Junos® OS

EVPNs Feature Guide for Routing Devices

Release
14.1

Published: 2014-06-12

Juniper Networks, Inc.
1194 North Mathilda Avenue
Sunnyvale, California 94089
USA
408-745-2000
www.juniper.net

Junos® OS EVPNs Feature Guide for Routing Devices
14.1

END USER LICENSE AGREEMENT

Table of Contents

List of Figures

List of Tables

About the Documentation

Documentation and Release Notes

To obtain the most current version of all Juniper Networks® technical documentation, see the product documentation page on the Juniper Networks website at http://www.juniper.net/techpubs/.

If the information in the latest release notes differs from the information in the documentation, follow the product Release Notes.

Juniper Networks Books publishes books by Juniper Networks engineers and subject matter experts. These books go beyond the technical documentation to explore the nuances of network architecture, deployment, and administration. The current list can be viewed at http://www.juniper.net/books.

Supported Platforms

For the features described in this document, the following platforms are supported:

- MX Series

Using the Examples in This Manual

If you want to use the examples in this manual, you can use the **load merge** or the **load merge relative** command. These commands cause the software to merge the incoming configuration into the current candidate configuration. The example does not become active until you commit the candidate configuration.

If the example configuration contains the top level of the hierarchy (or multiple hierarchies), the example is a *full example*. In this case, use the **load merge** command.

If the example configuration does not start at the top level of the hierarchy, the example is a *snippet*. In this case, use the **load merge relative** command. These procedures are described in the following sections.

Merging a Full Example

To merge a full example, follow these steps:

1. From the HTML or PDF version of the manual, copy a configuration example into a text file, save the file with a name, and copy the file to a directory on your routing platform.

   For example, copy the following configuration to a file and name the file **ex-script.conf**. Copy the **ex-script.conf** file to the **/var/tmp** directory on your routing platform.

   ```
   system {
     scripts {
       commit {
         file ex-script.xsl;
       }
     }
   }
   interfaces {
     fxp0 {
       disable;
       unit 0 {
         family inet {
           address 10.0.0.1/24;
         }
       }
     }
   }
   ```

2. Merge the contents of the file into your routing platform configuration by issuing the **load merge** configuration mode command:

   ```
   [edit]
   user@host# load merge /var/tmp/ex-script.conf
   load complete
   ```

Merging a Snippet

To merge a snippet, follow these steps:

1. From the HTML or PDF version of the manual, copy a configuration snippet into a text file, save the file with a name, and copy the file to a directory on your routing platform.

   For example, copy the following snippet to a file and name the file **ex-script-snippet.conf**. Copy the **ex-script-snippet.conf** file to the **/var/tmp** directory on your routing platform.

   ```
   commit {
     file ex-script-snippet.xsl; }
   ```

2. Move to the hierarchy level that is relevant for this snippet by issuing the following configuration mode command:


```
[edit]
user@host# edit system scripts
[edit system scripts]
```

3. Merge the contents of the file into your routing platform configuration by issuing the **load merge relative** configuration mode command:

```
[edit system scripts]
user@host# load merge relative /var/tmp/ex-script-snippet.conf
load complete
```

For more information about the **load** command, see the *CLI User Guide*.

Documentation Conventions

Table 1 on page xiii defines notice icons used in this guide.

Table 1: Notice Icons

Icon	Meaning	Description
	Informational note	Indicates important features or instructions.
	Caution	Indicates a situation that might result in loss of data or hardware damage.
	Warning	Alerts you to the risk of personal injury or death.
	Laser warning	Alerts you to the risk of personal injury from a laser.
	Tip	Indicates helpful information.
	Best practice	Alerts you to a recommended use or implementation.

Table 2 on page xiii defines the text and syntax conventions used in this guide.

Table 2: Text and Syntax Conventions

Convention	Description	Examples
Bold text like this	Represents text that you type.	To enter configuration mode, type the **configure** command: user@host> **configure**

Table 2: Text and Syntax Conventions *(continued)*

Convention	Description	Examples
`Fixed-width text like this`	Represents output that appears on the terminal screen.	`user@host>` **show chassis alarms** `No alarms currently active`
Italic text like this	• Introduces or emphasizes important new terms. • Identifies guide names. • Identifies RFC and Internet draft titles.	• A policy *term* is a named structure that defines match conditions and actions. • *Junos OS CLI User Guide* • RFC 1997, *BGP Communities Attribute*
Italic text like this	Represents variables (options for which you substitute a value) in commands or configuration statements.	Configure the machine's domain name: [edit] root@# **set system domain-name** *domain-name*
Text like this	Represents names of configuration statements, commands, files, and directories; configuration hierarchy levels; or labels on routing platform components.	• To configure a stub area, include the **stub** statement at the **[edit protocols ospf area area-id]** hierarchy level. • The console port is labeled **CONSOLE**.
< > (angle brackets)	Encloses optional keywords or variables.	**stub <default-metric *metric*>;**
| (pipe symbol)	Indicates a choice between the mutually exclusive keywords or variables on either side of the symbol. The set of choices is often enclosed in parentheses for clarity.	**broadcast | multicast** **(*string1* | *string2* | *string3*)**
# (pound sign)	Indicates a comment specified on the same line as the configuration statement to which it applies.	**rsvp { # Required for dynamic MPLS only**
[ ] (square brackets)	Encloses a variable for which you can substitute one or more values.	**community name members [*community-ids*]**

Indention and braces ({ })	Identifies a level in the configuration hierarchy.	[edit]
		routing-options {
		static {
		route default {
		nexthop *address*;
		retain;
		}
		}
		}

; (semicolon)	Identifies a leaf statement at a configuration hierarchy level.	
GUI Conventions		
Bold text like this	Represents graphical user interface (GUI) items you click or select.	• In the Logical Interfaces box, select **All Interfaces**. • To cancel the configuration, click **Cancel**.

Table 2: Text and Syntax Conventions *(continued)*

Convention	Description	Examples
> (bold right angle bracket)	Separates levels in a hierarchy of menu selections.	In the configuration editor hierarchy, select **Protocols>Ospf**.

Documentation Feedback

We encourage you to provide feedback, comments, and suggestions so that we can improve the documentation. You can provide feedback by using either of the following methods:

- Online feedback rating system—On any page at the Juniper Networks Technical Documentation site at http://www.juniper.net/techpubs/index.html, simply click the stars to rate the content, and use the pop-up form to provide us with information about your experience. Alternately, you can use the online feedback form at https://www.juniper.net/cgi-bin/docbugreport/.
- E-mail—Send your comments to techpubs-comments@juniper.net. Include the document or topic name, URL or page number, and software version (if applicable).

Requesting Technical Support

Technical product support is available through the Juniper Networks Technical Assistance Center (JTAC). If you are a customer with an active J-Care or JNASC support contract, or are covered under warranty, and need post-sales technical support, you can access our tools and resources online or open a case with JTAC.

- JTAC policies—For a complete understanding of our JTAC procedures and policies, review the *JTAC User Guide* located at http://www.juniper.net/us/en/local/pdf/resource-guides/7100059-en.pdf.
- Product warranties—For product warranty information, visit http://www.juniper.net/support/warranty/.
- JTAC hours of operation—The JTAC centers have resources available 24 hours a day, 7 days a week, 365 days a year.

Self-Help Online Tools and Resources

For quick and easy problem resolution, Juniper Networks has designed an online self-service portal called the Customer Support Center (CSC) that provides you with the following features:

- Find CSC offerings: http://www.juniper.net/customers/support/
- Search for known bugs: http://www2.juniper.net/kb/
- Find product documentation: http://www.juniper.net/techpubs/
- Find solutions and answer questions using our Knowledge Base: http://kb.juniper.net/

- Download the latest versions of software and review release notes: http://www.juniper.net/customers/csc/software/
- Search technical bulletins for relevant hardware and software notifications: http://kb.juniper.net/InfoCenter/
- Join and participate in the Juniper Networks Community Forum: http://www.juniper.net/company/communities/
- Open a case online in the CSC Case Management tool: http://www.juniper.net/cm/

To verify service entitlement by product serial number, use our Serial Number Entitlement (SNE) Tool: https://tools.juniper.net/SerialNumberEntitlementSearch/

Opening a Case with JTAC

You can open a case with JTAC on the Web or by telephone.

