3	00:00:10:00:33:00	DR,K,RT	vtep.32771
103.1.1.1	2001::192:100:1:3	00:00:10:00:33:00	DR,K,RT	vtep.32771
103.1.1.1	fe80::205:8600:6471:7400	00:00:10:00:33:00	DR,K,RT	vtep.32771
103.1.1.1	10.0.1.4	00:00:10:00:44:00	DR,K,RT	vtep.32770
104.1.1.1	2001::192:100:1:4	00:00:10:00:44:00	DR,K,RT	vtep.32770
104.1.1.1	fe80::205:8600:6471:c000	00:00:10:00:44:00	DR,K,RT	vtep.32770
104.1.1.1	10.0.1.254	40:00:10:11:11:00	S,K	esi.1874
05:00:00:00:64:00:00:00:64:00	2001::192:100:1:254	60:00:10:11:11:00	S,K	esi.1874
05:00:00:00:64:00:00:00:64:00	Paste device command output here			

show mac-vrf forwarding mac-ip-table extensive

IP address: 10.1.1.1
MAC address: 00:11:00:00:00:01
Routing instance: default
Bridging domain: v100
MAC-IP local mask: 0x0000000000000000
MAC-IP remote mask: 0x8000000000000000
MAC-IP RPD flags: 0x00000081
MAC-IP flags: remote, kernel

show mac-vrf forwarding mac-ip-table kernel

show mac-vrf forwarding mac-ip-table kernel				
BD	IP	MAC	Logical	Flags

id	address	address	Interface	
2	192.168.1.1	00:21:59:aa:77:f0	irb.1	0x00000209
2	192.168.1.100	00:00:5e:00:01:01	irb.1	0x00000609
3	192.168.2.1	00:21:59:aa:77:f0	irb.2	0x00000209
3	192.168.2.100	00:00:5e:00:01:01	irb.2	0x00000609
4	192.168.3.1	00:21:59:aa:77:f0	irb.3	0x00000209
4	192.168.3.100	00:00:5e:00:01:01	irb.3	0x00000609
5	192.168.4.1	00:21:59:aa:77:f0	irb.4	0x00000209
5	192.168.4.100	00:00:5e:00:01:01	irb.4	0x00000609

Release Information

Command introduced in Junos OS Release 20.4R1.

show mac-vrf forwarding mac-table

IN THIS SECTION

- [Syntax | 1951](#)
- [Description | 1952](#)
- [Options | 1952](#)
- [Additional Information | 1952](#)
- [Required Privilege Level | 1953](#)
- [Output Fields | 1953](#)
- [Sample Output | 1954](#)
- [Release Information | 1955](#)

Syntax

```
show mac-vrf forwarding mac-table
<brief | count | detail | extensive>
<mac address>
<age>
<instance vrf-instance-name>
```

```
<vlan-name vlan-name>
<vlan-id vlan-id>
<interface interface-name>
```

Description

Display information about the forwarding table in the mac-vrf instance. The commands work only within a mac-vrf type routing instance that is present on a supported platform.

NOTE: We've implemented the show mac-vrf commands on all supported platforms. If a particular platform doesn't support a specific sub-command or option, when you run the command, the output is empty. For example, show mac-vrf forwarding age is available only on MX Series platforms. If you run that command on any other supported platform, the output is empty.

Options

none	(Same as brief) Display information about the mac-vrf forwarding MAC table.
brief count detail extensive	(Optional) Display the specified level of output.
age	(Optional) For MX Series routers only. Display the age of a single MAC address.
instance <i>instance-name</i>	(Optional) Display MAC table information for the specified routing instance.
interface <i>interface-name</i>	(Optional) Display MAC table information for the specified interface.
vlan-name <i>vlan name</i>	(Optional) Display MAC table information for the specified VLAN.
vlan-id <i>vlan id</i>	(Optional) Display MAC table information for the specified VLAN.

Additional Information

On MX Series routers and the EX9200 line of switches, this command hierarchy is an alias for the show bridge mac-table command hierarchy. All the command options available in this command hierarchy are aliases to the respective command options in the show bridge mac-table command hierarchy.

On QFX Series switches, this command hierarchy is an alias for the `show ethernet-switching table` command hierarchy. All the command options available in this command hierarchy are aliases to the respective command options in the `show ethernet-switching table` command hierarchy.

Required Privilege Level

view

Output Fields

Table 94 on page 1953 lists the output fields for the `show mac-vrf forwarding mac-table` command.

Table 94: show mac-vrf forwarding mac-table Output Fields

Field Name	Description
VLAN name	Name of the associated VLAN.
MAC address	Learned MAC address
MAC flags	<p>Flags that provide information about the MAC addresses in the MAC-VRF instance.</p> <ul style="list-style-type: none">• S—Statically installed MAC-IP address entries.• D—Dynamic installed MAC-IP address entries.• L—Locally learned MAC-IP address entries• R— MAC-IP address entries that are learned from a remote device through the control plane.• K—Kernel-based MAC-IP address entries.• RT—Destination route that corresponds to an installed entry.• AD—Advertised to remote device.• RE—ARP/ND entry has expired and another ARP or network solicitation request has been sent.• R0—Router• OV—Override

Table 94: show mac-vrf forwarding mac-table Output Fields *(Continued)*

Field Name	Description
Logical interface	Interface on which the MAC address was learned.
SVLBNH/VENH index	Shared VXLAN load balancing next hop (SVLBNH)/ VXLAN encapsulated next hop (VENH) associated with the SVLBNH.
Active source	IP address or MAC address of the source interface.
Routing instance	Name of the routing instance with which this MAC table is associated.

Sample Output

show mac-vrf forwarding mac-table

```
root@vQFX-Leaf1> show mac-vrf forwarding mac-table
```

MAC flags (S - static MAC, D - dynamic MAC, L - locally learned, P - Persistent static
SE - statistics enabled, NM - non configured MAC, R - remote PE MAC, O - ovsdb MAC)

Ethernet switching table : 6 entries, 6 learned

Routing instance : macvrf_v150

Vlan name	MAC address	MAC flags	Logical interface	SVLBNH/ VENH Index	Active source
v150 10.255.1.2	02:05:86:71:bd:00	DR	vtep.32771		
v150 10.255.1.1	02:05:86:71:d1:00	DR	vtep.32769		
v150 05:00:00:fc:4d:00:00:00:96:00	40:00:10:11:11:00	DR	esi.1787		
v150 05:00:00:fc:4d:00:00:00:96:00	40:00:10:11:11:02	DR	esi.1787		
v150 05:00:00:fc:4d:00:00:00:96:00	50:00:10:11:11:00	DR	esi.1787		

```
v150          50:00:10:11:11:02  DR      esi.1787
05:00:00:fc:4d:00:00:00:96:00
```

Release Information

Command introduced in Junos OS Release 20.4R1.

show mac-vrf forwarding mgrp-policy

IN THIS SECTION

- [Syntax | 1955](#)
- [Description | 1955](#)
- [Options | 1956](#)
- [Additional Information | 1956](#)
- [Required Privilege Level | 1957](#)
- [Output Fields | 1957](#)
- [Sample Output | 1958](#)
- [Release Information | 1961](#)

Syntax

```
show mac-vrf forwarding mgrp-policy
<instance instance-name>
<instance instance-name mgrp-id mgrp-id>
```

Description

Display information about logical mesh groups, which are part of a mechanism that Juniper Networks uses to control the flooding of broadcast, unknown unicast, and multicast (BUM) traffic.

You can assign all interfaces that you configure on a spine or super spine device to a particular mesh group. We support the following default mesh groups for a VLAN:

- CE mesh group, to which you assign all directly connected interfaces.
- VE mesh group, to which you assign all virtual tunnel endpoint (VTEP) interfaces for the local POD or data center.
- WAN mesh group, to which you assign all remote VTEP interfaces within standalone or interconnected PODs or data centers.
- RE mesh group, to which you typically do not assign interfaces.

When a BUM packet that must be flooded enters an interface on the spine or super spine device, the system identifies the interface's mesh group. The mesh group associated with a particular VLAN determines the outgoing groups to which the packet should flood. For example, if a BUM packet enters an interface assigned to the CE mesh group, the outgoing flood groups might be the following:

- All interfaces in the CE mesh group except the interface through which the packet entered.
- The VE and WAN mesh groups.
- Whether the spine or super spine device that receives a packet acts as a designated forwarder (DF) or a non-DF.
- The local bias feature, also known as split horizon, which eliminates duplication of traffic to its destination in another POD or data center.

NOTE: We've implemented the `show mac-vrf` commands on all supported platforms. If a particular platform doesn't support a specific sub-command or option, when you run the command, the output is empty. For example, `show mac-vrf forwarding age` is available only on MX Series platforms. If you run that command on any other supported platform, the output is empty.

Options

none	Display the global mesh group policy.
instance <i>routing instance name</i>	(Optional) Display the mesh group policy for the specified routing instance.
mgrp-id <i>mesh group id</i>	(Optional) Display the mesh group policy for the specified mac-vrf routing instance and mesh group id number.

Additional Information

On MX Series routers and the EX92000 line of switches, this command hierarchy is an alias for the `show 12-learning mgrp-policy` command hierarchy.

On QFX Series switches, this command hierarchy is an alias for the `show ethernet-switching mgrp-policy` command hierarchy.

Required Privilege Level

admin

Output Fields

[Table 95 on page 1957](#) lists the output fields for the `show mac-vrf forwarding mgrp-policy` command.

[Table 96 on page 1957](#) lists the output fields for the `show mac-vrf forwarding mgrp-policy instance` command.

[Table 97 on page 1958](#) lists the output fields for the `show mac-vrf forwarding mgrp-policy instance mgrp-id` command.

Table 95: show mac-vrf forwarding mgrp-policy Output Fields

Field Name	Field Description
Role	<p>Roles include either DF (designated forwarder) or non-DF (non-designated forwarder).</p> <p>We organize the output by role. Mesh groups listed under each role provide a summary of the groups to which a BUM packet might get flooded.</p>
Mesh Group	Name of a mesh group. For each mesh group, the output displays the outgoing flood groups to which a BUM packet gets flooded.

Table 96: show mac-vrf forwarding mgrp-policy instance Output Fields

Field Name	Field Description
Role	Roles include either DF (designated forwarder) or non-DF (non-designated forwarder). The output displays the role for each mesh group.
Mesh Group	<p>Name of a mesh group. For each mesh group, the output displays:</p> <ul style="list-style-type: none"> Flags associated with the mesh group. The outgoing flood groups to which a BUM packet gets flooded.

Table 97: show mac-vrf forwarding mgrp-policy instance mgrp-id Output Fields

Field Name	Field Description
Mesh Group	Name of a mesh group. For the mesh group, the output displays the associated flags.
Interfaces	Interfaces associated with the mesh group. When a spine or super spine device floods a BUM packet to the next hop, it does so through these interfaces.

Sample Output

show mac-vrf forwarding mgrp-policy

```
user@host> show mac-vrf forwarding mgrp-policy

Role: DF

Mesh Group:  __all_ces__
    __all_ces__
    __ves__
    __wan_flood__
    __icl_flood__
    __etree_leaf__

Mesh Group:  __ar_flood__
    __all_ces__
    __ar_flood__
    __wan_flood__

Mesh Group:  __esi_flood__

Mesh Group:  __etree_leaf__
    __all_ces__
    __ves__

Mesh Group:  __icl_flood__
    __all_ces__
```

Mesh Group: __mlp_flood__

Mesh Group: __re_flood__

__all_ces__

__ves__

__wan_flood__

__icl_flood__

__etree_leaf__

Mesh Group: __ves__

__all_ces__

__wan_flood__

__etree_leaf__

Mesh Group: __wan_flood__

__all_ces__

__ves__

Role: NDF

Mesh Group: __all_ces__

__all_ces__

__ves__

__wan_flood__

__icl_flood__

Mesh Group: __ar_flood__

__all_ces__

__ar_flood__

__wan_flood__

Mesh Group: __esi_flood__

Mesh Group: __etree_leaf__

Mesh Group: __icl_flood__

Mesh Group: __mlp_flood__

Mesh Group: __re_flood__

__all_ces__

__ves__

```

__wan_flood__

Mesh Group:   __ves__
              __all_ces__

Mesh Group:   __wan_flood__
              __all_ces__
Paste device command output here

```

show mac-vrf forwarding mgrp-policy instance

```

user@host> show mac-vrf forwarding mgrp-policy
instance test
Role: DF
Mesh Group:   __all_ces__Flags: 0xa ( split-horizon, shared-RT)
              __all_ces__
              __ves__
              __wan_flood__

Role: DF
Mesh Group:   __mlp_flood__Flags: 0xc ( shared-RT, no-local-flooding)

Role: DF
Mesh Group:   __re_flood__Flags: 0x4c ( shared-RT, no-local-flooding, permanent-comp-nh)
              __all_ces__
              __ves__
              __wan_flood__

Role: DF
Mesh Group:   __ves__Flags: 0xc ( shared-RT, no-local-flooding)
              __all_ces__
              __wan_flood__

Role: DF
Mesh Group:   __wan_flood__Flags: 0xc ( shared-RT, no-local-flooding)
              __all_ces__
              __ves__

Role: Non-DF
Mesh Group:   __ves__Flags: 0xc ( shared-RT, no-local-flooding)

```

```

Role: Non-DF
Mesh Group:    __wan_flood__Flags: 0xc ( shared-RT, no-local-flooding)
               __all_ces__

```

show mac-vrf forwarding mgrp-policy instance mgrp-id

```

user@host> show mac-vfr forwarding mgrp-policy
instance evpn-vxlan mgrp-id 0
Mesh Group:    __ves__Flags: 0xc ( shared-RT, no-local-flooding)
Interfaces
  vtep.32769
  vtep.32770
  vtep.32771
  vtep.32772

```

Release Information

Command introduced in Junos OS Release 20.4R1.

RELATED DOCUMENTATION

[show ethernet-switching mac-ip-table | 1748](#)

[show mac-vrf forwarding flood | 1917](#)

[show mac-vrf forwarding mac-table | 1951](#)

show mac-vrf forwarding statistics

IN THIS SECTION

- [Syntax | 1962](#)
- [Description | 1962](#)
- [Options | 1962](#)
- [Additional Information | 1963](#)

- Required Privilege Level | 1963
- Output Fields | 1963
- Sample Output | 1965
- Release Information | 1965

Syntax

```
show mac-vrf statistics
<instance instance-name>
<vlan-name vlan-name>
```

Description

Display bridge statistics for the device. The statistics include a count of local MAC addresses, a count of remote MAC addresses, a count of remote VXLAN tunnel endpoints (VTEPs), and the number of enabled VLANs on the device including those in the current `mac-vrf` routing instance. The command output displays traffic statistics by packet and byte.

NOTE: `mac-statistics` must be enabled at the `edit routing-instances <instance-name> protocols evpn` hierarchy or no statistics are gathered.

NOTE: We've implemented the `show mac-vrf` commands on all supported platforms. If a particular platform doesn't support a specific sub-command or option, when you run the command, the output is empty. For example, `show mac-vrf forwarding age` is available only on MX Series platforms. If you run that command on any other supported platform, the output is empty.

Options

none	Display bridge statistics for all VLANs in all routing instances.
instance <i>instance-name</i>	(Optional) Display statistics for the specified routing instance.
vlan-name <i>vlan-name</i>	(Optional) Display statistics for the specified VLAN.

Additional Information

On supported MX Series routers, this command is an alias for the `show bridge statistics` and `show evpn statistics` commands.

On QFX Series switches, this command is an alias for the `show ethernet-switching statistics` command.

Required Privilege Level

view

Output Fields

Table 98: Output Fields

Field Name	Description
Routing instance	The displayed output is separated by routing instance. The name of the routing instance serves as the section separator.
Index	For the routing instance display: The index number associated with the routing instance. For the interface display: The index number of the local interface.
Sequence number	Sequence number assigned to this bridging domain. Used for debugging.
Non-config static MACs learned	
Bridging domain	The bridge domain associated with the specified routing instance.
MACs learned	Count of MAC addresses learned in the specified routing instance.
MAC limit	The maximum number of MAC addresses that can be associated with the specified routing instance.

Table 98: Output Fields *(Continued)*

Field Name	Description
Static MACs learned	Count of static MAC addresses learned in the specified routing instance.
Flags	Shows the type of routing instance configured.
Local interface	Shows the local interface associated with the specified routing instance.
Broadcast packets	Count of broadcast packets forwarded on the specified interface.
Broadcast bytes	Count of broadcast bytes forwarded on the specified interface
Multicast packets	Count of multicast packets forwarded on the specified interface.
Multicast bytes	Count of multicast bytes forwarded on the specified interface
Flooded packets	Count of packets flooded out on the specified interface
Flooded bytes	Count of bytes flooded out on the specified interface.
Unicast packets	Count of unicast packets forwarded on the specified interface
Unicast bytes	Count of unicast bytes forwarded on the specified interface.
Current MAC count	Count of the number of MAC addresses learned through the specified interface.

Sample Output

show mac-vrf forwarding statistics

```
user@host> show mac-vrf forwarding statistics
```

```
Local interface: vtep.32769, Index: 574
  Multicast packets:                0
  Multicast bytes   :                0
  Flooded packets   :                0
  Flooded bytes     :                0
  Current MAC count:                0
Local interface: vtep.32771, Index: 576
  Multicast packets:                0
  Multicast bytes   :                0
  Flooded packets   :                0
  Flooded bytes     :                0
  Current MAC count:                2
Local interface: vtep.32770, Index: 564
  Multicast packets:                0
  Multicast bytes   :                0
  Flooded packets   :                0
  Flooded bytes     :                0
  Current MAC count:                0
Local interface: ae0.0, Index: 575
  Multicast packets:                0
  Multicast bytes   :                0
  Flooded packets   :                0
  Flooded bytes     :                0
  Current MAC count:                0 (Limit 8192)
```

Release Information

Command introduced in Junos OS Release 20.4.

RELATED DOCUMENTATION

[show mac-vrf forwarding global-information](#) | [1926](#)

show mac-vrf forwarding vlans

IN THIS SECTION

- [Syntax](#) | [1966](#)
- [Description](#) | [1966](#)
- [Options](#) | [1967](#)
- [Additional Information](#) | [1967](#)
- [Required Privilege Level](#) | [1967](#)
- [Output Fields](#) | [1967](#)
- [Sample Output](#) | [1968](#)
- [Release Information](#) | [1969](#)

Syntax

```
show mac-vrf forwarding vlans
<brief | detail | extensive>
<instance instance-name>
<operational>
<vlan-name>
<interface interface-name>
```

Description

Display information about VLANs configured on mac-vrf routing instances.

NOTE: We've implemented the show mac-vrf commands on all supported platforms. If a particular platform doesn't support a specific sub-command or option, when you run the

command, the output is empty. For example, `show mac-vrf forwarding age` is available only on MX Series platforms. If you run that command on any other supported platform, the output is empty.

Options

none	Display brief information for all VLANs.
brief detail extensive	(Optional) Display the specified level of output.
instance <i>instance-name</i>	(Optional) Display information about the specified routing instance.
operational	(Optional) Display information about the operational routing instances.
vlan <i>vlan-name</i>	(Optional) Display information about the specified VLAN.
interface <i>interface-name</i>	(Optional) Display information about the specified logical interface.

Additional Information

On MX Series routers and the EX9200 line of switches, this command hierarchy is an alias for the `show bridge domain` command hierarchy.

On QFX Series switches, this command hierarchy is an alias for the `show vlans` command hierarchy.

Required Privilege Level

view

Output Fields

Table 99: show mac-vrf forwarding vlans Output Fields

Field Name	Field Description	Level of Output
VLAN name	Name of the VLAN.	none, brief
Tag	802.1Q tag applied to this VLAN. If none is displayed, no tag is applied.	All levels

Table 99: show mac-vrf forwarding vlans Output Fields (*Continued*)

Field Name	Field Description	Level of Output
Interfaces	Interface associated with learned MAC addresses or All-members option (flood entry). An asterisk (*) beside the interface indicates that the interface is UP .	All levels
Routing instance	Name of the routing instance on which the VLAN is configured.	All levels
Number of interfaces	Number of tagged and untagged interfaces associated with a VLAN.	detail, extensive
Internal Index	VLAN index internal to Junos OS software.	extensive
Origin	Manner in which the VLAN was created: static or learn .	extensive
Total MAC count	Total number of 802.1Q-tagged and untagged VLANs on the device.	detail, extensive
VXLAN Enabled	Shows whether VXLAN (Virtual Extensible LAN) is enabled within the mac-vrf routing instance.	detail, extensive
MAC aging time	Time, in seconds, for the MAC entry to age out.	detail, extensive

Sample Output

show mac-vrf forwarding vlans brief

```
user@router> show mac-vrf forwarding vlans brief
```

Routing instance	VLAN name	Tag	Interfaces
default-switch	c1	20	ge-0/0/0.0* ge-1/0/0.0* ge-2/0/0.0*
default-switch	c2	30	ge-0/0/0.0* ge-2/0/0.0*

default-switch	default	1	
default-switch	iso	10	ge-0/0/1.0*
default-switch	iso1	50	ge-0/0/0.0* ge-2/0/0.0*
default-switch	pri	100	ge-0/0/0.0* ge-1/0/0.0* ge-2/0/0.0*
test	v100	NA	ge-2/0/1.100*

Release Information

Command introduced in Junos OS Release 20.4R1.

show mac-vrf forwarding vxlan-tunnel-end-point esi

IN THIS SECTION

- [Syntax | 1970](#)
- [Description | 1970](#)
- [Options | 1970](#)
- [Additional Information | 1970](#)
- [Required Privilege Level | 1971](#)
- [Output Fields | 1971](#)
- [Sample Output | 1972](#)
- [Release Information | 1972](#)

Syntax

```
show mac-vrf forwarding vxlan-tunnel-end-point esi
<esi-identifier esi-identifier>
<instance instance>
<svlbnh>
```

Description

Displays the Ethernet segment identifier (ESI) information about VXLAN tunnel end-points (VTEPs).

NOTE: We've implemented the show mac-vrf commands on all supported platforms. If a particular platform doesn't support a specific sub-command or option, when you run the command, the output is empty. For example, show mac-vrf forwarding age is available only on MX Series platforms. If you run that command on any other supported platform, the output is empty.

Options

none	Display information about the VTEP.
esi-identifier <i>esi-identifier</i>	(Optional) Display information about the specified Ethernet segment identified by the ESI value.
instance <i>instance</i>	(Optional) Display ESI information about the specified routing instance.
svlbnh	<p>(Optional) Display the shared VXLAN load-balancing next hop (SVLBNH) index for a given ESI and instance.</p> <p>You must use this option in conjunction with both the esi-identifier and instance options. For detailed information about a particular SVLBNH index, see "show ethernet-switching vxlan-tunnel-end-point svlbnh" on page 1811.</p>

Additional Information

On MX Series routers and the EX9200 line of switches, this command hierarchy is an alias for the show 12-learning vxlan-tunnel-end-point command hierarchy.

On QFX Series switches, this command hierarchy is an alias for the show ethernet-switching vxlan-tunnel-end-point command hierarchy.

Required Privilege Level

view

Output Fields

Table 100: show mac-vrf forwarding vxlan-tunnel-end-point esi Output Fields

Field Name	Field Description
ESI	Ethernet segment identifier
RTT	Routing instance in which you have configured the ESI
VLBNH	VXLAN load balancing next hop.
INH	Indirect next hop
ESI-IFL	ESI logical interface
LOC-IFL	Local logical interface for a given Ethernet segment
#RVTEPS	Number of remote VTEPs
RVTEP-IP	IP address of the remote VTEP
RVTEP-IFL	Index number of the remote VTEP.
VENH	VXLAN encapsulated composite next hop
MASK-ID	Position of the remote VTEP in the ESI mask. Each remote VTEP is assigned a unique position.
FLAGS	Internal flags used for debugging. The flags help to identify the state of the remote VTEP.
MAC-COUNT	Count of MACs learned on the remote VTEP for a given ESI.
SVLBNH	Index number of the SVLBNH.

Sample Output

show mac-vrf forwarding vxlan-tunnel-end-point esi

user@host> show mac-vrf forwarding vxlan-tunnel-end-point esi

PESI	RTT		VLNBH	INH	ESI-IFL	LOC-IFL	#RVTEPs
00:10:11:12:13:14:15:16:17:11	default-switch		1758	131077	esi.1758		2
Aliasing							
RVTEP-IP	RVTEP-IFL	VENH	MASK-ID	FLAGS	MAC-COUNT		
111.1.1.1	vtep.32769	1753	0	2	0		
112.1.1.1	vtep.32773	1757	1	2	0		
ESI	RTT		VLNBH	INH	ESI-IFL	LOC-IFL	#RVTEPs
00:10:11:12:13:14:15:16:17:22	default-switch		1768	131080	esi.1768	ae0.0,	1
Aliasing							
RVTEP-IP	RVTEP-IFL	VENH	MASK-ID	FLAGS	MAC-COUNT		
113.1.1.1	vtep.32774	1767	0	2	0		
ESI	RTT		VLNBH	INH	ESI-IFL	LOC-IFL	#RVTEPs
05:00:00:00:64:00:00:00:64:00	default-switch		1762	131079	esi.1762		3
RVTEP-IP	RVTEP-IFL	VENH	MASK-ID	FLAGS	MAC-COUNT		
101.1.1.1	vtep.32770	1754	0	2	2		
103.1.1.1	vtep.32771	1755	1	2	2		
102.1.1.1	vtep.32772	1756	2	2	2		
ESI	RTT		VLNBH	INH	ESI-IFL	LOC-IFL	#RVTEPs
05:00:00:00:64:00:00:00:c8:00	default-switch		1760	131078	esi.1760		3
RVTEP-IP	RVTEP-IFL	VENH	MASK-ID	FLAGS	MAC-COUNT		
101.1.1.1	vtep.32770	1754	0	2	2		
103.1.1.1	vtep.32771	1755	1	2	2		
102.1.1.1	vtep.32772	1756	2	2	2		

Release Information

Command introduced in Junos OS Release 20.4R1.

RELATED DOCUMENTATION

| [show mac-vrf forwarding vxlan-tunnel-end-point svlnbh](#) | 1982

show mac-vrf forwarding vxlan-tunnel-end-point remote

IN THIS SECTION

- [Syntax | 1973](#)
- [Description | 1973](#)
- [Options | 1974](#)
- [Additional Information | 1974](#)
- [Required Privilege Level | 1974](#)
- [Output Fields | 1974](#)
- [Sample Output | 1980](#)
- [Release Information | 1982](#)

Syntax

```
show mac-vrf forwarding vxlan-tunnel-end-point remote
<detail>
<instance mac-vrf routing-instance name>
<ip remote vtep ip address>
<mac-table>
<summary>
<vtep-source-interface local vtep source interface ip address>
```

Description

Display information about remote VXLAN tunnel endpoints (VTEPs).

By default, this command displays VTEP information for multiple MAC-VRF routing instances, if configured.

NOTE: We've implemented the `show mac-vrf` commands on all supported platforms. If a particular platform doesn't support a specific sub-command or option, when you run the command, the

output is empty. For example, show mac-vrf forwarding age is available only on MX Series platforms. If you run that command on any other supported platform, the output is empty.

Options

none	Display information about remote VTEPs associated with all MAC-VRF routing instances.
detail	Display information about remote VTEPs associated with all MAC-VRF routing instances.
instance <i>mac-vrf routing-instance name</i>	(Optional) Display VTEP information about the specified MAC-VRF routing instance.
ip remote vtep <i>ip address</i>	(Optional) Display detailed information about the specified remote VTEP IP address.
mac-table	(Optional) Display MAC table information along with the remote VTEP information.
summary	(Optional) Display only summary information about all MAC-VRF routing instances.
vtep-source-interface <i>local vtep source interface ip address</i>	(Optional) Display detailed remote VTEP information associated with the specified local VTEP source IP address.

Additional Information

On MX Series routers and the EX9200 line of switches, this command hierarchy is an alias for the show l2-learning vxlan-tunnel-end-point remote command hierarchy.

On QFX Series switches, this command hierarchy is an alias for the show ethernet-switching vxlan-tunnel-end-point remote command hierarchy.

Required Privilege Level

view

Output Fields

[Table 1 on page 1975](#) lists the output fields for the show mac-vrf forwarding vxlan-tunnel-end-point remote command.

[Table 2 on page 1976](#) lists the output fields for the show mac-vrf forwarding vxlan-tunnel-end-point remote instance command.

[Table 3 on page 1977](#) lists the output fields for the `show mac-vrf forwarding vxlan-tunnel-end-point remote ip` command.

[Table 4 on page 1978](#) lists the output fields for the `show mac-vrf forwarding vxlan-tunnel-end-point remote mac-table` command.

[Table 5 on page 1978](#) lists the output fields for the `show mac-vrf forwarding vxlan-tunnel-end-point remote summary` command.

[Table 6 on page 1979](#) lists the output fields for the `show mac-vrf forwarding vxlan-tunnel-end-point remote vtep-source-interface` command.

Table 101: show mac-vrf forwarding vxlan-tunnel-end-point remote Output Fields

Field Name	Field Description
Logical System Name	Name of the logical system on which you have configured the MAC-VRF routing instance
Id	Logical system ID
SVTEP-IP	IP address of the (local) source VTEP
IFL	Logical interface name
L3-Idx	System index number of the associated Layer 3 interface
SVTEP-Mode	Source VTEP mode
RVTEP-IP	IP address of the remote VTEP
L2-RTT	Layer 2 routing instance name
IFL-Idx	System index number of the associated logical interface
Interface	Name of the VTEP interface
NH-Id	Next-hop ID
RVTEP-Mode	Mode of the remote VTEP interface
Flags	Flag
VNID	VLAN network ID

Table 101: show mac-vrf forwarding vxlan-tunnel-end-point remote Output Fields (*Continued*)

Field Name	Field Description
MC-Group-IP	Multicast group IP address. Defaults to 0.0.0.0 if not configured.