- Use the Case Management tool in the CSC at http://www.juniper.net/cm/.
- Call 1-888-314-JTAC (1-888-314-5822 toll-free in the USA, Canada, and Mexico).

For international or direct-dial options in countries without toll-free numbers, see http://www.juniper.net/support/requesting-support.html.

PART 1

Overview

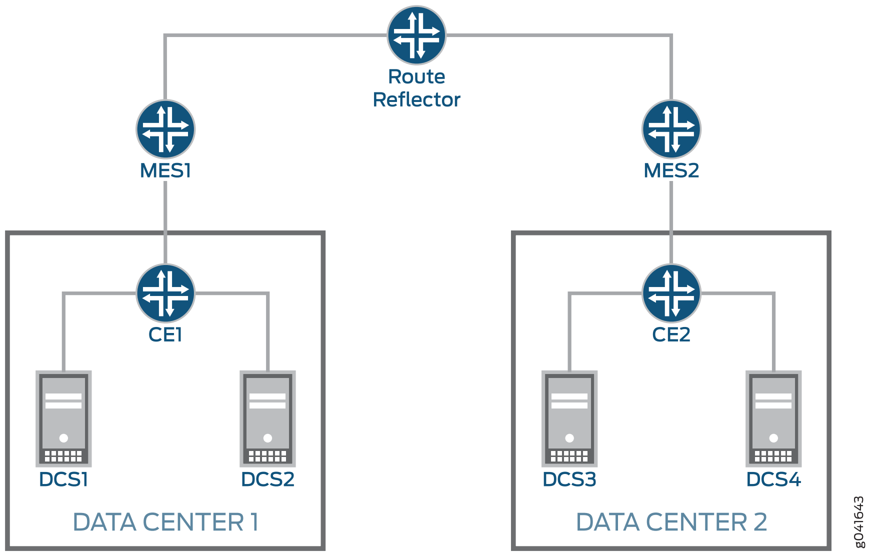
CHAPTER 1

Introduction to EVPNs

EVPN Overview

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 is comprised of 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, an MX Series 3D Universal Edge Router can be configured to act as an MES. You can deploy multiple EVPNs within the service provider's network, each providing network connectivity to customers while ensuring that the traffic sharing that network remains private. Figure 1 on page 4 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 1: 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 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 VLAN for each MAC VPN

- Automatic route distinguishers
- Dual homed EVPN connection with Active Standby multihoming

The following features are not supported for EVPNs:

- Graceful restart, graceful Routing Engine switchover (GRES), and nonstop active routing (NSR)
- Active Active multihoming

Related Documentation

- Supported EVPN Standards on page 23

EVPN with IRB Solution Overview

A Data Center Service Provider (DCSP) hosts the data center for its multiple customers onto 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 to more than one site. In order 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) motion.
- 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 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 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.

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.

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.

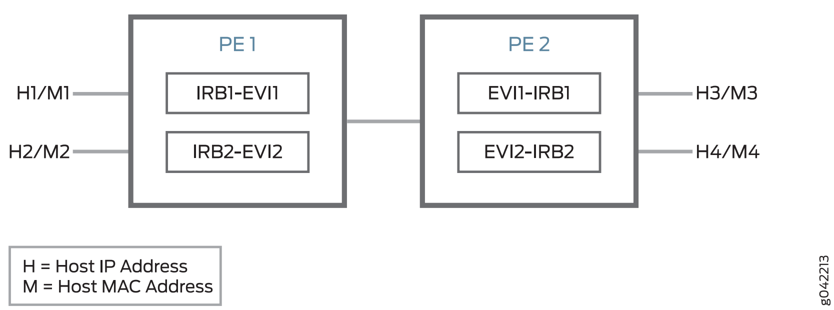
- Gateway MAC-IP synchronization

  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 2 on page 7 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 2: 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.

- Gateway MAC and IP Synchronization on page 8
- Layer 3 VPN Interworking on page 8

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

• Example: Configuring EVPN with IRB Solution on page 41

EVPN Multihoming Overview

Introduction to EVPN Multihoming

An Ethernet VPN (EVPN) is comprised of customer edge (CE) devices that are connected to provider edge (PE) routers, which form the edge of the MPLS infrastructure. A CE device can be a host, a router, or a switch. The PE routers 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.

The EVPN multihoming feature enables you to connect a customer site to two or more PE routers to provide redundant connectivity. A CE device can be multihomed to different PE routers or the same PE router. A redundant PE router 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 router to CE device link failure
- PE router failure
- MPLS-reachability failure between the local PE router and a remote PE router

Figure 3: CE Device Multihomed to Two PE Routers

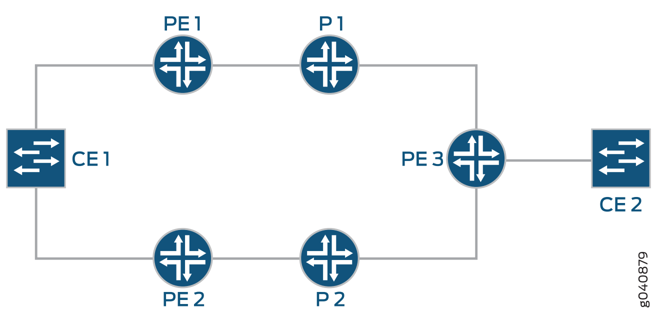


Figure 3 on page 10 illustrates how a CE device could be multihomed to two PE routers. Device CE1 is multihomed to Routers PE1 and PE2. Device CE2 has two potential paths to reach Device CE1, but only one path is active at any one time. If Router PE1 were the designated PE router to forward traffic to the CE device (also called a designated forwarder), PE1 forwards traffic to Device CE1 using MPLS LSP or GRE tunnels. If a failure occurs over this path, a new designated forwarder is elected to forward the traffic to Device CE1.

Understanding EVPN Multihoming Concepts

Figure 4 on page 12 explains the EVPN multihoming concepts using a simple EVPN network 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.

- **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 a single-homed CE device is attached to a Ethernet segment, the 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 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 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.

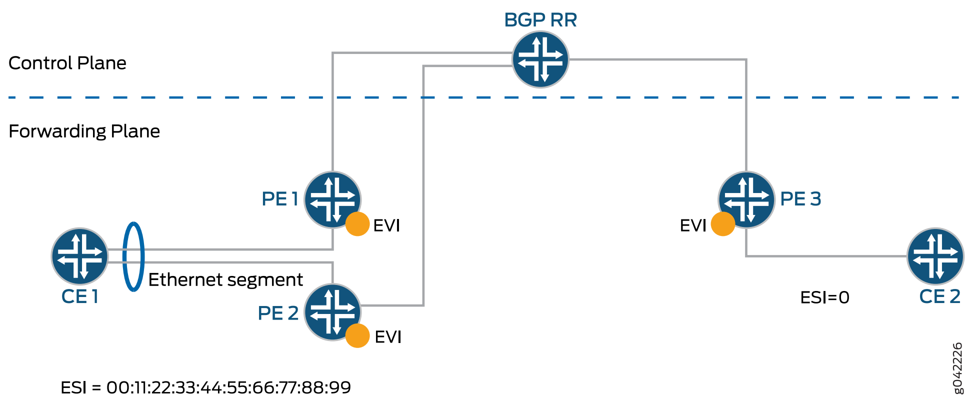
  Routers PE1 and PE2 advertise an ES route with an ES-import extended community (along with other extended communities like the route target). The PE routers also construct a filter based on an ES-import extended community, which results in only these PE routers importing the ES route and identifying 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 eight-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, one of the PE routers is used to reach the customer site. 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 (DF) election to select the DF and the BDF PE routers.