Table 102: show mac-vrf forwarding vxlan-tunnel-end-point remote instance *mac-vrf routing-instance name* Output Fields

Field Name	Field Description
Logical System Name	Name of the logical system on which you have configured the MAC-VRF routing instance
Id	Logical system ID
SVTEP-IP	IP address of the (local) source VTEP
IFL	Logical interface name
L3-Idx	System index number of the associated Layer 3 interface
SVTEP-Mode	Source VTEP mode
RVTEP-IP	IP address of the remote VTEP
L2-RTT	Layer 2 routing instance name
IFL-Idx	System index number of the associated logical interface
Interface	Name of the VTEP interface
NH-Id	Next-hop ID
RVTEP-Mode	Mode of the remote VTEP interface
Flags	Flag
VNID	VXLAN network ID
MC-Group-IP	Multicast group IP address. Defaults to 0.0.0.0 if not configured.

Table 103: show mac-vrf forwarding vxlan-tunnel-end-point remote ip *remote vtep ip address* Output Fields

Field Name	Field Description																
Logical System	Name of the logical system on which you have configured the MAC-VRF routing instance																
Routing instance	Name of the MAC-VRF routing instance whose tunnel connects with the remote IP address																
Bridging domain	Name of the bridging domain																
VLAN	VLAN ID of the tunnel																
VXLAN ID	VXLAN ID																
Multicast Group IP	Multicast group IP address. Defaults to 0.0.0.0 if not configured.																
Remote VTEP	IP address of the remote VTEP																
Nexthop Id	ID of the next hop																
MAC address	MAC address associated with the remote VTEP IP address																
MAC Flags	Flags: <table> <tr> <td>S</td><td>Static MAC address</td></tr> <tr> <td>D</td><td>Dynamic MAC address</td></tr> <tr> <td>L</td><td>Locally learned MAC address</td></tr> <tr> <td>C</td><td>Control MAC address</td></tr> <tr> <td>SE</td><td>Statistics enabled</td></tr> <tr> <td>NM</td><td>Non-configured MAC address</td></tr> <tr> <td>R</td><td>Remote PE MAC address</td></tr> <tr> <td>P</td><td>Pinned MAC address</td></tr> </table>	S	Static MAC address	D	Dynamic MAC address	L	Locally learned MAC address	C	Control MAC address	SE	Statistics enabled	NM	Non-configured MAC address	R	Remote PE MAC address	P	Pinned MAC address
S	Static MAC address																
D	Dynamic MAC address																
L	Locally learned MAC address																
C	Control MAC address																
SE	Statistics enabled																
NM	Non-configured MAC address																
R	Remote PE MAC address																
P	Pinned MAC address																
Logical interface	Logical interface associated with the remote MAC address and the remote IP address																
Remote VTEP IP address	IP address of the remote VTEP																

Table 104: show mac-vrf forwarding vxlan-tunnel-end-point remote mac-table Output Fields

Field Name	Field Description																
Logical System	Name of the logical system on which you have configured the MAC-VRF routing instance																
Routing instance	Name of the MAC-VRF routing instance whose tunnel connects with the remote IP address																
Bridging domain	Name of the bridging domain																
VLAN	VLAN ID of the tunnel																
VNID	VXLAN network ID																
MAC address	MAC address associated with the remote VTEP IP address																
MAC Flags	Flags: <table> <tr> <td>S</td><td>Static MAC address</td></tr> <tr> <td>D</td><td>Dynamic MAC address</td></tr> <tr> <td>L</td><td>Locally learned MAC address</td></tr> <tr> <td>C</td><td>Control MAC address</td></tr> <tr> <td>SE</td><td>Statistics enabled</td></tr> <tr> <td>NM</td><td>Non-configured MAC address</td></tr> <tr> <td>R</td><td>Remote PE MAC address</td></tr> <tr> <td>P</td><td>Pinned MAC address</td></tr> </table>	S	Static MAC address	D	Dynamic MAC address	L	Locally learned MAC address	C	Control MAC address	SE	Statistics enabled	NM	Non-configured MAC address	R	Remote PE MAC address	P	Pinned MAC address
S	Static MAC address																
D	Dynamic MAC address																
L	Locally learned MAC address																
C	Control MAC address																
SE	Statistics enabled																
NM	Non-configured MAC address																
R	Remote PE MAC address																
P	Pinned MAC address																
Logical interface	Logical interface associated with the remote MAC address and the remote IP address																
Remote VTEP IP address	IP address of the remote VTEP																

Table 105: show mac-vrf forwarding vxlan-tunnel-end-point remote summary Output Fields

Field Name	Field Description
Logical System Name	Name of the logical system on which you have configured the MAC-VRF routing instance
Id	Logical system ID

Table 105: show mac-vrf forwarding vxlan-tunnel-end-point remote summary Output Fields
(Continued)

Field Name	Field Description
SVTEP-IP	IP address of the (local) source VTEP
IFL	Logical interface name
L3-Idx	System index number of the associated Layer 3 interface
SVTEP-Mode	Source VTEP mode
RVTEP-IP	IP address of the remote VTEP
L2-RTT	Layer 2 routing instance name
IFL-Idx	System index number of the associated logical interface
Interface	Name of the VTEP interface
NH-Id	Next-hop ID
RVTEP-Mode	Mode of the remote VTEP interface.
Flags	Flag

Table 106: show mac-vrf forwarding vxlan-tunnel-end-point remote vtep-source-interface *local vtep source ip address* Output Fields

Field Name	Field Description
Logical System Name	Name of the logical system on which you have configured the MAC-VRF routing instance
Id	Logical system ID
SVTEP-IP	IP address of the (local) source VTEP
IFL	Logical interface name
L3-Idx	System index number of the associated Layer 3 interface

Table 106: show mac-vrf forwarding vxlan-tunnel-end-point remote vtep-source-interface *local vtep source ip address* Output Fields (Continued)

Field Name	Field Description
SVTEP-Mode	Source VTEP mode
RVTEP-IP	IP address of the remote VTEP
L2-RTT	Layer 2 routing instance name
IFL-Idx	System index number of the associated logical interface
Interface	Name of the VTEP interface
NH-Id	Next-hop ID
RVTEP-Mode	Mode of the remote VTEP interface
Flags	Flag
VNID	VXLAN network ID
MC-Group-IP	Multicast group IP address. Defaults to 0.0.0.0 if not configured.

Sample Output

show mac-vrf forwarding vxlan-tunnel-end-point remote

```
user@host> show mac-vrf forwarding vxlan-tunnel-end-point remote
```

Logical System Name	Id	SVTEP-IP	IFL	L3-Idx	SVTEP-Mode			
<default>	0	10.255.1.1	lo0.0	0				
RVTEP-IP	L2-RTT		IFL-Idx	Interface	NH-Id	RVTEP-Mode	Flags	
10.255.1.2	macvrf_v150		557	vtep.32770	1793	RNVE		
VNID	MC-Group-IP							
150	0.0.0.0							
RVTEP-IP	L2-RTT		IFL-Idx	Interface	NH-Id	RVTEP-Mode	Flags	

10.255.1.3	macvrf_v150	556	vtep.32769	1791	RNVE
10.255.1.4	macvrf_v150	558	vtep.32771	1799	RNVE

show mac-vrf forwarding vxlan-tunnel-end-point remote ip *remote vtep ip address*

```
user@host> show mac-vrf forwarding vxlan-tunnel-end-point remote ip 10.255.1.2
```

MAC flags (S -static MAC, D -dynamic MAC, L -locally learned, C -Control MAC
SE -Statistics enabled, NM -Non configured MAC, R -Remote PE MAC , P -Pinned MAC)

Logical system : <default>

Routing instance : macvrf_v150

Bridging domain : v150+150, VLAN : 150

VXLAN ID : 150, Multicast Group IP : 0.0.0.0

Remote VTEP : 10.255.1.2, Nexthop ID : 1793

MAC address	MAC flags	Logical interface	Remote VTEP IP address
02:05:86:71:14:00	DR	vtep.32770	10.255.1.2
40:00:10:11:11:02	DR	esi.1792	10.255.1.2
50:00:10:11:11:02	DR	esi.1792	10.255.1.2

show mac-vrf forwarding vxlan-tunnel-end-point remote mac-table

```
user@host> show mac-vrf forwarding vxlan-tunnel-end-point remote mac-table
```

MAC flags (S -static MAC, D -dynamic MAC, L -locally learned, C -Control MAC
SE -Statistics enabled, NM -Non configured MAC, R -Remote PE MAC , P -Pinned MAC)

Logical system : <default>

Routing instance : macvrf_v150

Bridging domain : v150+150, VLAN : 150

VXLAN ID : 150, Multicast Group IP : 0.0.0.0

Remote VTEP : 10.255.1.2, Nexthop ID : 1793

MAC address	MAC flags	Logical interface	Remote VTEP IP address
----------------	--------------	----------------------	---------------------------

02:05:86:71:14:00	DR	vtep.32770	10.255.1.2
40:00:10:11:11:02	DR	esi.1792	10.255.1.2
50:00:10:11:11:02	DR	esi.1792	10.255.1.2

Release Information

Command introduced in Junos OS Release 20.4R1.

RELATED DOCUMENTATION

[show mac-vrf forwarding vxlan-tunnel-end-point svlbh | 1982](#)

[show mac-vrf forwarding vxlan-tunnel-end-point esi | 1969](#)

show mac-vrf forwarding vxlan-tunnel-end-point svlbh

IN THIS SECTION

- [Syntax | 1982](#)
- [Description | 1983](#)
- [Options | 1983](#)
- [Additional Information | 1983](#)
- [Required Privilege Level | 1983](#)
- [Output Fields | 1983](#)
- [Sample Output | 1984](#)
- [Release Information | 1985](#)

Syntax

```
show mac-vrf forwarding vxlan-tunnel-end-point svlbh
<brief | detail | extensive>
<overflow>
<svlbh-index index-number>
```

Description

Display information about a shared VXLAN load-balancing next hop (SVLBNH) and the virtual tunnel end-points (VTEPs) associated with the SVLBNH.

NOTE: We've implemented the show mac-vrf commands on all supported platforms. If a particular platform doesn't support a specific sub-command or option, when you run the command, the output is empty. For example, show mac-vrf forwarding age is available only on MX Series platforms. If you run that command on any other supported platform, the output is empty.

Options

- none** (Same as brief) Display information for all SVLBNHs.
- brief | detail | extensive** (Optional) Display the specified level of output.
- overflow** (Optional) Display the pending requests for SVLBNH allocation.
- svlbnh-index *index-number*** (Optional) Display information about the specified SVLBNH index.

Additional Information

On MX Series routers and the EX9200 line of switches, this command hierarchy is an alias for the show 12-learning vxlan-tunnel-end-point svlbnh command hierarchy.

On QFX Series switches, this command hierarchy is an alias for the show ethernet-switching vxlan-tunnel-end-point svlbnh command hierarchy.

Required Privilege Level

view

Output Fields

Field Name	Field Description
SVLBNH <i>index-number</i>	Index number of the SVLBNH.

(Continued)

Field Name	Field Description
RVTEP count	Number of remote VTEPs associated with the SVLBNH.
ESI count	Number of Ethernet segment identifiers (ESIs) associated with the SVLBNH.
RVTEP-IP	IP address of the remote VTEP.
RVTEP-IFL	Index number of remote VTEP.
VENH	VXLAN encapsulated next hop (VENH) associated with the SVLBNH.
ESI	ESIs associated with the SVLBNH.
RTT	Routing instance in which you have configured the ESI.

Sample Output

show mac-vrf forwarding vxlan-tunnel-end-point svlbnh detail

```
user@host> show mac-vrf forwarding vxlan-tunnel-end-point svlbnh detail
```

```
SVLBNH: 1882      RVTEP count: 2      ESI count: 1
RVTEP-IP          RVTEP-IFL          VENH
10.0.5.1          vtep.32770          1859
10.0.3.1          vtep.32769          1835
ESI               RTT
00:00:00:00:00:00:00:00:01:03 default-switch
```

```
SVLBNH: 1879      RVTEP count: 2      ESI count: 1
RVTEP-IP          RVTEP-IFL          VENH
10.0.2.1          vtep.32771          1869
10.0.3.1          vtep.32769          1835
```

```

ESI                                RTT
00:00:00:00:00:00:00:01:01 default-switch

SVLBNH: 1881      RVTEP count: 2      ESI count: 1
RVTEP-IP          RVTEP-IFL          VENH
10.0.2.1          vtep.32771         1869
10.0.5.1          vtep.32770         1859
ESI                                RTT
00:00:00:00:00:00:00:01:02 default-switch

SVLBNH: 1870      RVTEP count: 3      ESI count: 5
RVTEP-IP          RVTEP-IFL          VENH
10.0.2.1          vtep.32771         1869
10.0.5.1          vtep.32770         1859
10.0.3.1          vtep.32769         1835
ESI                                RTT
00:00:00:00:00:00:00:01:04 default-switch
05:00:00:02:9a:00:00:00:65:00 default-switch
05:00:00:02:9a:00:00:00:66:00 default-switch
05:00:00:02:9a:00:00:00:67:00 default-switch
05:00:00:02:9a:00:00:00:68:00 default-switch

```

Release Information

Command introduced in Junos OS Release 20.4R1.

RELATED DOCUMENTATION

[show mac-vrf forwarding vxlan-tunnel-end-point svlbnh](#) | [1982](#)

show mld snooping evpn database

IN THIS SECTION

- [Syntax](#) | [1986](#)
- [Description](#) | [1986](#)

- Options | 1986
- Required Privilege Level | 1987
- Output Fields | 1987
- Sample Output | 1988
- Release Information | 1989

Syntax

```
show mld snooping evpn database
<brief | detail>
<group>
<bridge-domain bridge-domain-id>
<instance routing-instance-name>
<interface interface-name>
<logical-system (logical-system-name | all)>
<vlan vlan-id>
```

Description

Display MLD snooping information for IPv6 multicast forwarding in an EVPN network.

Options

brief detail	Level of output to display. The default output is the same as brief.
<i>group</i>	Display output only for the specified IPv6 multicast group IP address range.
bridge-domain	Display output only for the specified bridge domain name.
instance	Display output for the specified EVPN instance (EVI).
interface	Display output only for the specified interface.
logical-system	Display output for a specific logical system name or all logical systems.
vlan	Display output only for a specified VLAN ID (0 through 4094).

Required Privilege Level

routing

Output Fields

Table 107 on page 1987 lists the output fields for the `show mld snooping evpn database` command. Output fields are listed in the approximate order in which they appear.