Figure 4: Simple EVPN Topology



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. This 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 **single-active** mode under interface configuration.
- 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 an active-active redundancy mode.

NOTE: Junos OS supports only the active-standby mode of operation for EVPN multihoming. The active-active mode of operation is not supported.

Active-Standby 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. The active-standby multihoming feature is based on the EVPN *draft-ietf-l2vpn-evpn-03*.

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

New BGP NLRIs

To support active-standby EVPN multihoming, two new BGP network layer reachability information (NLRI) routes have been introduced:

- Autodiscovery route per Ethernet segment

  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.

  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.

- Ethernet segment route

  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.

  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.

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, two new extended communities have been introduced:

- ESI-import extended community—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 should be configured to attach all the route targets to the autodiscovery route per Ethernet segment.**

- Split horizon extended community—This 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.

New EVPN Route Types

Active-standby multihoming mode supports the following EVPN route types:

- Autodiscovery route per Ethernet segment
- Ethernet segment route

These route types conform to the following naming convention:

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

For example:

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

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.
  - 4—Ethernet segment route.
- **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—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.

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 active-standby multihoming mode, there are two types of traffic flows:

- Unicast

  Unicast traffic is a point-to-point communication with one sender and one receiver. In a multihomed EVPN, unicast traffic is forwarded as follows:

  a. CE to core—Traffic is learned and forwarded by the DF PE router.

b. Core to CE—The remote PE router learns the MAC addresses from the DF, and forwards all unicast traffic to the DF PE router.

- BUM

  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:

  a. 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.

  b. 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.

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 this Ethernet segment.

Designated Forwarder Election

The following sections discuss DF election:

DF Election Roles

The designated forwarder (DF) election process involves selecting the following two 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).

  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 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 to 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.

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.

Handling Failover

A failover can happen due to two things:

- When the DF PE router loses its DF role.
- When 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, flow of traffic is affected as follows:

- Unicast Traffic on page 20
- BUM Traffic on page 20

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.

Related Documentation

- Example: Configuring EVPN with Multihoming Support on page 66

Virtual Switch Support for EVPN Overview

A Data Center Service Provider (DCSP) hosts the data center for its multiple customers onto 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 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) motion.
- Supporting multiple tenants with independent VLAN and subnet space.

Ethernet VPN is targeted to handle all of the above 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

Because each data center hosts multiple tenants with independent VLAN and private subnet space, if an EVPN instance (EVI) was to stretch just one VLAN, it would be a severe problem to scale the data center services. In order to overcome this limitation, starting with Junos OS Release 14.1, a new service called the VLAN-aware bundling service is introduced. This feature 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.

Junos OS has a very flexible and scalable virtual switch interface. With virtual switch a single MX Series router can be divided into multiple logical switches. Layer 2 domains (also called bridge-domains) can be defined independently in each virtual switch. To achieve the VLAN-aware bundling service, an EVPN would be allowed to run in a virtual-switch routing instance.

A single EVPN instance can stretch up to 4094 bridge domains defined in a virtual switch to remote sites. A virtual switch can have more than 4094 bridge domains with a combination of none, single, and dual VLANs. However, because EVPN signaling deals only with single VLAN tags, only a maximum of 4094 bridge domains can be stretched. The EVPN virtual switch also provides support for trunk and access interfaces.

NOTE:

- The none VLAN option is not supported with bridge domains under the virtual switch instance type for EVPNs.
- Dual VLANs are not supported with EVPN although they can be configured.

There are two types of VLAN-aware bundling services:

- VLAN-aware bundling without translation

  The service interface provides bundling of customer VLANs into a single Layer 2 VPN service instance with a guarantee for end-to-end customer VLAN transparency. The data-plane separation between the customer VLANs is maintained by creating a dedicated bridge-domain per VLAN.

- VLAN-aware bundling 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 per 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 bundling with translation only.

Related Documentation

- Example: Configuring EVPN with Support for Virtual Switch on page 98

CHAPTER 2

EVPN Standards

Supported EVPN Standards

Junos OS substantially supports the following 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*
- Internet draft draft-ietf-l2vpn-evpn-00.txt, *BGP MPLS Based Ethernet VPN*

Related Documentation

- EVPN Overview on page 3

- *Accessing Standards Documents on the Internet*

PART 2

Configuration

CHAPTER 3

Configuring EVPNs

Configuring an IGP on the PE and P Routers

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

- *Example: Configuring IS-IS*
- *Examples: Configuring Static Routes*
- *OSPF Feature Guide for Routing Devices*

Configuring IBGP Sessions Between PE Routers in VPNs

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. If you are using RSVP signaling, this IP address is the same address you specify in the **to** statement at the **[edit mpls label-switched-path *lsp-path-name*]** hierarchy level when you configure the MPLS LSP.

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 an IGP on the PE and P Routers on page 27
- Configuring a Signaling Protocol and LSPs for VPNs on page 29

Configuring a Signaling Protocol and LSPs for VPNs

For VPNs to function, you must enable a signaling protocol, either the LDP or RSVP on the provider edge (PE) routers and on the provider (P) routers. You also need to configure label-switched paths (LSPs) between the ingress and egress routers. In a typical VPN configuration, you need to configure LSPs from each PE router to all of the other PE routers participating in the VPN in a full mesh.

NOTE: As with any configuration involving MPLS, you cannot configure any of the core-facing interfaces on the PE routers over dense Fast Ethernet PICs.

To enable a signaling protocol, perform the steps in one of the following sections:

- Using LDP for VPN Signaling on page 29
- Using RSVP for VPN Signaling on page 31

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 service provider's 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) 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 Feature Guide for Routing Devices* and *IS-IS Feature Guide for Routing Devices*.

Using RSVP for VPN Signaling

To use RSVP for VPN signaling, perform the following steps:

1. On each PE router, configure traffic engineering.

   To do this, you must configure an interior gateway protocol (IGP) that supports traffic engineering (either IS-IS or OSPF) and enable traffic engineering support for that protocol.

   To enable OSPF traffic engineering support, include the **traffic-engineering** statement at the **[edit protocols ospf]** hierarchy level:

   ```
   [edit protocols ospf]
   traffic-engineering {
      shortcuts;
   }
   ```

   For IS-IS, traffic engineering support is enabled by default.

2. On each PE and P router, enable RSVP on the interfaces that participate in the label-switched path (LSP).

   On the PE router, these interfaces are the ingress and egress points to the LSP. On the P router, these interfaces connect the LSP between the PE routers. Do not enable RSVP on the interface between the PE and the CE routers, because this interface is not part of the LSP.

   To configure RSVP on the PE and P routers, include the **interface** statement at the **[edit protocols rsvp]** hierarchy level. Include one **interface** statement for each interface on which you are enabling RSVP.

   ```
   [edit protocols]
   rsvp {
      interface interface-name;
      interface interface-name;
   }
   ```

3. On each PE router, configure an MPLS LSP to the PE router that is the LSP's egress point.

   To do this, include the **interface** and **label-switched-path** statements at the **[edit protocols mpls]** hierarchy level:

   ```
   [edit protocols]
   mpls {
      interface interface-name;
      label-switched-path path-name {
   ```

```
    to ip-address;
  }
}
```

In the **to** statement, specify the address of the LSP's egress point, which is an address on the remote PE router.

In the **interface** statement, specify the name of the interface (both the physical and logical portions). Include one **interface** statement for the interface associated with the LSP.

When you configure the logical portion of the same interface at the **[edit interfaces]** hierarchy level, you must also configure the **family inet** and **family mpls** statements:

```
[edit interfaces]
interface-name {
  unit logical-unit-number {
    family inet;
    family mpls;
  }
}
```

4. On all P routers that participate in the LSP, enable MPLS by including the **interface** statement at the **[edit mpls]** hierarchy level.

   Include one **interface** statement for each connection to the LSP.

```
[edit]
mpls {
  interface interface-name;
  interface interface-name;
}
```

5. Enable MPLS on the interface between the PE and CE routers by including the **interface** statement at the **[edit mpls]** hierarchy level.

   Doing this allows the PE router to assign an MPLS label to traffic entering the LSP or to remove the label from traffic exiting the LSP.

```
[edit]
mpls {
  interface interface-name;
}
```

   For information about configuring MPLS, see the *Minimum MPLS Configuration*.

Related Documentation

- *Minimum MPLS Configuration*9

Configuring EVPN Routing Instances

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 **routing-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:

   a. Configure each interface using the interface statement at the **[edit routing-instances *routing-instance-name* protocols** evpn**]** hierarchy level.

   b. (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.

   c. (Optional) 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 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 on page 35

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 on page 33
- **traceoptions on page 139**

Example: Configuring EVPN with IRB Solution

EVPN with IRB Solution Overview

A Data Center Service Provider (DCSP) hosts the data center for its multiple customers onto 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 to more than one site. In order 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) motion.
- 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 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 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.

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.

There are two major functions that are supported for IRB in EVPN.

- Host MAC-IP synchronization

  This includes:

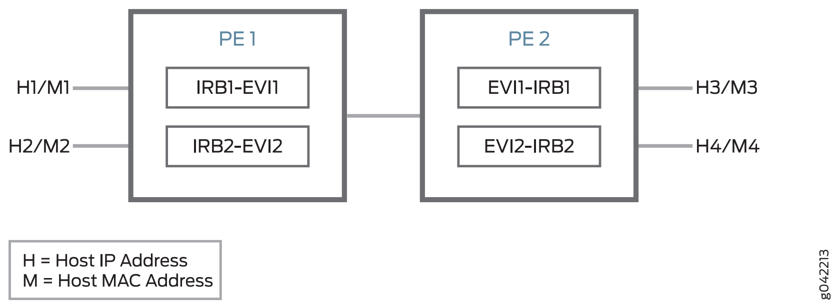
  - Advertising the IP address along with the MAC advertisement route in EVPN. This is done by using the IP field in the EVPN MAC advertisement route.
  - The receiving PE router installs MAC into the EVPN instance (EVI) table and installs IP into the associated VRF.
- Gateway MAC-IP synchronization

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 2 on page 7 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 5: 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.

- Gateway MAC and IP Synchronization on page 40
- Layer 3 VPN Interworking on page 40

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.

Example: Configuring EVPN with IRB Solution

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 3D Universal Edge Routers 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 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.

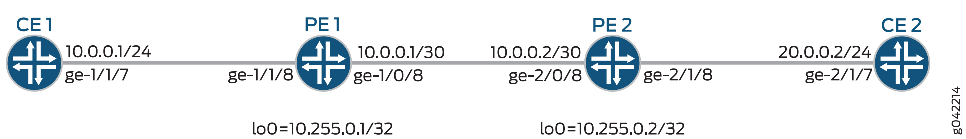
To support this flexibility and scalability factor, the EVPN solution provides support for the IRB interfaces on MX Series routers containing MPC FPCs to facilitate optimal Layer 2 and Layer 3 forwarding along with virtual machine mobility. The IRB interfaces are configured on each configured bridge 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 6 on page 42 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 6: EVPN with IRB Soultion



Configuration

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 20.0.0.0/24 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 100.255.0.1/32
set routing-options router-id 100.255.0.1
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 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 100.255.0.1:100
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 100.255.0.1:300
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 20.0.0.251/24
set interfaces lo0 unit 0 family inet address 100.255.0.2/32
set routing-options router-id 100.255.0.2
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 100.100.100.2
set protocols mpls label-switched-path PE2-to-PE1 to 100.100.100.1
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 100.100.100.2
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 100.100.100.1
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 routing-interface irb.0
set routing-instances evpna route-distinguisher 100.255.0.2:100
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 100.255.0.2:300
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 20.0.0.2/24
set routing-options static route 10.0.0.0/24 next-hop 20.0.0.252
```

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 100.255.0.1/32
```

2. Set the router ID and autonomous system number for Router PE1.

```
[edit routing-options]
user@PE1# set router-id 100.255.0.1
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. Configure the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpna instance-type evpn
```

12. Set the VLAN identifier for the bridging domain in the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna vlan-id 10
```

13. Configure the interface name for the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna interface ge-1/1/8.0
```

14. Configure the IRB interface as the routing interface for the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna routing-interface irb.0
```

15. Configure the route distinguisher for the evpna routing instance.


```
[edit routing-instances]
user@PE1# set evpna route-distinguisher 100.255.0.1:100
```

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:100:100
```

17. 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
```

18. Configure the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf instance-type vrf
```

19. Configure the IRB interface as the routing interface for the vrf routing instance.

```
[edit routing-instances]
user@PE1# set vrf interface irb.0
```

20. Configure the route distinguisher for the vrf routing instance.

```
[edit routing-instances]
user@PE1# set vrf route-distinguisher 100.255.0.1:300
```

21. 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 100.255.0.1/32 {
            }
        }
    }
}
```

```
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 {
    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 100.255.0.1:100;
  vrf-target target:100:100;
  protocols {
    evpn {
      interface ge-1/1/8.0;
    }
  }
}
vrf {
  instance-type vrf;
  interface irb.0;
  route-distinguisher 100.255.0.1:300;
  vrf-target target:100:300;
  vrf-table-label;
}
```

Verification

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.

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 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 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 20.0.0.2 interval 0.1 count 10
PING 20.0.0.2 (20.0.0.2): 56 data bytes
64 bytes from 20.0.0.2: icmp_seq=0 ttl=63 time=0.919 ms
64 bytes from 20.0.0.2: icmp_seq=1 ttl=63 time=0.727 ms
64 bytes from 20.0.0.2: icmp_seq=2 ttl=63 time=0.671 ms
64 bytes from 20.0.0.2: icmp_seq=3 ttl=63 time=0.671 ms
64 bytes from 20.0.0.2: icmp_seq=4 ttl=63 time=0.666 ms
64 bytes from 20.0.0.2: icmp_seq=5 ttl=63 time=0.704 ms
64 bytes from 20.0.0.2: icmp_seq=6 ttl=63 time=0.763 ms
64 bytes from 20.0.0.2: icmp_seq=7 ttl=63 time=0.750 ms
64 bytes from 20.0.0.2: icmp_seq=8 ttl=63 time=12.967 ms
64 bytes from 20.0.0.2: icmp_seq=9 ttl=63 time=0.752 ms

--- 20.0.0.2 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.

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 20.0.0.252 interval 0.1 count 10
PING 20.0.0.252 (20.0.0.252): 56 data bytes
64 bytes from 20.0.0.252: icmp_seq=0 ttl=64 time=0.959 ms
64 bytes from 20.0.0.252: icmp_seq=1 ttl=64 time=0.710 ms
64 bytes from 20.0.0.252: icmp_seq=2 ttl=64 time=0.832 ms
64 bytes from 20.0.0.252: icmp_seq=3 ttl=64 time=0.754 ms
64 bytes from 20.0.0.252: icmp_seq=4 ttl=64 time=3.642 ms
64 bytes from 20.0.0.252: icmp_seq=5 ttl=64 time=0.660 ms
64 bytes from 20.0.0.252: icmp_seq=6 ttl=64 time=0.728 ms
64 bytes from 20.0.0.252: icmp_seq=7 ttl=64 time=0.725 ms
64 bytes from 20.0.0.252: icmp_seq=8 ttl=64 time=0.674 ms
64 bytes from 20.0.0.252: icmp_seq=9 ttl=64 time=0.760 ms

--- 20.0.0.252 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 evpna.inet.0 20.0.0.252** command.