Table 107: show mld snooping evpn database Output Fields

Field Name	Field Description	Level of Output
Instance	Name of the EVPN routing instance for which information is displayed.	All levels
Bridge-Domain, VLAN ID	Bridge domain name and VLAN ID.	All levels
Group IP	Multicast IPv6 address of the multicast group.	All levels
Source IP	IPv6 address of the multicast data source for the group.	All levels
Access OIF Count	Number of outgoing interface (OIF) list entries on the access side.	All levels
Core NH	Next hop ID corresponding to the next hop interface.	All levels
Access OIF (more information)	More details about OIF entries: <ul style="list-style-type: none"> • Interface—Interface name • Local—Local receiver. • Remote—Remote receiver. 	detail

Sample Output

show mld snooping evpn database

```

user@device> show mld snooping evpn database
Instance: EVPN-2
  Bridge-Domain: V100, VLAN ID: 100
    Group IP: ff02::db8:25:1, Source IP: ::, Access OIF Count: 2, Core NH: 1049231
    Group IP: ff02::db8:33:1, Source IP: ::, Access OIF Count: 2, Core NH: 1049231

Instance: EVPN-4
  Bridge-Domain: V200, VLAN ID: 200
    Group IP: ff02::db8:25:1, Source IP: ::, Access OIF Count: 1, Core NH: 1049232
    Group IP: ff02::db8:33:1, Source IP: ::, Access OIF Count: 1, Core NH: 1049232

```

show mld snooping evpn database detail

```

user@device> show mld snooping evpn database
detail
Instance: EVPN-2
  Bridge-Domain: V100, VLAN ID: 100
    Group IP: ff02::db8:25:1, Source IP: ::
      Core NH: 1049231
      Access OIF Count: 2
        Interface    Local  Remote
        ae3.0        0      1
        ge-0/0/4.0    1      0
    Group IP: ff02::db8:33:1, Source IP: ::
      Core NH: 1049231
      Access OIF Count: 2
        Interface    Local  Remote
        ge-0/0/4.0    1      0
        ae3.0        1      0

Instance: EVPN-4
  Bridge-Domain: V200, VLAN ID: 200
    Group IP: ff02::db8:25:1, Source IP: ::
      Core NH: 1049232
      Access OIF Count: 1

```

```

      Interface    Local    Remote
      ae4.0        0        1
Group IP: ff02::db8:33:1, Source IP: ::
Core NH: 1049232
Access OIF Count: 1
      Interface    Local    Remote
      ae4.0        1        0

```

Release Information

Introduced in Junos OS Release 18.4R1.

Introduced in Junos OS Release 19.4R2 on ACX Series routers.

RELATED DOCUMENTATION

[Configure Multicast Forwarding with MLD Snooping in an EVPN-MPLS Environment | 1038](#)

[Overview of Multicast Forwarding with IGMP or MLD Snooping in an EVPN-MPLS Environment | 1013](#)

show route forwarding-table

IN THIS SECTION

- [Syntax | 1990](#)
- [Syntax \(MX Series Routers\) | 1990](#)
- [Syntax \(TX Matrix and TX Matrix Plus Routers\) | 1990](#)
- [Description | 1991](#)
- [Options | 1991](#)
- [Required Privilege Level | 1993](#)
- [Output Fields | 1993](#)
- [Sample Output | 1999](#)
- [Release Information | 2003](#)

Syntax

```
show route forwarding-table
<detail | extensive | summary>
<all>
<ccc interface-name>
<destination destination-prefix>
<family family | matching matching>
<interface-name interface-name>
<label name>
<matching matching>
<multicast>
<table (default | logical-system-name/routing-instance-name | routing-instance-name)>
<vlan (all | vlan-name)>
<vpn vpn>
```

Syntax (MX Series Routers)

```
show route forwarding-table
<detail | extensive | summary>
<all>
<bridge-domain (all | domain-name)>
<ccc interface-name>
<destination destination-prefix>
<family family | matching matching>
<interface-name interface-name>
<label name>
<learning-vlan-id learning-vlan-id>
<matching matching>
<multicast>
<table (default | logical-system-name/routing-instance-name | routing-instance-name)>
<vlan (all | vlan-name)>
<vpn vpn>
```


Syntax (TX Matrix and TX Matrix Plus Routers)

```
show route forwarding-table
<detail | extensive | summary>
<all>
```

```
<ccc interface-name>
<destination destination-prefix>
<family family | matching matching>
<interface-name interface-name>
<matching matching>
<label name>
<lcc number>
<multicast>
<table routing-instance-name>
<vpn vpn>
```

Description

Display the Routing Engine's forwarding table, including the network-layer prefixes and their next hops. This command is used to help verify that the routing protocol process has relayed the correction information to the forwarding table. The Routing Engine constructs and maintains one or more routing tables. From the routing tables, the Routing Engine derives a table of active routes, called the forwarding table.

**NOTE:** The Routing Engine copies the forwarding table to the Packet Forwarding Engine, the part of the router that is responsible for forwarding packets. To display the entries in the Packet Forwarding Engine's forwarding table, use the `show pfe route` command.

Options

none	Display the routes in the forwarding tables. By default, the <code>show route forwarding-table</code> command does not display information about private, or internal, forwarding tables.
detail extensive summary	(Optional) Display the specified level of output.
all	(Optional) Display routing table entries for all forwarding tables, including private, or internal, tables.
bridge-domain (all bridge-domain-name)	(MX Series routers only) (Optional) Display route entries for all bridge domains or the specified bridge domain.
ccc interface-name	(Optional) Display route entries for the specified circuit cross-connect interface.
destination destination-prefix	(Optional) Destination prefix.

family <i>family</i>	(Optional) Display routing table entries for the specified family: bridge (ccc destination detail extensive interface-name label learning-vlan-id matching multicast summary table vlan vpn), ethernet-switching, evpn, fibre-channel, fmembers, inet, inet6, iso, mcsnoop-inet, mcsnoop-inet6, mpls, satellite-inet, satellite-inet6, satellite-vpls, tnp, unix, vpls, or vlan-classification.
interface-name <i>interface-name</i>	(Optional) Display routing table entries for the specified interface.
label <i>name</i>	(Optional) Display route entries for the specified label.
lcc <i>number</i>	<p>(TX Matrix and TX matrix Plus routers only) (Optional) On a routing matrix composed of a TX Matrix router and T640 routers, display information for the specified T640 router (or line-card chassis) connected to the TX Matrix router. On a routing matrix composed of the TX Matrix Plus router and T1600 or T4000 routers, display information for the specified router (line-card chassis) connected to the TX Matrix Plus router.</p> <p>Replace <i>number</i> with the following values depending on the LCC configuration:</p> <ul style="list-style-type: none"> • 0 through 3, when T640 routers are connected to a TX Matrix router in a routing matrix. • 0 through 3, when T1600 routers are connected to a TX Matrix Plus router in a routing matrix. • 0 through 7, when T1600 routers are connected to a TX Matrix Plus router with 3D SIBs in a routing matrix. • 0, 2, 4, or 6, when T4000 routers are connected to a TX Matrix Plus router with 3D SIBs in a routing matrix.
learning-vlan-id <i>learning-vlan-id</i>	(MX Series routers only) (Optional) Display learned information for all VLANs or for the specified VLAN.
matching <i>matching</i>	(Optional) Display routing table entries matching the specified prefix or prefix length.
multicast	(Optional) Display routing table entries for multicast routes.
table	(Optional) Display route entries for all the routing tables in the main routing instance or for the specified routing instance. If your device supports logical systems, you can also display route entries for the specified logical system and routing instance. To view the routing instances on your device, use the <code>show route instance</code> command.

- vlan

(all | *vlan-name*)

(Optional) Display information for all VLANs or for the specified VLAN.
- vpn

vpn

(Optional) Display routing table entries for a specified VPN.

Required Privilege Level

view

Output Fields

Table 108 on page 1993 lists the output fields for the `show route forwarding-table` command. Output fields are listed in the approximate order in which they appear. Field names might be abbreviated (as shown in parentheses) when no level of output is specified, or when the `detail` keyword is used instead of the `extensive` keyword.

Table 108: show route forwarding-table Output Fields

Field Name	Field Description	Level of Output
Logical system	Name of the logical system. This field is displayed if you specify the table <i>logical-system-name/routing-instance-name</i> option on a device that is configured for and supports logical systems.	All levels
Routing table	Name of the routing table (for example, inet, inet6, mpls).	All levels

Table 108: show route forwarding-table Output Fields *(Continued)*

Field Name	Field Description	Level of Output
Enabled protocols	<p>The features and protocols that have been enabled for a given routing table. This field can contain the following values:</p> <ul style="list-style-type: none"> • BUM hashing—BUM hashing is enabled. • MAC Stats—Mac Statistics is enabled. • Bridging—Routing instance is a normal layer 2 bridge. • No VLAN—No VLANs are associated with the bridge domain. • All VLANs—The <code>vlan-id all</code> statement has been enabled for this bridge domain. • Single VLAN—Single VLAN ID is associated with the bridge domain. • MAC action drop—New MACs will be dropped when the MAC address limit is reached. • Dual VLAN—Dual VLAN tags are associated with the bridge domain • No local switching—No local switching is enabled for this routing instance.. • Learning disabled—Layer 2 learning is disabled for this routing instance. • MAC limit reached—The maximum number of MAC addresses that was configured for this routing instance has been reached. • VPLS—The VPLS protocol is enabled. • No IRB I2-copy—The <code>no-irb-layer-2-copy</code> feature is enabled for this routing instance. • ACKed by all peers—All peers have acknowledged this routing instance. • BUM Pruning—BUM pruning is enabled on the VPLS instance. • Def BD VXLAN—VXLAN is enabled for the default bridge domain. • EVPN—EVPN protocol is enabled for this routing instance. 	All levels

Table 108: show route forwarding-table Output Fields *(Continued)*

Field Name	Field Description	Level of Output
	<ul style="list-style-type: none"> • Def BD OVSDb—Open vSwitch Database (OVSDb) is enabled on the default bridge domain. • Def BD Ingress replication—VXLAN ingress node replication is enabled on the default bridge domain. • L2 backhaul—Layer 2 backhaul is enabled. • FRR optimize—Fast reroute optimization • MAC pinning—MAC pinning is enabled for this bridge domain. • MAC Aging Timer—The MAC table aging time is set per routing instance. • EVPN VXLAN—This routing instance supports EVPN with VXLAN encapsulation. • PBBN—This routing instance is configured as a provider backbone bridged network. • PBN—This routing instance is configured as a provider bridge network. • ETREE—The ETREE protocol is enabled on this EVPN routing instance. • ARP/NDP suppression—EVPN ARP NDP suppression is enabled in this routing instance. • Def BD EVPN VXLAN—EVPN VXLAN is enabled for the default bridge domain. • MPLS control word—Control word is enabled for this MPLS routing instance. 	
Address family	Address family (for example, IP, IPv6, ISO, MPLS, and VPLS).	All levels
Destination	Destination of the route.	detail extensive

Table 108: show route forwarding-table Output Fields (Continued)

Field Name	Field Description	Level of Output
Route Type (Type)	<p>How the route was placed into the forwarding table. When the detail keyword is used, the route type might be abbreviated (as shown in parentheses):</p> <ul style="list-style-type: none"> • cloned (clon)—(TCP or multicast only) Cloned route. • destination (dest)—Remote addresses directly reachable through an interface. • destination down (iddn)—Destination route for which the interface is unreachable. • interface cloned (ifcl)—Cloned route for which the interface is unreachable. • route down (ifdn)—Interface route for which the interface is unreachable. • ignore (ignr)—Ignore this route. • interface (intf)—Installed as a result of configuring an interface. • permanent (perm)—Routes installed by the kernel when the routing table is initialized. • user—Routes installed by the routing protocol process or as a result of the configuration. 	All levels
Route Reference (RtRef)	Number of routes to reference.	detail extensive

Table 108: show route forwarding-table Output Fields (*Continued*)

Field Name	Field Description	Level of Output
Flags	<p>Route type flags:</p> <ul style="list-style-type: none"> • none—No flags are enabled. • accounting—Route has accounting enabled. • cached—Cache route. • incoming-iface <i>interface-number</i>—Check against incoming interface. • prefix load balance—Load balancing is enabled for this prefix. • rt nh decoupled—Route has been decoupled from the next hop to the destination. • sent to PFE—Route has been sent to the Packet Forwarding Engine. • static—Static route. 	extensive
Next hop	<p>IP address of the next hop to the destination.</p> <p>NOTE: For static routes that use point-to-point (P2P) outgoing interfaces, the next-hop address is not displayed in the output.</p>	detail extensive

Table 108: show route forwarding-table Output Fields (*Continued*)

Field Name	Field Description	Level of Output
Next hop Type (Type)	<p>Next-hop type. When the detail keyword is used, the next-hop type might be abbreviated (as indicated in parentheses):</p> <ul style="list-style-type: none"> • broadcast (bcst)—Broadcast. • deny—Deny. • discard (dscd) —Discard. • hold—Next hop is waiting to be resolved into a unicast or multicast type. • indexed (idxd)—Indexed next hop. • indirect (indr)—Indirect next hop. • local (locl)—Local address on an interface. • routed multicast (mcrt)—Regular multicast next hop. • multicast (mcst)—Wire multicast next hop (limited to the LAN). • multicast discard (mdsc)—Multicast discard. • multicast group (mgrp)—Multicast group member. • receive (recv)—Receive. • reject (rjct)—Discard. An ICMP unreachable message was sent. • resolve (rslv)—Resolving the next hop. • unicast (ucst)—Unicast. • unilist (ulst)—List of unicast next hops. A packet sent to this next hop goes to any next hop in the list. • VxLAN Local—EVPN Type 5 route in kernel. 	detail extensive
Index	Software index of the next hop that is used to route the traffic for a given prefix.	detail extensive none

Table 108: show route forwarding-table Output Fields *(Continued)*

Field Name	Field Description	Level of Output
Route interface-index	Logical interface index from which the route is learned. For example, for interface routes, this is the logical interface index of the route itself. For static routes, this field is zero. For routes learned through routing protocols, this is the logical interface index from which the route is learned.	extensive
Reference (NhRef)	Number of routes that refer to this next hop.	detail extensive none
Next-hop interface (Netif)	Interface used to reach the next hop.	detail extensive none
Weight	Value used to distinguish primary, secondary, and fast reroute backup routes. Weight information is available when MPLS label-switched path (LSP) link protection, node-link protection, or fast reroute is enabled, or when the standby state is enabled for secondary paths. A lower weight value is preferred. Among routes with the same weight value, load balancing is possible (see the Balance field description).	extensive
Balance	Balance coefficient indicating how traffic of unequal cost is distributed among next hops when a router is performing unequal-cost load balancing. This information is available when you enable BGP multipath load balancing.	extensive
RPF interface	List of interfaces from which the prefix can be accepted. Reverse path forwarding (RPF) information is displayed only when rpf-check is configured on the interface.	extensive