```
user@PE1> show route table evpna.inet.0 20.0.0.252
evpna.inet.0: 8 destinations, 8 routes (8 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

20.0.0.252/32      *[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 20.0.0.252 interval 0.1 count 10
PING 20.0.0.252 (20.0.0.252): 56 data bytes
64 bytes from 20.0.0.252: icmp_seq=0 ttl=64 time=9.613 ms
64 bytes from 20.0.0.252: icmp_seq=1 ttl=64 time=0.789 ms
64 bytes from 20.0.0.252: icmp_seq=2 ttl=64 time=0.803 ms
64 bytes from 20.0.0.252: icmp_seq=3 ttl=64 time=0.695 ms
64 bytes from 20.0.0.252: icmp_seq=4 ttl=64 time=0.742 ms
64 bytes from 20.0.0.252: icmp_seq=5 ttl=64 time=0.702 ms
64 bytes from 20.0.0.252: icmp_seq=6 ttl=64 time=0.725 ms
64 bytes from 20.0.0.252: icmp_seq=7 ttl=64 time=0.730 ms
64 bytes from 20.0.0.252: icmp_seq=8 ttl=64 time=0.713 ms
64 bytes from 20.0.0.252: icmp_seq=9 ttl=64 time=7.370 ms

--- 20.0.0.252 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.

Example: Configuring EVPN Multihoming

EVPN Multihoming Overview

Introduction to EVPN Multihoming

An Ethernet VPN (EVPN) is comprised of customer edge (CE) devices that are connected to provider edge (PE) routers, which form the edge of the MPLS infrastructure. A CE device can be a host, a router, or a switch. The PE routers 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.

The EVPN multihoming feature enables you to connect a customer site to two or more PE routers to provide redundant connectivity. A CE device can be multihomed to different PE routers or the same PE router. A redundant PE router 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 router to CE device link failure
- PE router failure
- MPLS-reachability failure between the local PE router and a remote PE router

Figure 7: CE Device Multihomed to Two PE Routers

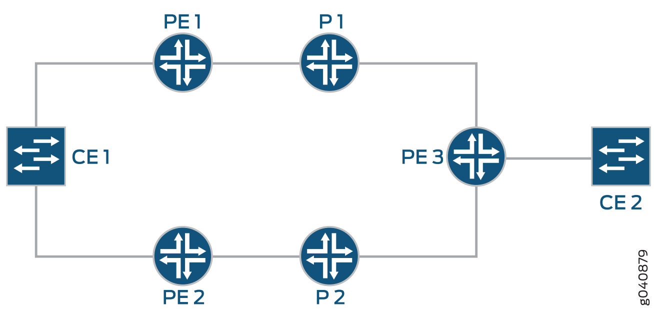


Figure 3 on page 10 illustrates how a CE device could be multihomed to two PE routers. Device CE1 is multihomed to Routers PE1 and PE2. Device CE2 has two potential paths to reach Device CE1, but only one path is active at any one time. If Router PE1 were the designated PE router to forward traffic to the CE device (also called a designated forwarder), PE1 forwards traffic to Device CE1 using MPLS LSP or GRE tunnels. If a failure occurs over this path, a new designated forwarder is elected to forward the traffic to Device CE1.

Understanding EVPN Multihoming Concepts

Figure 4 on page 12 explains the EVPN multihoming concepts using a simple EVPN network 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.

- **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 a single-homed CE device is attached to a Ethernet segment, the 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 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 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 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 eight-octet value divided into two main sections.

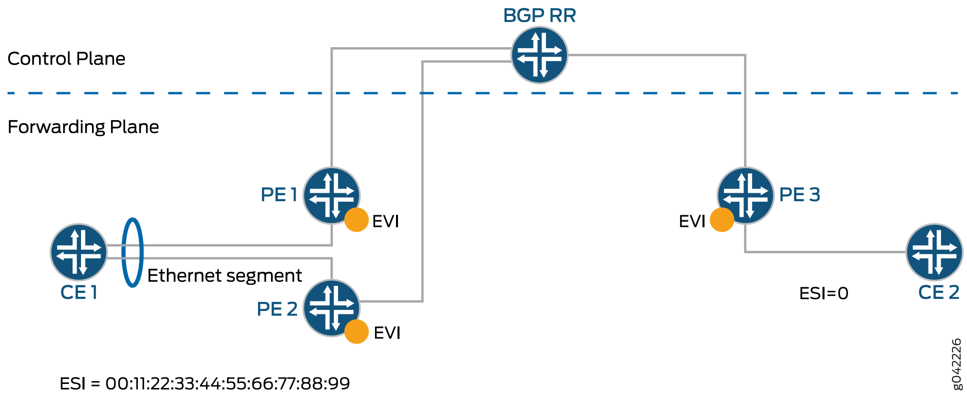
- **BUM traffic**—This type of traffic is sent to multiple destinations, including broadcast traffic, unknown unicast traffic that is broadcast in the Ethernet segment, and multicast traffic.

- **DF**—When a CE device is multihomed to two or more PE routers, one of the PE routers is used to reach the customer site. 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 (DF) election to select the DF and the BDF PE routers.

Figure 8: Simple EVPN Topology



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. This 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 **single-active** mode under interface configuration.
- 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 an active-active redundancy mode.

NOTE: **Junos OS supports only the active-standby mode of operation for EVPN multihoming. The active-active mode of operation is not supported.**

Active-Standby 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. The active-standby multihoming feature is based on the EVPN *draft-ietf-l2vpn-evpn-03*.

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

- New BGP NLRIs on page 59
- New Extended Communities on page 60

New BGP NLRIs

To support active-standby EVPN multihoming, two new BGP network layer reachability information (NLRI) routes have been introduced:

- Autodiscovery route per Ethernet segment

  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.

  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.

- Ethernet segment route

  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.

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.

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, two new extended communities have been introduced:

- ESI-import extended community—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 should be configured to attach all the route targets to the autodiscovery route per Ethernet segment.**

- Split horizon extended community—This 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.

New EVPN Route Types

Active-standby multihoming mode supports the following EVPN route types:

- Autodiscovery route per Ethernet segment
- Ethernet segment route

These route types conform to the following naming convention:

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

For example:

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

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.
  - 4—Ethernet segment route.
- **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—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.

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 active-standby multihoming mode, there are two types of traffic flows:

- Unicast

  Unicast traffic is a point-to-point communication with one sender and one receiver. In a multihomed EVPN, unicast traffic is forwarded as follows:

  a. CE to core—Traffic is learned and forwarded by the DF PE router.

  b. Core to CE—The remote PE router learns the MAC addresses from the DF, and forwards all unicast traffic to the DF PE router.

- BUM

  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:

  a. 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.

  b. 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.

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 Ehernet 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 this Ethernet segment.

Designated Forwarder Election

The following sections discuss DF election:

DF Election Roles

The designated forwarder (DF) election process involves selecting the following two 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).

  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 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 to 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.

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.

Handling Failover

A failover can happen due to two things:

- When the DF PE router loses its DF role.
- When 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, flow of traffic is affected as follows:

- Unicast Traffic on page 65
- BUM Traffic on page 66

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.

Example: Configuring EVPN with Multihoming Support

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 3D Universal Edge Routers with MPC interfaces only or an EX9200 Ethernet Switch, 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

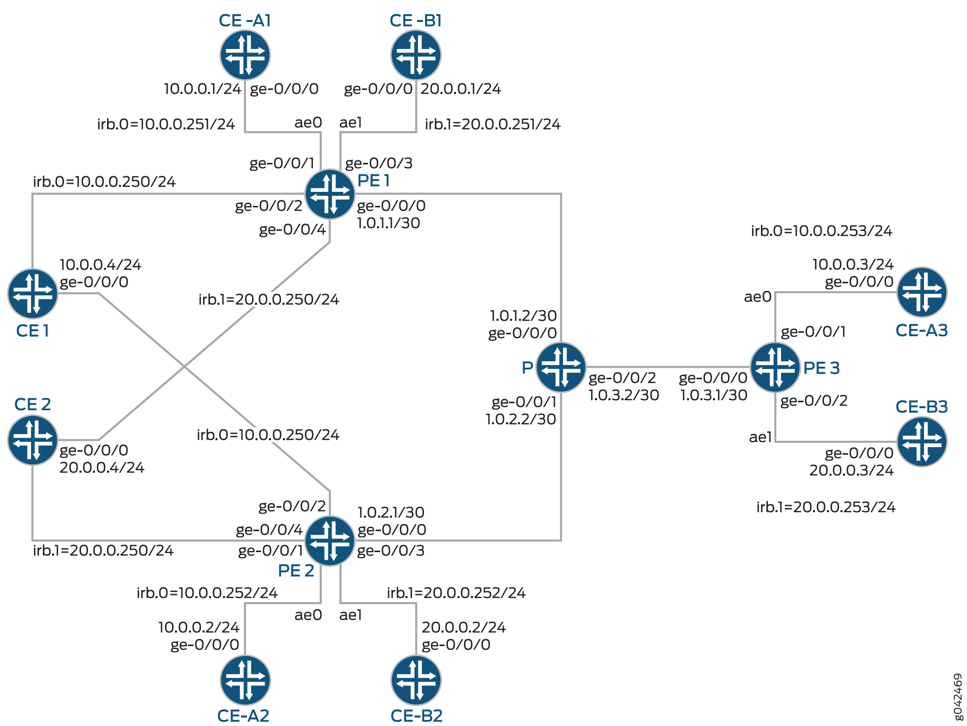
Starting with Junos OS Release 14.1, the EVPN solution on MX Series routers with MPC interfaces is extended to provide multihoming functionality with active-standby mode of operation. The multihoming functions include autodiscovery of Ethernet segments, Ethernet segment route construction, and Ethernet segment identifier (ESI) label assignment.

NOTE:

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-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.

Figure 9: EVPN Active-Standby Multihoming



In Figure 9 on page 68, Routers PE1 and PE2 are provider edge (PE) routers connected to multihomed customer edge (CE) devices, Device CE1 and CE2. Router PE3 is a remote PE router connected to a single-homed customer site, and 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.

Configuration

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-0/0/0 vlan-tagging
set interfaces ge-0/0/0 mac 00:00:00:00:00:04
set interfaces ge-0/0/0 unit 0 vlan-id 100
```

```
set interfaces ge-0/0/0 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 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 10.0.0.1/24
set routing-options static route 0.0.0.0/0 next-hop 10.0.0.251
```

CE-A2

```
set interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 mac 00:00:00:00:00:02
set interfaces ge-0/0/0 unit 0 vlan-id 100
set interfaces ge-0/0/0 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 interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 mac 00:00:00:00:00:03
set interfaces ge-0/0/0 unit 0 vlan-id 100
set interfaces ge-0/0/0 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 interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 mac 00:00:00:00:00:04
set interfaces ge-0/0/0 unit 0 vlan-id 300
set interfaces ge-0/0/0 unit 0 family inet address 20.0.0.4/24
set routing-options static route 0.0.0.0/0 next-hop 20.0.0.250
```

CE-B1

```
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 300
set interfaces ge-0/0/0 unit 0 family inet address 20.0.0.1/24
set routing-options static route 0.0.0.0/0 next-hop 20.0.0.251
```

CE-B2

```
set interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 mac 00:00:00:00:00:02
set interfaces ge-0/0/0 unit 0 vlan-id 300
set interfaces ge-0/0/0 unit 0 family inet address 20.0.0.2/24
set routing-options static route 0.0.0.0/0 next-hop 20.0.0.252
```

CE-B3

```
set interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 mac 00:00:00:00:00:03
set interfaces ge-0/0/0 unit 0 vlan-id 300
set interfaces ge-0/0/0 unit 0 family inet address 20.0.0.3/24
set routing-options static route 0.0.0.0/0 next-hop 20.0.0.253
```

PE1