Sample Output

show route forwarding-table

```

user@host> show route forwarding-table
Routing table: default.inet
Internet:

```

Destination	Type	RtRef	Next hop	Type	Index	NhRef	Netif
default	perm	0		rjct	46	4	
0.0.0.0/32	perm	0		dscd	44	1	
172.16.1.0/24	ifdn	0		rslv	608	1	ge-2/0/1.0
172.16.1.0/32	iddn	0	172.16.1.0	recv	606	1	ge-2/0/1.0
172.16.1.1/32	user	0		rjct	46	4	
172.16.1.1/32	intf	0	172.16.1.1	loc1	607	2	
172.16.1.1/32	iddn	0	172.16.1.1	loc1	607	2	
172.16.1.255/32	iddn	0	ff:ff:ff:ff:ff:ff	bcst	605	1	ge-2/0/1.0
10.0.0.0/24	intf	0		rslv	616	1	ge-2/0/0.0
10.0.0.0/32	dest	0	10.0.0.0	recv	614	1	ge-2/0/0.0
10.0.0.1/32	intf	0	10.0.0.1	loc1	615	2	
10.0.0.1/32	dest	0	10.0.0.1	loc1	615	2	
10.0.0.255/32	dest	0	10.0.0.255	bcst	613	1	ge-2/0/0.0
10.1.1.0/24	ifdn	0		rslv	612	1	ge-2/0/1.0
10.1.1.0/32	iddn	0	10.1.1.0	recv	610	1	ge-2/0/1.0
10.1.1.1/32	user	0		rjct	46	4	
10.1.1.1/32	intf	0	10.1.1.1	loc1	611	2	
10.1.1.1/32	iddn	0	10.1.1.1	loc1	611	2	
10.1.1.255/32	iddn	0	ff:ff:ff:ff:ff:ff	bcst	609	1	ge-2/0/1.0
10.209.0.0/16	user	0	10.209.63.254	ucst	419	20	fxp0.0
10.209.0.0/16	user	1	0:12:1e:ca:98:0	ucst	419	20	fxp0.0
10.209.0.0/18	intf	0		rslv	418	1	fxp0.0
10.209.0.0/32	dest	0	10.209.0.0	recv	416	1	fxp0.0
10.209.2.131/32	intf	0	10.209.2.131	loc1	417	2	
10.209.2.131/32	dest	0	10.209.2.131	loc1	417	2	
10.209.17.55/32	dest	0	0:30:48:5b:78:d2	ucst	435	1	fxp0.0
10.209.63.42/32	dest	0	0:23:7d:58:92:ca	ucst	434	1	fxp0.0
10.209.63.254/32	dest	0	0:12:1e:ca:98:0	ucst	419	20	fxp0.0
10.209.63.255/32	dest	0	10.209.63.255	bcst	415	1	fxp0.0
10.227.0.0/16	user	0	10.209.63.254	ucst	419	20	fxp0.0

...

Routing table: iso

ISO:

Destination	Type	RtRef	Next hop	Type	Index	NhRef	Netif
default	perm	0		rjct	27	1	
47.0005.80ff.f800.0000.0108.0003.0102.5524.5220.00							
intf 0			loc1 28				1

Routing table: inet6

Internet6:

Destination	Type	RtRef	Next hop	Type	Index	NhRef	Netif
default	perm	0		rjct	6	1	
ff00::/8	perm	0		mdsc	4	1	
ff02::1/128	perm	0	ff02::1	mcst	3	1	

Routing table: ccc

MPLS:

Interface.Label	Type	RtRef	Next hop	Type	Index	NhRef	Netif
default	perm	0		rjct	16	1	
100004(top)fe-0/0/1.0							

show route forwarding-table detail

```
user@host> show route forwarding-table detail
```

Routing table: inet

Internet:

Destination	Type	RtRef	Next hop	Type	Index	NhRef	Netif
default	user	2	0:90:69:8e:b1:1b	ucst	132	4	fxp0.0
default	perm	0		rjct	14	1	
10.1.1.0/24	intf	0	ff.3.0.21	ucst	322	1	so-5/3/0.0
10.1.1.0/32	dest	0	10.1.1.0	recv	324	1	so-5/3/0.0
10.1.1.1/32	intf	0	10.1.1.1	loc1	321	1	
10.1.1.255/32	dest	0	10.1.1.255	bcst	323	1	so-5/3/0.0
10.21.21.0/24	intf	0	ff.3.0.21	ucst	326	1	so-5/3/0.0
10.21.21.0/32	dest	0	10.21.21.0	recv	328	1	so-5/3/0.0
10.21.21.1/32	intf	0	10.21.21.1	loc1	325	1	
10.21.21.255/32	dest	0	10.21.21.255	bcst	327	1	so-5/3/0.0
127.0.0.1/32	intf	0	127.0.0.1	loc1	320	1	
172.17.28.19/32	clon	1	192.168.4.254	ucst	132	4	fxp0.0
172.17.28.44/32	clon	1	192.168.4.254	ucst	132	4	fxp0.0

...

Routing table: private1__.inet

Internet:

Destination	Type	RtRef	Next hop	Type	Index	NhRef	Netif
default	perm	0		rjct	46	1	
10.0.0.0/8	intf	0		rslv	136	1	fxp1.0
10.0.0.0/32	dest	0	10.0.0.0	recv	134	1	fxp1.0
10.0.0.4/32	intf	0	10.0.0.4	loc1	135	2	
10.0.0.4/32	dest	0	10.0.0.4	loc1	135	2	

...

Routing table: iso

ISO:

Destination	Type	RtRef	Next hop	Type	Index	NhRef	Netif
default	perm	0		rjct	38	1	

Routing table: inet6

Internet6:

Destination	Type	RtRef	Next hop	Type	Index	NhRef	Netif
default	perm	0		rjct	22	1	
ff00::/8	perm	0		mdsc	21	1	
ff02::1/128	perm	0	ff02::1	mcst	17	1	

...

Routing table: mpls

MPLS:

Destination	Type	RtRef	Next hop	Type	Index	NhRef	Netif
default	perm	0		rjct	28	1	

show route forwarding-table destination extensive (EVPN Type 5 route with Type 2 and Type 5 route coexistence)

```
user@device> show route forwarding-table destination 10.1.1.20 table vrf1 extensive
```

Routing table: vrf1.inet [Index 9]

Internet:

Destination: 10.1.1.20/32

Route type: user

Route reference: 0

Route interface-index: 0

Multicast RPF nh index: 0

P2mpidx: 0

Flags: sent to PFE, VxLAN Local

Nexthop:

Next-hop type: composite Index: 2694 Reference: 7

Next-hop type: indirect Index: 524326 Reference: 2

Next-hop type: unilist Index: 524288 Reference: 5

Nexthop: 10.1.1.1

Next-hop type: unicast Index: 1724 Reference: 15

Next-hop interface: xe-0/0/1.0 Weight: 0x0

Nexthop: 10.1.1.4 Next-hop type: unicast Index: 1725 Reference: 15
 Next-hop interface: xe-0/0/4.0 Weight: 0x0

show route forwarding-table extensive (RPF)

The next example is based on the following configuration, which enables an RPF check on all routes that are learned from this interface, including the interface route:

```

so-1/1/0 {
  unit 0 {
    family inet {
      rpf-check;
      address 192.0.2.2/30;
    }
  }
}

```

Release Information

Command introduced before Junos OS Release 7.4.

Option `bridge-domain` introduced in Junos OS Release 7.5

Option `learning-vlan-id` introduced in Junos OS Release 8.4

Options `all` and `vlan` introduced in Junos OS Release 9.6.

RELATED DOCUMENTATION

| *show route instance*

show route table

IN THIS SECTION

● Syntax | 2004

- Syntax (EX Series Switches, QFX Series Switches) | 2004
- Description | 2004
- Options | 2004
- Required Privilege Level | 2005
- Output Fields | 2005
- Sample Output | 2021
- Release Information | 2027

Syntax

```
show route table routing-table-name
<brief | detail | extensive | terse>
<logical-system (all | logical-system-name)>
```

Syntax (EX Series Switches, QFX Series Switches)

```
show route table routing-table-name
<brief | detail | extensive | terse>
```

Description

Display the route entries in a particular routing table.

Options

brief detail extensive terse	(Optional) Display the specified level of output.
logical-system (all <i>logical-system-name</i>)	(Optional) Perform this operation on all logical systems or on a particular logical system. This option is only supported on Junos OS.
<i>routing-table-name</i>	Display route entries for all routing tables whose names begin with this string (for example, inet.0 and inet6.0 are both displayed when you run the <code>show route table inet</code> command).

Required Privilege Level

view

Output Fields

Table 109 on page 2005 describes the output fields for the `show route table` command. Output fields are listed in the approximate order in which they appear.

Table 109: show route table Output Fields

Field Name	Field Description
<i>routing-table-name</i>	Name of the routing table (for example, inet.0).
Restart complete	<p>All protocols have restarted for this routing table.</p> <p>Restart state:</p> <ul style="list-style-type: none"> Pending:<i>protocol-name</i>—List of protocols that have not yet completed graceful restart for this routing table. Complete—All protocols have restarted for this routing table. <p>For example, if the output shows-</p> <ul style="list-style-type: none"> LDP.inet.0 : 5 routes (4 active, 1 holddown, 0 hidden) Restart Pending: OSPF LDP VPN <p>This indicates that OSPF, LDP, and VPN protocols did not restart for the LDP.inet.0 routing table.</p> <ul style="list-style-type: none"> vpls_1.l2vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden) Restart Complete <p>This indicates that all protocols have restarted for the vpls_1.l2vpn.0 routing table.</p>
<i>number destinations</i>	Number of destinations for which there are routes in the routing table.

Table 109: show route table Output Fields *(Continued)*

Field Name	Field Description
<i>number routes</i>	<p>Number of routes in the routing table and total number of routes in the following states:</p> <ul style="list-style-type: none">• active (routes that are active)• holddown (routes that are in the pending state before being declared inactive)• hidden (routes that are not used because of a routing policy)

Table 109: show route table Output Fields (*Continued*)

Field Name	Field Description
<i>route-destination</i> (entry, announced)	<p>Route destination (for example:10.0.0.1/24). The entry value is the number of routes for this destination, and the announced value is the number of routes being announced for this destination. Sometimes the route destination is presented in another format, such as:</p> <ul style="list-style-type: none"> • <i>MPLS-label</i>(for example, 80001). • <i>interface-name</i> (for example, ge-1/0/2). • <i>neighbor-address:control-word-status:encapsulation type:vc-id:source</i> (Layer 2 circuit only; for example, 10.1.1.195:NoCtrlWord:1:1:Local/96). • <i>neighbor-address</i>—Address of the neighbor. • <i>control-word-status</i>—Whether the use of the control word has been negotiated for this virtual circuit: NoCtrlWord or CtrlWord. • <i>encapsulation type</i>—Type of encapsulation, represented by a number: (1) Frame Relay DLCI, (2) ATM AAL5 VCC transport, (3) ATM transparent cell transport, (4) Ethernet, (5) VLAN Ethernet, (6) HDLC, (7) PPP, (8) ATM VCC cell transport, (10) ATM VPC cell transport. • <i>vc-id</i>—Virtual circuit identifier. • <i>source</i>—Source of the advertisement: Local or Remote. • <i>inclusive multicast Ethernet tag route</i>—Type of route destination represented by (for example, 3:100.100.100.10:100::0::10::100.100.100.10/384): <ul style="list-style-type: none"> • <i>route distinguisher</i>—(8 octets) Route distinguisher (RD) must be the RD of the EVPN instance (EVI) that is advertising the NLRI. • <i>Ethernet tag ID</i>—(4 octets) Identifier of the Ethernet tag. Can set to 0 or to a valid Ethernet tag value. • <i>IP address length</i>—(1 octet) Length of IP address in bits. • <i>originating router's IP address</i>—(4 or 16 octets) Must set to the provider edge (PE) device's IP address. This address should be common for all EVIs on the PE device, and may be the PE device's loopback address.

Table 109: show route table Output Fields *(Continued)*

Field Name	Field Description
label stacking	<p>(Next-to-the-last-hop routing device for MPLS only) Depth of the MPLS label stack, where the label-popping operation is needed to remove one or more labels from the top of the stack. A pair of routes is displayed, because the pop operation is performed only when the stack depth is two or more labels.</p> <ul style="list-style-type: none"> • S=0 route indicates that a packet with an incoming label stack depth of 2 or more exits this routing device with one fewer label (the label-popping operation is performed). • If there is no S= information, the route is a normal MPLS route, which has a stack depth of 1 (the label-popping operation is not performed).
[<i>protocol</i> , <i>preference</i>]	<p>Protocol from which the route was learned and the preference value for the route.</p> <ul style="list-style-type: none"> • +-A plus sign indicates the active route, which is the route installed from the routing table into the forwarding table. • -—A hyphen indicates the last active route. • *-An asterisk indicates that the route is both the active and the last active route. An asterisk before a to line indicates the best subpath to the route. <p>In every routing metric except for the BGP LocalPref attribute, a lesser value is preferred. In order to use common comparison routines, Junos OS stores the 1's complement of the LocalPref value in the Preference2 field. For example, if the LocalPref value for Route 1 is 100, the Preference2 value is -101. If the LocalPref value for Route 2 is 155, the Preference2 value is -156. Route 2 is preferred because it has a higher LocalPref value and a lower Preference2 value.</p>
Level	<p>(IS-IS only). In IS-IS, a single AS can be divided into smaller groups called areas. Routing between areas is organized hierarchically, allowing a domain to be administratively divided into smaller areas. This organization is accomplished by configuring Level 1 and Level 2 intermediate systems. Level 1 systems route within an area. When the destination is outside an area, they route toward a Level 2 system. Level 2 intermediate systems route between areas and toward other ASs.</p>
Route Distinguisher	IP subnet augmented with a 64-bit prefix.
PMSI	Provider multicast service interface (MVPN routing table).

Table 109: show route table Output Fields (Continued)

Field Name	Field Description
Next-hop type	Type of next hop. For a description of possible values for this field, see Table 110 on page 2014 .
Next-hop reference count	Number of references made to the next hop.
Flood nexthop branches exceed maximum message	Indicates that the number of flood next-hop branches exceeded the system limit of 32 branches, and only a subset of the flood next-hop branches were installed in the kernel.
Source	IP address of the route source.
Next hop	Network layer address of the directly reachable neighboring system.
via	<p>Interface used to reach the next hop. If there is more than one interface available to the next hop, the name of the interface that is actually used is followed by the word Selected. This field can also contain the following information:</p> <ul style="list-style-type: none"> • Weight—Value used to distinguish primary, secondary, and fast reroute backup routes. Weight information is available when MPLS label-switched path (LSP) link protection, node-link protection, or fast reroute is enabled, or when the standby state is enabled for secondary paths. A lower weight value is preferred. Among routes with the same weight value, load balancing is possible. • Balance—Balance coefficient indicating how traffic of unequal cost is distributed among next hops when a routing device is performing unequal-cost load balancing. This information is available when you enable BGP multipath load balancing.
Label-switched-path <i>lsp-path-name</i>	Name of the LSP used to reach the next hop.
Label operation	MPLS label and operation occurring at this routing device. The operation can be pop (where a label is removed from the top of the stack), push (where another label is added to the label stack), or swap (where a label is replaced by another label).