```
set interfaces ge-0/0/0 unit 0 family inet address 1.0.1.1/30
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 20.0.0.250/24
set interfaces irb unit 1 family inet address 20.0.0.251/24
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 1.0.2.1/30
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 20.0.0.250/24
set interfaces irb unit 1 family inet address 20.0.0.252/24
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 1.0.3.1/30
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 20.0.0.253/24
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 1.0.1.2/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 family inet address 1.0.2.2/30
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/2 unit 0 family inet address 1.0.3.2/30
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
```

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 1.0.1.1/30
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 20.0.0.250/24
user@PE1# set irb unit 1 family inet address 20.0.0.251/24
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 1.0.1.1/30;
      }
      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 {
   gigether-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 20.0.0.250/24;
        address 20.0.0.251/24;
      }
      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

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:

a. EVPN routing instance configuration

b. EVPN multihoming routes

c. DF election process

d. IRB and virtual switch routing instance configuration

e. Host route entries

Verifying the EVPN Instance Status

Purpose Verify the EVPN routing instances and their status.

Router PE1 **Action**

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           00:00:00:00:00:00:00:00:00:00  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           00:00:00:00:00:00:00:00:00:00  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.

Router PE1 **Action**

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 1.0.1.2 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 1.0.1.2 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 1.0.1.2 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 1.0.1.2 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 1.0.1.2 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 1.0.1.2 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 1.0.1.2 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 1.0.1.2 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 1.0.1.2 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 1.0.1.2 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 1.0.1.2 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 1.0.1.2 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 1.0.2.2 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 1.0.2.2 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 1.0.2.2 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 1.0.2.2 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 1.0.2.2 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 1.0.2.2 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 1.0.2.2 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 1.0.2.2 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 1.0.2.2 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 1.0.2.2 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 1.0.2.2 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 1.0.2.2 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 1.0.2.2 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 1.0.2.2 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 1.0.3.2 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 1.0.3.2 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 1.0.3.2 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 1.0.3.2 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 1.0.3.2 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 1.0.3.2 via ge-0/0/0.0, Push 299808
2:10.255.0.2:100::100::00:11:22:33:44:55/304
                   *[BGP/170] 10:47:34, localpref 100, from 10.255.0.2
                      AS path: I, validation-state: unverified
                    > to 1.0.3.2 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 1.0.3.2 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] 10:46:36, localpref 100, from 10.255.0.1
                      AS path: I, validation-state: unverified
                    > to 1.0.3.2 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 1.0.3.2 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 1.0.3.2 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
                   *[BGP/170] 10:41:55, localpref 100, from 10.255.0.2
                      AS path: I, validation-state: unverified
                    > to 1.0.3.2 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] 10:47:34, localpref 100, from 10.255.0.2
                      AS path: I, validation-state: unverified
```



```
                        > to 1.0.3.2 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] 10:47:34, localpref 100, from 10.255.0.2
                          AS path: I, validation-state: unverified
                        > to 1.0.3.2 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
                       *[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 1.0.3.2 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 1.0.3.2 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.

Router PE1 **Action**

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 1.0.1.2 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 1.0.1.2 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 1.0.2.2 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 1.0.2.2 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.

Router PE1 **Action**

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.

Router PE1 **Action**

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 1.0.1.2 via ge-0/0/0.0, Push 300208, Push 299808(top)
10.0.0.3/32        *[EVPN/7] 11:25:54, metric2 1
                    > to 1.0.1.2 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 1.0.1.2 via ge-0/0/0.0, Push 299776, Push 299792(top)
20.0.0.0/24        *[Direct/0] 11:25:55
                    > via irb.1
                    [Direct/0] 11:25:55
                    > via irb.1
20.0.0.1/32        *[EVPN/7] 11:21:20
                    > via irb.1
20.0.0.2/32        *[EVPN/7] 11:19:54, metric2 1
                    > to 1.0.1.2 via ge-0/0/0.0, Push 300240, Push 299808(top)
20.0.0.3/32        *[EVPN/7] 11:25:54, metric2 1
                    > to 1.0.1.2 via ge-0/0/0.0, Push 299808, Push 299792(top)
20.0.0.4/32        *[EVPN/7] 11:24:40
                    > via irb.1
20.0.0.250/32      *[Local/0] 11:38:29
                      Local via irb.1
20.0.0.251/32      *[Local/0] 11:38:29
```



```
                          Local via irb.1
20.0.0.253/32          *[EVPN/7] 11:25:54, metric2 1
                        > to 1.0.1.2 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 1.0.2.2 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 1.0.2.2 via ge-0/0/0.0, Push 299776, Push 299792(top)
10.0.0.4/32            *[EVPN/7] 11:29:04, metric2 1
                        > to 1.0.2.2 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 1.0.2.2 via ge-0/0/0.0, Push 299776, Push 299792(top)
20.0.0.0/24            *[Direct/0] 11:30:02
                        > via irb.1
                        [Direct/0] 11:30:02
                        > via irb.1
20.0.0.1/32            *[EVPN/7] 11:25:37, metric2 1
                        > to 1.0.2.2 via ge-0/0/0.0, Push 300176, Push 299776(top)
20.0.0.2/32            *[EVPN/7] 11:24:11
                        > via irb.1
20.0.0.3/32            *[EVPN/7] 11:30:02, metric2 1
                        > to 1.0.2.2 via ge-0/0/0.0, Push 299808, Push 299792(top)
20.0.0.4/32            *[EVPN/7] 11:28:57, metric2 1
                        > to 1.0.2.2 via ge-0/0/0.0, Push 300176, Push 299776(top)
20.0.0.250/32          *[Local/0] 11:42:33
                          Local via irb.1
20.0.0.252/32          *[Local/0] 11:42:33
                          Local via irb.1
20.0.0.253/32          *[EVPN/7] 11:30:02, metric2 1
                        > to 1.0.2.2 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
                        > to 1.0.3.2 via ge-0/0/0.0, Push 300144, Push 299776(top)
```

```
10.0.0.2/32         *[EVPN/7] 11:24:29, metric2 1
                     > to 1.0.3.2 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 1.0.3.2 via ge-0/0/0.0, Push 300144, Push 299776(top)
10.0.0.250/32       *[EVPN/7] 11:30:08, metric2 1
                     > to 1.0.3.2 via ge-0/0/0.0, Push 300208, Push 299808(top)
10.0.0.252/32       *[EVPN/7] 11:30:08, metric2 1
                     > to 1.0.3.2 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
20.0.0.0/24         *[Direct/0] 11:42:15
                     > via irb.1
20.0.0.1/32         *[EVPN/7] 11:25:43, metric2 1
                     > to 1.0.3.2 via ge-0/0/0.0, Push 300176, Push 299776(top)
20.0.0.2/32         *[EVPN/7] 11:24:17, metric2 1
                     > to 1.0.3.2 via ge-0/0/0.0, Push 300240, Push 299808(top)
20.0.0.3/32         *[EVPN/7] 11:42:04
                     > via irb.1
20.0.0.4/32         *[EVPN/7] 11:29:03, metric2 1
                     > to 1.0.3.2 via ge-0/0/0.0, Push 300176, Push 299776(top)
20.0.0.250/32       *[EVPN/7] 11:30:08, metric2 1
                     > to 1.0.3.2 via ge-0/0/0.0, Push 300240, Push 299808(top)
20.0.0.252/32       *[EVPN/7] 11:30:08, metric2 1
                     > to 1.0.3.2 via ge-0/0/0.0, Push 300240, Push 299808(top)
20.0.0.253/32       *[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.

Router PE1

Action

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
20.0.0.1            00:00:00:00:00:01  irb.1         BETA          __BETA__
20.0.0.4            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
20.0.0.2            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
20.0.0.3            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.

Router PE1 **Action**

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.

Router PE1

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 : ALPHA
 Bridging domain : ONE, VLAN : 100
   MAC                 MAC      Logical          NH      RTR
   address             flags    interface        Index   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                 MAC      Logical          NH      RTR
   address             flags    interface        Index   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.

Example: Configuring EVPN with Support for Virtual Switch

Virtual Switch Support for EVPN Overview

A Data Center Service Provider (DCSP) hosts the data center for its multiple customers onto 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 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) motion.
- Supporting multiple tenants with independent VLAN and subnet space.

Ethernet VPN is targeted to handle all of the above 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

Because each data center hosts multiple tenants with independent VLAN and private subnet space, if an EVPN instance (EVI) was to stretch just one VLAN, it would be a severe problem to scale the data center services. In order to overcome this limitation, starting with Junos OS Release 14.1, a new service called the VLAN-aware bundling service is introduced. This feature 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.

Junos OS has a very flexible and scalable virtual switch interface. With virtual switch a single MX Series router can be divided into multiple logical switches. Layer 2 domains (also called bridge-domains) can be defined independently in each virtual switch. To achieve the VLAN-aware bundling service, an EVPN would be allowed to run in a virtual-switch routing instance.

A single EVPN instance can stretch up to 4094 bridge domains defined in a virtual switch to remote sites. A virtual switch can have more than 4094 bridge domains with a combination of none, single, and dual VLANs. However, because EVPN signaling deals only with single VLAN tags, only a maximum of 4094 bridge domains can be stretched. The EVPN virtual switch also provides support for trunk and access interfaces.

NOTE:

- The none VLAN option is not supported with bridge domains under the virtual switch instance type for EVPNs.
- Dual VLANs are not supported with EVPN although they can be configured.

There are two types of VLAN-aware bundling services:

- VLAN-aware bundling without translation

  The service interface provides bundling of customer VLANs into a single Layer 2 VPN service instance with a guarantee for end-to-end customer VLAN transparency. The data-plane separation between the customer VLANs is maintained by creating a dedicated bridge-domain per VLAN.

- VLAN-aware bundling 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 per 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 bundling with translation only.

Example: Configuring EVPN with Support for Virtual Switch

This example shows how to configure virtual switch in an Ethernet VPN (EVPN) deployment.

- Requirements on page 98
- Overview on page 99
- Configuration on page 99
- Verification on page 106

Requirements

This example uses the following hardware and software components:

- Two MX Series 3D Universal Edge Routers 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

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.

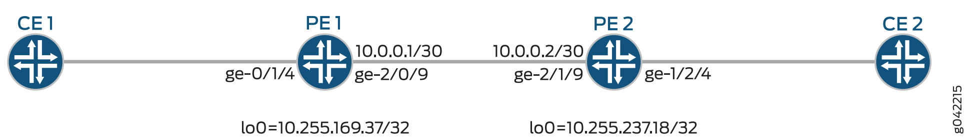
When configuring virtual switch for EVPN, be aware of the following considerations:

- 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 10 on page 99 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 10: EVPN with Virtual Switch Support



Configuration

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 19.1.1.1/16
set interfaces irb unit 1 family inet address 19.2.1.1/16
```



```
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 100.255.169.37:2
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 19.1.2.1/16
set interfaces irb unit 1 family inet address 19.2.2.1/16
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 100.255.237.18:2
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 19.1.1.1/16
user@PE1# set irb unit 1 family inet address 19.2.1.1/16
```