Table 109: show route table Output Fields (Continued)

Field Name	Field Description
Interface	(Local only) Local interface name.
Protocol next hop	Network layer address of the remote routing device that advertised the prefix. This address is used to derive a forwarding next hop.
Indirect next hop	Index designation used to specify the mapping between protocol next hops, tags, kernel export policy, and the forwarding next hops.
State	State of the route (a route can be in more than one state). See Table 111 on page 2016 .
Local AS	AS number of the local routing devices.
Age	How long the route has been known.
AI GP	Accumulated interior gateway protocol (AIGP) BGP attribute.
Metric <i>n</i>	Cost value of the indicated route. For routes within an AS, the cost is determined by IGP and the individual protocol metrics. For external routes, destinations, or routing domains, the cost is determined by a preference value.
MED-plus-IGP	Metric value for BGP path selection to which the IGP cost to the next-hop destination has been added.
TTL-Action	For MPLS LSPs, state of the TTL propagation attribute. Can be enabled or disabled for all RSVP-signaled and LDP-signaled LSPs or for specific VRF routing instances.
Task	Name of the protocol that has added the route.

Table 109: show route table Output Fields (Continued)

Field Name	Field Description
Announcement bits	<p>The number of BGP peers or protocols to which Junos OS has announced this route, followed by the list of the recipients of the announcement. Junos OS can also announce the route to the kernel routing table (KRT) for installing the route into the Packet Forwarding Engine, to a resolve tree, a Layer 2 VC, or even a VPN. For example, <i>n-Resolve inet</i> indicates that the specified route is used for route resolution for next hops found in the routing table.</p> <ul style="list-style-type: none"> <i>n</i>—An index used by Juniper Networks customer support only.
AS path	<p>AS path through which the route was learned. The letters at the end of the AS path indicate the path origin, providing an indication of the state of the route at the point at which the AS path originated:</p> <ul style="list-style-type: none"> I—IGP. E—EGP. Recorded—The AS path is recorded by the sample process (sampled). ?—Incomplete; typically, the AS path was aggregated. <p>When AS path numbers are included in the route, the format is as follows:</p> <ul style="list-style-type: none"> []—Brackets enclose the number that precedes the AS path. This number represents the number of ASs present in the AS path, when calculated as defined in RFC 4271. This value is used in the AS-path merge process, as defined in RFC 4893. []—If more than one AS number is configured on the routing device, or if AS path prepending is configured, brackets enclose the local AS number associated with the AS path. { }—Braces enclose AS sets, which are groups of AS numbers in which the order does not matter. A set commonly results from route aggregation. The numbers in each AS set are displayed in ascending order. ()—Parentheses enclose a confederation. ([])—Parentheses and brackets enclose a confederation set. <p>NOTE: In Junos OS Release 10.3 and later, the AS path field displays an unrecognized attribute and associated hexadecimal value if BGP receives attribute 128 (attribute set) and you have not configured an independent domain in any routing instance.</p>

Table 109: show route table Output Fields *(Continued)*

Field Name	Field Description
validation-state	<p>(BGP-learned routes) Validation status of the route:</p> <ul style="list-style-type: none"> • Invalid—Indicates that the prefix is found, but either the corresponding AS received from the EBGP peer is not the AS that appears in the database, or the prefix length in the BGP update message is longer than the maximum length permitted in the database. • Unknown—Indicates that the prefix is not among the prefixes or prefix ranges in the database. • Unverified—Indicates that the origin of the prefix is not verified against the database. This is because the database got populated and the validation is not called for in the BGP import policy, although origin validation is enabled, or the origin validation is not enabled for the BGP peers. • Valid—Indicates that the prefix and autonomous system pair are found in the database.
FECs bound to route	Indicates point-to-multipoint root address, multicast source address, and multicast group address when multipoint LDP (M-LDP) inband signaling is configured.
Primary Upstream	When multipoint LDP with multicast-only fast reroute (MoFRR) is configured, indicates the primary upstream path. MoFRR transmits a multicast join message from a receiver toward a source on a primary path, while also transmitting a secondary multicast join message from the receiver toward the source on a backup path.
RPF Nexthops	When multipoint LDP with MoFRR is configured, indicates the reverse-path forwarding (RPF) next-hop information. Data packets are received from both the primary path and the secondary paths. The redundant packets are discarded at topology merge points due to the RPF checks.
Label	Multiple MPLS labels are used to control MoFRR stream selection. Each label represents a separate route, but each references the same interface list check. Only the primary label is forwarded while all others are dropped. Multiple interfaces can receive packets using the same label.
weight	Value used to distinguish MoFRR primary and backup routes. A lower weight value is preferred. Among routes with the same weight value, load balancing is possible.

Table 109: show route table Output Fields (Continued)

Field Name	Field Description
VC Label	MPLS label assigned to the Layer 2 circuit virtual connection.
MTU	Maximum transmission unit (MTU) of the Layer 2 circuit.
VLAN ID	VLAN identifier of the Layer 2 circuit.
Prefixes bound to route	Forwarding equivalent class (FEC) bound to this route. Applicable only to routes installed by LDP.
Communities	Community path attribute for the route. See Table 112 on page 2020 for all possible values for this field.
Layer2-info: encaps	Layer 2 encapsulation (for example, VPLS).
control flags	Control flags: none or Site Down.
mtu	Maximum transmission unit (MTU) information.
Label-Base, range	First label in a block of labels and label block size. A remote PE routing device uses this first label when sending traffic toward the advertising PE routing device.
status vector	Layer 2 VPN and VPLS network layer reachability information (NLRI).
Accepted Multipath	Current active path when BGP multipath is configured.
Accepted LongLivedStale	The LongLivedStale flag indicates that the route was marked LLGR-stale by this router, as part of the operation of LLGR receiver mode. Either this flag or the LongLivedStaleImport flag might be displayed for a route. Neither of these flags is displayed at the same time as the Stale (ordinary GR stale) flag.

Table 109: show route table Output Fields (Continued)

Field Name	Field Description
Accepted LongLivedStaleImport	<p>The LongLivedStaleImport flag indicates that the route was marked LLGR-stale when it was received from a peer, or by import policy. Either this flag or the LongLivedStale flag might be displayed for a route. Neither of these flags is displayed at the same time as the Stale (ordinary GR stale) flag.</p> <p>Accept all received BGP long-lived graceful restart (LLGR) and LLGR stale routes learned from configured neighbors and import into the inet.0 routing table</p>
ImportAccepted LongLivedStaleImport	<p>Accept all received BGP long-lived graceful restart (LLGR) and LLGR stale routes learned from configured neighbors and imported into the inet.0 routing table</p> <p>The LongLivedStaleImport flag indicates that the route was marked LLGR-stale when it was received from a peer, or by import policy.</p>
Accepted MultipathContrib	Path currently contributing to BGP multipath.
Localpref	Local preference value included in the route.
Router ID	BGP router ID as advertised by the neighbor in the open message.
Primary Routing Table	In a routing table group, the name of the primary routing table in which the route resides.
Secondary Tables	In a routing table group, the name of one or more secondary tables in which the route resides.

[Table 110 on page 2014](#) describes all possible values for the Next-hop Types output field.

Table 110: Next-hop Types Output Field Values

Next-Hop Type	Description
Broadcast (bcast)	Broadcast next hop.

Table 110: Next-hop Types Output Field Values *(Continued)*

Next-Hop Type	Description
Deny	Deny next hop.
Discard	Discard next hop.
Flood	Flood next hop. Consists of components called branches, up to a maximum of 32 branches. Each flood next-hop branch sends a copy of the traffic to the forwarding interface. Used by point-to-multipoint RSVP, point-to-multipoint LDP, point-to-multipoint CCC, and multicast.
Hold	Next hop is waiting to be resolved into a unicast or multicast type.
Indexed (idxd)	Indexed next hop.
Indirect (indr)	Used with applications that have a protocol next hop address that is remote. You are likely to see this next-hop type for internal BGP (IBGP) routes when the BGP next hop is a BGP neighbor that is not directly connected.
Interface	Used for a network address assigned to an interface. Unlike the router next hop, the interface next hop does not reference any specific node on the network.
Local (locl)	Local address on an interface. This next-hop type causes packets with this destination address to be received locally.
Multicast (mcst)	Wire multicast next hop (limited to the LAN).
Multicast discard (mdsc)	Multicast discard.
Multicast group (mgrp)	Multicast group member.
Receive (recv)	Receive.

Table 110: Next-hop Types Output Field Values (Continued)

Next-Hop Type	Description
Reject (rjct)	Discard. An ICMP unreachable message was sent.
Resolve (rslv)	Resolving next hop.
Routed multicast (mcrt)	Regular multicast next hop.
Router	<p>A specific node or set of nodes to which the routing device forwards packets that match the route prefix.</p> <p>To qualify as a next-hop type router, the route must meet the following criteria:</p> <ul style="list-style-type: none"> • Must not be a direct or local subnet for the routing device. • Must have a next hop that is directly connected to the routing device.
Table	Routing table next hop.
Unicast (ucst)	Unicast.
Unilist (ulst)	List of unicast next hops. A packet sent to this next hop goes to any next hop in the list.

[Table 111 on page 2016](#) describes all possible values for the State output field. A route can be in more than one state (for example, <Active NoReadvrt Int Ext>).

Table 111: State Output Field Values

Value	Description
Accounting	Route needs accounting.
Active	Route is active.

Table 111: State Output Field Values (Continued)

Value	Description
Always Compare MED	Path with a lower multiple exit discriminator (MED) is available.
AS path	Shorter AS path is available.
Cisco Non-deterministic MED selection	Cisco nondeterministic MED is enabled, and a path with a lower MED is available.
Clone	Route is a clone.
Cluster list length	Length of cluster list sent by the route reflector.
Delete	Route has been deleted.
Ex	Exterior route.
Ext	BGP route received from an external BGP neighbor.
FlashAll	Forces all protocols to be notified of a change to any route, active or inactive, for a prefix. When not set, protocols are informed of a prefix only when the active route changes.
Hidden	Route not used because of routing policy.
IfCheck	Route needs forwarding RPF check.
IGP metric	Path through next hop with lower IGP metric is available.
Inactive reason	Flags for this route, which was not selected as best for a particular destination.
Initial	Route being added.

Table 111: State Output Field Values (Continued)

Value	Description
Int	Interior route.
Int Ext	BGP route received from an internal BGP peer or a BGP confederation peer.
Interior > Exterior > Exterior via Interior	Direct, static, IGP, or EBGp path is available.
Local Preference	Path with a higher local preference value is available.
Martian	Route is a martian (ignored because it is obviously invalid).
MartianOK	Route exempt from martian filtering.
Next hop address	Path with lower metric next hop is available.
No difference	Path from neighbor with lower IP address is available.
NoReadvrt	Route not to be advertised.
NotBest	Route not chosen because it does not have the lowest MED.
Not Best in its group	Incoming BGP AS is not the best of a group (only one AS can be the best).
NotInstall	Route not to be installed in the forwarding table.
Number of gateways	Path with a greater number of next hops is available.
Origin	Path with a lower origin code is available.

Table 111: State Output Field Values (Continued)

Value	Description
Pending	Route pending because of a hold-down configured on another route.
Release	Route scheduled for release.
RIB preference	Route from a higher-numbered routing table is available.
Route Distinguisher	64-bit prefix added to IP subnets to make them unique.
Route Metric or MED comparison	Route with a lower metric or MED is available.
Route Preference	Route with lower preference value is available.
Router ID	Path through a neighbor with lower ID is available.
Secondary	Route not a primary route.
Unusable path	Path is not usable because of one of the following conditions: <ul style="list-style-type: none"> • The route is damped. • The route is rejected by an import policy. • The route is unresolved.
Update source	Last tiebreaker is the lowest IP address value.
VxlanLocalRT	Route is an EVPN Type 5 route (IP prefix route).

[Table 112 on page 2020](#) describes the possible values for the Communities output field.

Table 112: Communities Output Field Values

Value	Description
<i>area-number</i>	4 bytes, encoding a 32-bit area number. For AS-external routes, the value is 0. A nonzero value identifies the route as internal to the OSPF domain, and as within the identified area. Area numbers are relative to a particular OSPF domain.
<i>bandwidth: local AS number:link-bandwidth-number</i>	Link-bandwidth community value used for unequal-cost load balancing. When BGP has several candidate paths available for multipath purposes, it does not perform unequal-cost load balancing according to the link-bandwidth community unless all candidate paths have this attribute.
<i>domain-id</i>	Unique configurable number that identifies the OSPF domain.
<i>domain-id-vendor</i>	Unique configurable number that further identifies the OSPF domain.
<i>link-bandwidth-number</i>	Link-bandwidth number: from 0 through 4,294,967,295 (bytes per second).
<i>local AS number</i>	Local AS number: from 1 through 65,535.
<i>options</i>	1 byte. Currently this is only used if the route type is 5 or 7. Setting the least significant bit in the field indicates that the route carries a type 2 metric.
<i>origin</i>	(Used with VPNs) Identifies where the route came from.
<i>ospf-route-type</i>	1 byte, encoded as 1 or 2 for intra-area routes (depending on whether the route came from a type 1 or a type 2 LSA); 3 for summary routes; 5 for external routes (area number must be 0); 7 for NSSA routes; or 129 for sham link endpoint addresses.
<i>route-type-vendor</i>	Displays the area number, OSPF route type, and option of the route. This is configured using the BGP extended community attribute 0x8000. The format is <i>area-number:ospf-route-type:options</i> .
<i>rte-type</i>	Displays the area number, OSPF route type, and option of the route. This is configured using the BGP extended community attribute 0x0306. The format is <i>area-number:ospf-route-type:options</i> .

Table 112: Communities Output Field Values (Continued)

Value	Description
target	Defines which VPN the route participates in; target has the format <i>32-bit IP address:16-bit number</i> . For example, 10.19.0.0:100.
unknown IANA	Incoming IANA codes with a value between 0x1 and 0x7fff. This code of the BGP extended community attribute is accepted, but it is not recognized.
unknown OSPF vendor community	Incoming IANA codes with a value above 0x8000. This code of the BGP extended community attribute is accepted, but it is not recognized.
evpn-mcast-flags	Identifies the value in the multicast flags extended community and whether snooping is enabled. A value of 0x1 indicates that the route supports IGMP proxy.
evpn-l2-info	<p>Identifies whether Multihomed Proxy MAC and IP Address Route Advertisement is enabled. A value of 0x20 indicates that the proxy bit is set. .</p> <p>Use the <code>show bridge mac-ip-table extensive</code> statement to determine whether the MAC and IP address route was learned locally or from a PE device.</p>

Sample Output

show route table bgp.l2vpn.0

```

user@host> show route table bgp.l2vpn.0
bgp.l2vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.24.1:1:4:1/96
    *[BGP/170] 01:08:58, localpref 100, from 192.168.24.1
    AS path: I
    > to 10.0.16.2 via fe-0/0/1.0, label-switched-path am

```


show route table inet.0

```

user@host> show route table inet.0
inet.0: 12 destinations, 12 routes (11 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[Static/5] 00:51:57
                   > to 172.16.5.254 via fxp0.0
10.0.0.1/32        *[Direct/0] 00:51:58
                   > via at-5/3/0.0
10.0.0.2/32        *[Local/0] 00:51:58
                   Local
10.12.12.21/32     *[Local/0] 00:51:57
                   Reject
10.13.13.13/32     *[Direct/0] 00:51:58
                   > via t3-5/2/1.0
10.13.13.14/32     *[Local/0] 00:51:58
                   Local
10.13.13.21/32     *[Local/0] 00:51:58
                   Local
10.13.13.22/32     *[Direct/0] 00:33:59
                   > via t3-5/2/0.0
127.0.0.1/32      [Direct/0] 00:51:58
                   > via lo0.0
10.222.5.0/24     *[Direct/0] 00:51:58
                   > via fxp0.0
10.222.5.81/32    *[Local/0] 00:51:58
                   Local

```

show route table inet.3

```

user@host> show route table inet.3
inet.3: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.5/32        *[LDP/9] 00:25:43, metric 10, tag 200
                   to 10.2.94.2 via lt-1/2/0.49
                   > to 10.2.3.2 via lt-1/2/0.23

```

show route table inet.3 protocol ospf

```

user@host> show route table inet.3 protocol ospf
inet.3: 9 destinations, 18 routes (9 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1.1.1.20/32      [L-OSPF/10] 1d 00:00:56, metric 2
                 > to 10.0.10.70 via lt-1/2/0.14, Push 800020
                 to 10.0.6.60 via lt-1/2/0.12, Push 800020, Push 800030(top)
1.1.1.30/32      [L-OSPF/10] 1d 00:01:01, metric 3
                 > to 10.0.10.70 via lt-1/2/0.14, Push 800030
                 to 10.0.6.60 via lt-1/2/0.12, Push 800030
1.1.1.40/32      [L-OSPF/10] 1d 00:01:01, metric 4
                 > to 10.0.10.70 via lt-1/2/0.14, Push 800040
                 to 10.0.6.60 via lt-1/2/0.12, Push 800040
1.1.1.50/32      [L-OSPF/10] 1d 00:01:01, metric 5
                 > to 10.0.10.70 via lt-1/2/0.14, Push 800050
                 to 10.0.6.60 via lt-1/2/0.12, Push 800050
1.1.1.60/32      [L-OSPF/10] 1d 00:01:01, metric 6
                 > to 10.0.10.70 via lt-1/2/0.14, Push 800060
                 to 10.0.6.60 via lt-1/2/0.12, Pop

```

show route table inet6.0

```

user@host> show route table inet6.0
inet6.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Route, * = Both

fec0:0:0:3::/64 *[Direct/0] 00:01:34
>via fe-0/1/0.0

fec0:0:0:3::/128 *[Local/0] 00:01:34
>Local

fec0:0:0:4::/64 *[Static/5] 00:01:34
>to fec0:0:0:3::ffff via fe-0/1/0.0

```

show route table inet6.3

```

user@router> show route table inet6.3
inet6.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

::10.255.245.195/128
    *[LDP/9] 00:00:22, metric 1
    > via so-1/0/0.0
::10.255.245.196/128
    *[LDP/9] 00:00:08, metric 1
    > via so-1/0/0.0, Push 100008

```

show route table l2circuit.0

```

user@host> show route table l2circuit.0
l2circuit.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.1.1.195:NoCtrlWord:1:1:Local/96
    *[L2CKT/7] 00:50:47
    > via so-0/1/2.0, Push 100049
    via so-0/1/3.0, Push 100049
10.1.1.195:NoCtrlWord:1:1:Remote/96
    *[LDP/9] 00:50:14
    Discard
10.1.1.195:CtrlWord:1:2:Local/96
    *[L2CKT/7] 00:50:47
    > via so-0/1/2.0, Push 100049
    via so-0/1/3.0, Push 100049
10.1.1.195:CtrlWord:1:2:Remote/96
    *[LDP/9] 00:50:14
    Discard

```

show route table lsdist.0

```

user@host> show route table lsdist.0
lsdist.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

```

```

LINK { Local { AS:4 BGP-LS ID:100 IPv4:4.4.4.4 }.{ IPv4:4.4.4.4 } Remote { AS:4 BGP-LS ID:100
IPv4:7.7.7.7 }.{ IPv4:7.7.7.7 } Undefined:0 }/1152
      *[BGP-LS-EPE/170] 00:20:56
      Fictitious
LINK { Local { AS:4 BGP-LS ID:100 IPv4:4.4.4.4 }.{ IPv4:4.4.4.4 IfIndex:339 } Remote { AS:4 BGP-
LS ID:100 IPv4:7.7.7.7 }.{ IPv4:7.7.7.7 } Undefined:0 }/1152
      *[BGP-LS-EPE/170] 00:20:56
      Fictitious
LINK { Local { AS:4 BGP-LS ID:100 IPv4:4.4.4.4 }.{ IPv4:50.1.1.1 } Remote { AS:4 BGP-LS ID:100
IPv4:5.5.5.5 }.{ IPv4:50.1.1.2 } Undefined:0 }/1152
      *[BGP-LS-EPE/170] 00:20:56
      Fictitious

```

show route table lsdist.0 detail

```

user@host> show route table lsdist.0 detail
lsdist.0: 14 destinations, 14 routes (14 active, 0 holddown, 0 hidden)
NODE { AS:200 ISO:1282.2113.1154.00 ISIS-L1:0 }/1216 (1 entry, 1 announced)
*IS-IS Preference: 15
Level: 1
Next hop type: Fictitious, Next hop index: 0
Address: 0xc5b3054
Next-hop reference count: 14
.....
.....
.....
Area membership:
47 00 05 80 ff f8 00 00 00 01 08 00 01
SPRING-Capabilities:
- SRGB block [Start: 800000, Range: 4096, Flags: 0xc0]
SPRING-Algorithms:
- Algo: 0
SPRING Flex-Algorithms Definition:
- Flex-Algo: 129
Metric: 0, Calc: 0, priority: 129
- Flags: 0x02, - Inc Any: 0x00040000, - Exclude: 0x00008000, - Inc All: 0x00004000
.....
.....
.....

```

```

PREFIX { Node { AS:200 ISO:1282.2113.3158.00 } { IPv4:128.220.13.196/32 } ISIS-L1:0 }/1216 (1
entry, 1 announced)
*IS-IS Preference: 15
Level: 1
Next hop type: Fictitious, Next hop index: 0
Address: 0xc5b3054
Next-hop reference count: 14
Next hop:
State: <Active NotInstall>
Local AS: 200
Age: 16:16:25
.....
.....
Prefix SID: 10, Flags: 0xe0, Algo: 0
Prefix SID: 780, Flags: 0xe0, Algo: 129
Flex Algo: 129, Flex Algo Metric: 10

```

show route table mpls

```

user@host> show route table mpls
mpls.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0          *[MPLS/0] 00:13:55, metric 1
           Receive
1          *[MPLS/0] 00:13:55, metric 1
           Receive
2          *[MPLS/0] 00:13:55, metric 1
           Receive
1024       *[VPN/0] 00:04:18
           to table red.inet.0, Pop

```

show route table mpls.0 protocol ospf

```

user@host> show route table mpls.0 protocol ospf
mpls.0: 29 destinations, 29 routes (29 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

299952     *[L-OSPF/10] 23:59:42, metric 0
           > to 10.0.10.70 via lt-1/2/0.14, Pop

```

```

                to 10.0.6.60 via lt-1/2/0.12, Swap 800070, Push 800030(top)
299952(S=0)      *[L-OSPF/10] 23:59:42, metric 0
                  > to 10.0.10.70 via lt-1/2/0.14, Pop
                  to 10.0.6.60 via lt-1/2/0.12, Swap 800070, Push 800030(top)
299968           *[L-OSPF/10] 23:59:48, metric 0
                  > to 10.0.6.60 via lt-1/2/0.12, Pop

```

show route table VPN-AB.inet.0

```

user@host> show route table VPN-AB.inet.0
VPN-AB.inet.0: 8 destinations, 8 routes (8 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.39.1.0/30      *[OSPF/10] 00:07:24, metric 1
                  > via so-7/3/1.0
10.39.1.4/30      *[Direct/0] 00:08:42
                  > via so-5/1/0.0
10.39.1.6/32      *[Local/0] 00:08:46
                  Local
10.255.71.16/32   *[Static/5] 00:07:24
                  > via so-2/0/0.0
10.255.71.17/32   *[BGP/170] 00:07:24, MED 1, localpref 100, from
10.255.71.15
                  AS path: I
                  > via so-2/1/0.0, Push 100020, Push 100011(top)
10.255.71.18/32   *[BGP/170] 00:07:24, MED 1, localpref 100, from
10.255.71.15
                  AS path: I
                  > via so-2/1/0.0, Push 100021, Push 100011(top)
10.255.245.245/32 *[BGP/170] 00:08:35, localpref 100
                  AS path: 2 I
                  > to 10.39.1.5 via so-5/1/0.0
10.255.245.246/32 *[OSPF/10] 00:07:24, metric 1
                  > via so-7/3/1.0

```

Release Information

Command introduced before Junos OS Release 7.4.

Show route table evpn statement introduced in Junos OS Release 15.1X53-D30 for QFX Series switches.

RELATED DOCUMENTATION

[show route summary](#)

show vlans evpn nd-table

IN THIS SECTION

- [Syntax | 2028](#)
- [Description | 2028](#)
- [Options | 2028](#)
- [Required Privilege Level | 2029](#)
- [Output Fields | 2029](#)
- [Sample Output | 2030](#)
- [Release Information | 2032](#)

Syntax

```
show vlans evpn nd-table  
<brief | detail | extensive>  
<count>  
<instance-name>  
<mac-address>  
<vlan-name>
```

Description

Display information about INET entries associated with MAC addresses learned through Network Discovery Protocol (NDP).

Options

none	Display information for all INET entries.
-------------	---

brief detail extensive	(Optional) Display the specified level of output.
count	(Optional) Display the number of INET addresses learned in a routing instance.
instance	(Optional) Display information for a specified instance.
mac-address	(Optional) Display information for a specified MAC address.
vlan-name (all)	(Optional) Display information for a specified VLAN or for all VLANs.

Required Privilege Level

view

Output Fields

[Table 113 on page 2029](#) lists the output fields for the `show vlans evpn nd-table` command. Output fields are listed in the approximate order in which they appear.

Table 113: show vlans evpn nd-table Output Fields

Field Name	Field Description	Level of Output
INET address	The INET address related to the INET entries that are added to the NDP table.	All levels
MAC address	MAC addresses learned through NDP.	brief, detail, extensive, instance, mac-address, vlan-name,,
Logical Interface	Logical interface associated with the routing instance in which the NDP INET address is learned.	brief, instance, mac-address, vlan-name,,
Routing instance	Routing instance in which the NDP INET address is learned.	all levels
Bridging domain	Bridging domain in which the NDP INET address is learned.	all levels
Learning interface	Interface on which the NDP INET address is learned.	detail, extensive

Table 113: show vlans evpn nd-table Output Fields (Continued)

Field Name	Field Description	Level of Output
Count	Indicates the number of NDP INET addresses learned in a routing instance in a bridge domain.	count

Sample Output

show vlans evpn nd-table brief

```
user@switch> show vlans evpn nd-table brief
INET          MAC          Logical      Routing      Bridging
address       address      interface    instance     domain
8002::2       00:05:86:a0:d5:00  irb.0        evpn1        vlan10
```

show vlans evpn nd-table detail

```
user@switch> show vlans evpn nd-table detail
INET address: 8002::2
MAC address: 00:05:86:a0:d5:00
  Routing instance: evpn1
  Bridging domain: vlan10
  Learning interface: irb.0
```

show vlans evpn nd-table extensive

```
user@switch> show vlans evpn nd-table extensive
INET address: 8002::2
MAC address: 00:05:86:a0:d5:00
  Routing instance: evpn1
  Bridging domain: vlan10
  Learning interface: irb.0
```

show vlans evpn nd-table count

```
user@switch> show vlans evpn nd-table count
1 ND INET addresses learned in routing instance evpn1 bridge domain vlan10
```

show vlans evpn nd-table instance evpn1

```
user@switch> show vlans evpn nd-table instance
evpn1
INET          MAC          Logical   Routing   Bridging
address       address      interface instance   domain
8002::2
              00:05:86:a0:d5:00
  irb.0       evpn1      vlan10
```

show vlans evpn nd-table instance 00:05:86:90:bd:f0 (MAC address)

```
user@switch> show vlans evpn nd-table
INET          MAC          Logical   Routing   Bridging
address       address      interface instance   domain
8002::2
              00:05:86:a0:d5:00
  irb.0       evpn1      __evpn1__
```

show vlans evpn nd-table vlan-name vlan10

```
user@switch> show vlans evpn nd-table vlan-name
vlan10

INET          MAC          Logical   Routing   Bridging
address       address      interface instance   domain
8002::2       00:05:86:a0:d5:00
  irb.0       evpn1      vlan10
```

Release Information

Command introduced in Junos OS Release 17.3R1.

NOTE: Starting with Junos OS Release 17.4R2, the `show vlans evpn nd-table` command is replaced by the `show ethernet-switching evpn nd-table` command.

VXLAN Operational Commands

IN THIS CHAPTER

- [show flexible-tunnels profiles | 2033](#)
- [show security flow tunnel inspection statistics | 2039](#)
- [show security policies policy set | 2042](#)
- [show security tunnel inspection | 2045](#)

show flexible-tunnels profiles

IN THIS SECTION

- [Syntax | 2033](#)
- [Description | 2034](#)
- [Options | 2034](#)
- [Required Privilege Level | 2034](#)
- [Output Fields | 2034](#)
- [Sample Output | 2035](#)
- [Release Information | 2039](#)

Syntax

```
show flexible-tunnels profiles  
<brief | detail>  
<tunnel-profile-name>  
<route ip-address>
```

Description

Display information about flexible Virtual Extensible LAN (VXLAN) tunnel de-encapsulation profiles, which are programmed by controllers through the Juniper Extension Toolkit (JET) APIs.

Options

- none

(Same as brief) Display information about all tunnel de-encapsulation profiles.
- brief | detail

(Optional) Display the specified level of output.
- tunnel-profile-name

Display information about the specified tunnel de-encapsulation profile.
- route ip-address

Display information about the specified internal route in the __flexible_tunnel_profiles__.inet.0 routing table.

Required Privilege Level

admin

Output Fields

Table 114 on page 2034 lists the output fields for the show flexible-tunnels profiles command. Output fields are listed in the approximate order in which they appear.

Table 114: show flexible-tunnels profiles Output Fields

Field Name	Field Description
tunnel-profile-name	Tunnel de-encapsulation profile name, if one was specified.
Client ID	An identifier of the client that injected the route specified in the tunnel profile.
Route prefix	An internal route that is associated with a particular tunnel de-encapsulation profile.
Table	Name of routing table, which is __flexible_tunnel_profiles__.inet.0.

Table 114: show flexible-tunnels profiles Output Fields (Continued)

Field Name	Field Description
Action	Action is decapsulate. If the contents of an incoming encapsulated packet matches the attributes specified in a particular tunnel de-capsulation profile, the packet is de-encapsulated as specified in the profile.
Tunnel type	Tunnel type is either VXLAN-IPv4 or VXLAN-IPv6.
Interface	Name of the flexible tunnel interface for use with statistics, policing, filtering, and default encapsulation parameters.
VNI	VXLAN network identifier (VNI) specified in the tunnel de-encapsulation profile
Source prefix	IPv4 or IPv6 network address and length of the source that originated the packet.
Source UDP port range	A range of UDP port numbers, one of which is configured on the port that originated the packet.
Destination address	IPv4 or IPv6 network address of the host to which the packet is destined.
Destination UDP port	UDP port number that is configured on the host port to which the packet is destined.
VXLAN flags	An 8-bit VXLAN flag.

Sample Output

show flexible-tunnels profiles (none and brief)

```

user@host> show flexible-tunnels profiles
user@host> show flexible-tunnels profiles brief
CUSTOMER_0001-decap-mvnet1_1
  Client ID: ABCD, Route Prefix: 1.0.0.0/32, Table: __flexible_tunnel_profiles__.inet.0

```

```

    Action: Decapsulate, Tunnel type: VXLAN-IPv6
CUSTOMER_0001-decap-mvnet1_10
    Client ID: ABCD, Route Prefix: 1.0.0.9/32, Table: __flexible_tunnel_profiles__inet.0
    Action: Decapsulate, Tunnel type: VXLAN-IPv6
CUSTOMER_0001-decap-mvnet1_11
    Client ID: ABCD, Route Prefix: 1.0.0.10/32, Table: __flexible_tunnel_profiles__inet.0
    Action: Decapsulate, Tunnel type: VXLAN-IPv6
...
CUSTOMER_0001-decap-mvnet1_50001
    Client ID: ABCD, Route Prefix: 1.0.195.80/32, Table: __flexible_tunnel_profiles__inet.0
    Action: Decapsulate, Tunnel type: VXLAN-IPv6
...

```

show flexible-tunnels profiles detail

```

user@host> show flexible-tunnels profiles detail
CUSTOMER_0001-decap-mvnet1_1
    Client ID: ABCD, Route Prefix: 1.0.0.0/32, Table: __flexible_tunnel_profiles__inet.0
    Flexible IPv6 VXLAN tunnel profile
        Action: Decapsulate
        Interface: fti0.6 (Index: 10921)
        VNI: 16777213
        Source Prefix: 2a01:13:80:1:1:1::/96
        Source UDP Port Range: 49152 - 65535
        Destination Address: 2a01:10:255::2
        Destination UDP Port: 4789
        VXLAN Flags: 0x08
CUSTOMER_0001-decap-mvnet1_10
    Client ID: ABCD, Route Prefix: 1.0.0.9/32, Table: __flexible_tunnel_profiles__inet.0
    Flexible IPv6 VXLAN tunnel profile
        Action: Decapsulate
        Interface: fti0.9 (Index: 10918)
        VNI: 16777213
        Source Prefix: 2a01:13:80:1:1:10::/96
        Source UDP Port Range: 49152 - 65535
        Destination Address: 2a01:10:255::2
        Destination UDP Port: 4789
        VXLAN Flags: 0x08
CUSTOMER_0001-decap-mvnet1_11
    Client ID: ABCD, Route Prefix: 1.0.0.10/32, Table: __flexible_tunnel_profiles__inet.0
    Flexible IPv6 VXLAN tunnel profile

```

```

    Action: Decapsulate
    Interface: fti0.10 (Index: 10917)
    VNI: 16777213
    Source Prefix: 2a01:13:80:1:1:11::/96
    Source UDP Port Range: 49152 - 65535
    Destination Address: 2a01:10:255::2
    Destination UDP Port: 4789
    VXLAN Flags: 0x08
...
CUSTOMER_0001-decap-mvnet1_50001
  Client ID: ABCD, Route Prefix: 1.0.195.80/32, Table: __flexible_tunnel_profiles__.inet.0
  Flexible IPv6 VXLAN tunnel profile
    Action: Decapsulate
    Interface: fti0.6201 (Index: 10915)
    VNI: 16777214
    Source Prefix: 2a01:13:80:1:1:1::/96
    Source UDP Port Range: 49152 - 65535
    Destination Address: 2a01:10:255:1::2
    Destination UDP Port: 4789
    VXLAN Flags: 0x08
...

```

show flexible-tunnels profiles detail (to display GRE tunnel information)

```

user@host> show flexible-tunnels profiles detail
user@host> show flexible-tunnels profiles detail
decap-001
  Client ID: app1, Route Prefix: 10.0.0.1/32, Table: __flexible_tunnel_profiles__.inet.0
  Flexible IPv4 GRE tunnel profile
    Action: Decapsulate
    Source Prefix: 10.1.1.1
    Destination Address: 20.2.2.2
    GRE Key : 1

```

show flexible-tunnels profiles (Specific Profile, none and brief)

```

user@host> show flexible-tunnels profiles CUSTOMER_0001-decap-mvnet1_50001
user@host> show flexible-tunnels profiles CUSTOMER_0001-decap-mvnet1_50001
brief
CUSTOMER_0001-decap-mvnet1_50001

```



```
Client ID: ABCD, Route Prefix: 1.0.195.80/32, Table: __flexible_tunnel_profiles__.inet.0
Action: Decapsulate, Tunnel type: VXLAN-IPv6
```

show flexible-tunnels profiles detail (Specific Profile)

```
user@host> show flexible-tunnels profiles CUSTOMER_0001-decap-mvnet1_50001
detail
CUSTOMER_0001-decap-mvnet1_50001
Client ID: ABCD, Route Prefix: 1.0.195.80/32, Table: __flexible_tunnel_profiles__.inet.0
Flexible IPv6 VXLAN tunnel profile
  Action: Decapsulate
  Interface: fti0.6201 (Index: 10915)
  VNI: 16777214
  Source Prefix: 2a01:13:80:1:1:1::/96
  Source UDP Port Range: 49152 - 65535
  Destination Address: 2a01:10:255:1::2
  Destination UDP Port: 4789
  VXLAN Flags: 0x08
```

show flexible-tunnels profiles (Specific Route, none and brief)

```
user@host> show flexible-tunnels profiles route
1.0.195.80
user@host> show flexible-tunnels profiles route
1.0.195.80 brief
CUSTOMER_0001-decap-mvnet1_50001
Client ID: ABCD, Route Prefix: 1.0.195.80/32, Table: __flexible_tunnel_profiles__.inet.0
Action: Decapsulate, Tunnel type: VXLAN-IPv6
```

show flexible-tunnels profiles detail (Specific Route)

```
user@host> show flexible-tunnels profiles route
10.0.195.80 detail
CUSTOMER_0001-decap-mvnet1_50001
Client ID: ABCD, Route Prefix: 1.0.195.80/32, Table: __flexible_tunnel_profiles__.inet.0
Flexible IPv6 VXLAN tunnel profile
  Action: Decapsulate
  Interface: fti0.6201 (Index: 10915)
```

```
VNI: 16777214
Source Prefix: 2a01:13:80:1:1:1::/96
Source UDP Port Range: 49152 - 65535
Destination Address: 2a01:10:255:1::2
Destination UDP Port: 4789
VXLAN Flags: 0x08
```

Release Information

Command introduced in Junos OS Release 19.1.

RELATED DOCUMENTATION

[Understanding Programmable Flexible VXLAN Tunnels](#) | 1529

show security flow tunnel inspection statistics

IN THIS SECTION

- [Syntax](#) | 2039
- [Description](#) | 2039
- [Required Privilege Level](#) | 2040
- [Output Fields](#) | 2040
- [Sample Output](#) | 2041
- [Release Information](#) | 2041

Syntax

```
show security flow tunnel-inspection statistics
```

Description

Required Privilege Level

Output Fields

Table 1 lists the output fields for the show security flow tunnel-inspection statistics Output Fields command.

Table 115: show security flow tunnel-inspection statistics Output Fields

Field Name	Field Description
Tunnel inspection type	Tunnel inspection type. Displays VXLAN for VXLAN tunnel traffic
overlay session active	Number of active overlay sessions
overlay session create	Number of overlay sessions created
overlay session close	Number of overlay sessions closed
underlay session active	Number of active underlay sessions
underlay session create	Number of underlay sessions created
underlay session close	Number of underlay sessions closed
input packets	Total number of input packets
input bytes	Total number of input Bytes
output packets	Total number of output packets
output bytes	Total number of output bytes

Table 115: show security flow tunnel-inspection statistics Output Fields *(Continued)*

Field Name	Field Description
bypass packets	Number of packets bypassing security policy
bypass bytes	Number of packets bypassing security policy

Sample Output

show security flow tunnel-inspection statistics

```
user@host> show security flow tunnel-inspection statistics
```

```
Flow Tunnel-inspection statistics:
```

```
Tunnel-inspection type VXLAN:
```

```
  overlay session active:      0
  overlay session create:     275
  overlay session close:      275
  underlay session active:     0
  underlay session create:    615
  underlay session close:     615
  input packets:              0
  input bytes:                0
  output packets:             0
  output bytes:               0
  bypass packets:             0
  bypass bytes:               0
```

Release Information

Command introduced in Junos OS Release 21.1R1.

RELATED DOCUMENTATION

[Tunnel Inspection for EVPN-VXLAN by SRX Series Devices](#) | 834

show security policies policy set

IN THIS SECTION

- [Syntax | 2042](#)
- [Description | 2042](#)
- [Required Privilege Level | 2042](#)
- [Output Fields | 2042](#)
- [Sample Output | 2044](#)
- [Release Information | 2044](#)

Syntax

```
show security policies policy-set
```

Description

Displays a summary of all policy set you have created for inner session tunnel inspection for EVPN-VXLAN tunnel traffic.

Required Privilege Level

view

Output Fields

["No Link Title" on page 2042](#) lists the output fields for the show security policies command. Output fields are listed in the approximate order in which they appear.

show security policies policy-set Output Fields

Field Name	Field Description
From zone	Name of the source zone.

Field Name	Field Description
To zone	Name of the destination zone.
Policy-name	Name of the policy-set
State	<p>Status of the policy:</p> <ul style="list-style-type: none"> enabled: The policy can be used in the policy lookup process, which determines access rights for a packet and the action taken in regard to it. disabled: The policy cannot be used in the policy lookup process, and therefore it is not available for access control.
Index	Internal number associated with the policy.
Sequence number	Number of the policy within a given context. For example, three policies that are applicable in a from-zoneA-to-zoneB context might be ordered with sequence numbers 1, 2, 3. Also, in a from-zoneC-to-zoneD context, four policies might have sequence numbers 1, 2, 3, 4.
Scope Policy	Policy identifier.
Log Profile ID	Internal log profile number.
from-zone	source zone of the traffic.
to-zone	Destination zone of the traffic.
Source vrf group	Source virtual routing and forwarding (VRF). One or many source VRF instances, for example, the VRF routing instance associated with an incoming packet
Destination vrf group	Destination (VRF) . One or many destination VRF instances.
Source address	For standard display mode, the names of the source addresses for a policy. Address sets are resolved to their individual names.
Destination Address	Name of the destination address (or address set) as it was entered in the destination zone's address book.
Application	Name of a preconfigured or custom application whose type the packet matches, as specified at configuration time.
Source identity feeds	One or more user roles specified for a policy.
Destination identity feeds	One or more user roles specified for a policy.

Field Name	Field Description
Action	The action taken for a packet that matches the policy's tuples. Actions include : deny reject permit

Sample Output

show security policies policy-set

```
user@host> show security policies policy-set
```

```
From zone: PSET-1, To zone: PSET-1
  Policy: PSET-1-P1, State: enabled, Index: 5, Scope Policy: 0, Sequence number: 1, Log Profile
  ID: 0
    From zones: any
    To zones: any
    Source vrf group: any
    Destination vrf group: any
    Source addresses: a2-untrust
    Destination addresses: a2-trust
    Applications: any
    Source identity feeds: any
    Destination identity feeds: any
    Action: permit, application services
```

Release Information

Command introduced in Junos OS Release 21.1R1.

RELATED DOCUMENTATION

| [Tunnel Inspection for EVPN-VXLAN by SRX Series Devices](#) | 834

show security tunnel inspection

IN THIS SECTION

- [Syntax | 2045](#)
- [Description | 2045](#)
- [Required Privilege Level | 2045](#)
- [Output Fields | 2046](#)
- [Sample Output | 2046](#)
- [Release Information | 2047](#)

Syntax

```
show security tunnel-inspection profiles
```

```
<profiles>
```

```
<vnis>
```

Description

Display details of the tunnel inspection profile created for EVPN-VXLAN tunnel traffic.

Required Privilege Level

View

Output Fields

Table 1 lists the output fields for the show security tunnel-inspection profiles command.

Table 116: show security tunnel-inspection profiles Output Fields

Field Name	Field Description
Profile count	Number of tunnel inspection profiles.
Profile	Tunnel inspection profile name
Type	Type of tunnel inspection profile. Displays VXLAN for the profile created to inspect EVPN-VXLAN tunnel traffic.
Vxlan count	Number of associated VXLAN identifiers.
Vxlan name	VXLAN tunnel identifier used to uniquely identify for security inspection.
VNI count	Number of virtual network identifier (VNI) associated.
VNI	Name of the VNI.
VNI id count	VNI range.
Policy set	Associated policy set for inner session inspection.
Inspection level	Level of inspection.

Sample Output

show security tunnel-inspection profiles

```
user@host> show security tunnel-inspection profiles

Logical system: root-logical-system
  Profile count: 1
```

```
Profile: TP-1
Type: VXLAN
Vxlan count: 1
Vxlan name: VXT-1
VNI count: 1
VNI:VNI-1
Policy set: PSET-1
Inspection level: 1
```

show security tunnel-inspection vnis

```
user@host> show security tunnel-inspection vnis
```

```
Logical system: root-logical-system
VNI count: 1
VNI name: VNI-1
VNI id count: 1
[50 - 100]
```

Release Information

Command introduced in Junos OS Release 21.1R1.

RELATED DOCUMENTATION

[Tunnel Inspection for EVPN-VXLAN by SRX Series Devices](#) | 834