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 100.255.169.37:2
```

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 19.1.1.1/16;
        }
    }
    unit 1 {
        family inet {
            address 19.2.1.1/16;
        }
    }
}
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

Confirm that the configuration is working properly.

- Verifying the Bridge Domain Configuration on page 107
- Verifying MAC Table Routes on page 107
- Verifying the Bridge EVPN Peer Gateway MAC on page 108

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                MAC      Logical          NH     RTR
   address            flags    interface        Index  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.

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 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 EVPN with IRB Solution on page 36

Configuring EVPN 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. In configuration mode, go to the following hierarchy level:

   [edit]
   user@PE1# **edit routing-instances**

3. Configure the virtual switch routing instance.

   [edit routing-instances]
   user@PE1# **set *virtual-switch-instance* instance-type virtual-switch**

4. Configure the extended VLAN list for the virtual switch routing instance.

   [edit routing-instances]
   user@PE1# **set *virtual-switch-instance* protocols evpn extended-vlan-list *VLAN-ID***

5. Set the type for the bridging domain in the virtual switch routing instance.

   [edit routing-instances]
   user@PE1# **set *virtual-switch-instance* bridge-domains *bridge-domain-name* domain-type bridge**

6. Set the VLAN identifier for the bridging domain in the virtual switch routing instance.

   [edit routing-instances]
   user@PE1# **set *virtual-switch-instance* bridge-domains *bridge-domain-name* vlan-id *VLAN-ID***

7. Configure the interfaces for the virtual switch routing instance.

   [edit routing-instances]
   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***

8. Configure the route distinguisher for the virtual switch routing instance.

   [edit routing-instances]
   user@PE1# **set *virtual-switch-instance* route-distinguisher *route-distinguisher-value***

9. Configure the VPN routing and forwarding (VRF) target community for the virtual switch routing instance.

   [edit routing-instances]
   user@PE1# **set *virtual-switch-instance* vrf-target *vrf-target***

10. Configure the EVPN routing instance.

    [edit routing-instances]
    user@PE1# **set *evpn-instance* instance-type evpn**

11. 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***

12. Configure the interface names for the EVPN routing instance.

    [edit routing-instances]
    user@PE1# **set *evpn-instance* interface *interface-name***
    user@PE1# **set *evpn-instance* routing-interface *interface-name***

13. Configure the route distinguisher for the EVPN routing instance.

    [edit routing-instances]
    user@PE1# **set *evpn-instance* route-distinguisher *route-distinguisher-value***

14. 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***

15. Configure the VRF routing instance.

    [edit routing-instances]
    user@PE1# **set *vrf-instance* instance-type vrf**

16. Configure the interface names for the VRF routing instance.

    [edit routing-instances]
    user@PE1# **set *vrf-instance* interface *interface-name***

17. Configure the route distinguisher for the VRF routing instance.

    [edit routing-instances]
    user@PE1# **set *vrf-instance* route-distinguisher *route-distinguisher-value***

18. Configure the VPN routing and forwarding (VRF) target community for the VRF routing instance.

    [edit routing-instances]
    user@PE1# **set *vrf-instance* vrf-target *vrf-target***
    user@PE1# **set *vrf-instance* vrf-table-label**

19. 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 Multihoming on page 55

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 100.255.169.37:2**
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 on page 97

CHAPTER 4

EVPN Configuration Statements

evpn

Syntax

```
evpn {
  designated-forwarder-election-hold-time seconds;
  extended-vlan-list vlan-id | [vlan-id set];
  interface interface-name{
    ignore-encapsulation-mismatch;
    interface-mac-limit limit {
      packet-action drop;
    }
    no-mac-learning;
    static-mac mac-address;
  }
  interface-mac-limit limit {
    packet-action drop;
  }
  label-allocation per-instance;
  mac-statistics;
  mac-table-size limit {
    packet-action drop;
  }
  no-mac-learning;
  traceoptions {
    file filename <files number> <size size> <world-readable | no-world-readable>;
    flag flag <flag-modifier>;
  }
}
```

Hierarchy Level [edit routing-instances *routing-instance-name* protocols]

Release Information Statement introduced in Junos OS Release 13.2 for EVPNs on MX 3D Series routers.
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.

Description Enables an Ethernet VPN (EVPN) on the routing instance.

Options **designated-forwarder-election-hold-time *seconds***—Time in seconds to wait before electing a designated forwarder (DF).

Range: 1 through 1800 seconds

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.

Related Documentation

- Configuring EVPN Routing Instances on page 33
- Tracing EVPN Traffic and Operations on page 35

esi

Syntax

```
esi {
  esi;
  single-active;
}
```

Hierarchy Level [edit interfaces *interface-name*]

Release Information Statement introduced in Junos OS Release 14.1.

Description Configure the Ethernet Segment Identifier (ESI) value for the multihomed interfaces.

Options ***esi***—Ten octet value. ESI value 0 and all fixed filters are reserved, and not used for configuring a multihomed Ethernet segment.

NOTE: **Two interfaces in the same EVPN instance (EVI) cannot be configured with the same ESI value.**

single-active—Configure the EVPN active-standy multihoming mode of operation.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- evpn on page 118

extended-vlan-list

Syntax extended-vlan-list *vlan-id* | [*vlan-id set*];

Hierarchy Level [edit routing-instances *routing-instance-name* instance-type virtual-switch protocols evpn]

Release Information Statement introduced in Junos OS Release 14.1.

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.

Options *vlan-id*—VLAN ID to be EVPN extended.

vlan-id set—List of VLAN IDs to be EVPN extended.

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.

Related Documentation

- evpn on page 118

family (Protocols BGP)

Syntax

```
family {
  (inet | inet6 | inet-vpn | inet6-vpn | iso-vpn) {
    (any | flow | labeled-unicast | multicast | unicast) {
      accepted-prefix-limit {
        maximum number;
        teardown <percentage-threshold> idle-timeout (forever | minutes);
      }
      add-path {
        send {
          path-count number;
          prefix-policy [ policy-names ];
          }
        receive;
      }
      aigp [disable];
      loops number;
      prefix-limit {
        maximum number;
        teardown <percentage> <idle-timeout (forever | minutes)>;
      }
      protection;
    rib-group group-name;
      topology name {
        community {
          target identifier;
        }
      }
    flow {
      no-validate policy-name;
    }
    labeled-unicast {
      accepted-prefix-limit {
        maximum number;
        teardown <percentage> <idle-timeout (forever | minutes)>;
      }
      aggregate-label {
        community community-name:
      }
      explicit-null {
        connected-only;
      }
      prefix-limit {
        maximum number;
        teardown <percentage> <idle-timeout (forever | minutes)>;
      }
      resolve-vpn;
      rib (inet.3 | inet6.3);
      rib-group group-name;
      traffic-statistics {
        file filename <world-readable | no-world-readable>;
        interval seconds;
      }
```

```
      }
    }
    route-target {
      accepted-prefix-limit {
        maximum number;
        proxy-generate <route-target-policy route-target-policy-name>;
        teardown <percentage> <idle-timeout (forever | minutes)>;
      }
      advertise-default;
      external-paths number;
      prefix-limit {
        maximum number;
        teardown <percentage> <idle-timeout (forever | minutes)>;
      }
    }
    (evpn | inet-mdt | inet-mvpn | inet6-mvpn | l2vpn) {
      signaling {
        accepted-prefix-limit {
          maximum number;
          teardown <percentage-threshold> idle-timeout (forever | minutes);
        }
        add-path {
          send {
            path-count number;
            prefix-policy [ policy-names ];
            }
          receive;
        }
        aigp [disable];
        damping;
        loops number;
        prefix-limit {
          maximum number;
          teardown <percentage> <idle-timeout (forever | minutes)>;
        }
        rib-group group-name;
      }
    }
  }
```

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 *address*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols bgp],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols bgp group *group-name*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols bgp group *group-name* neighbor *address*],
[edit protocols bgp],
[edit protocols bgp group *group-name*],
[edit protocols bgp group *group-name* neighbor *address*],
[edit routing-instances *routing-instance-name* protocols bgp],
[edit routing-instances *routing-instance-name* protocols bgp group *group-name*],
[edit routing-instances *routing-instance-name* protocols bgp group *group-name* neighbor *address*]

Release Information
Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.3 for the QFX Series.
inet-mvpn and **inet6-mpvn** statements introduced in Junos OS Release 8.4.
inet-mdt statement introduced in Junos OS Release 9.4.
Support for the **loops** statement introduced in Junos OS Release 9.6.
evpn statement introduced in Junos OS Release 13.2.

Description
Enable multiprotocol BGP (MP-BGP) by configuring BGP to carry network layer reachability information (NLRI) for address families other than unicast IPv4, to specify MP-BGP to carry NLRI for the IPv6 address family, or to carry NLRI for VPNs.

Options
any—Configure the family type to be both unicast and multicast.

evpn—Configure NLRI parameters for Ethernet VPNs (EVPNs).

inet—Configure NLRI parameters for IPv4.

inet6—Configure NLRI parameters for IPv6.

inet-mdt—Configure NLRI parameters for the multicast distribution tree (MDT) subaddress family identifier (SAFI) for IPv4 traffic in Layer 3 VPNs.

inet-mvpn—Configure NLRI parameters for IPv4 for multicast VPNs.

inet6-mvpn—Configure NLRI parameters for IPv6 for multicast VPNs.

inet-vpn—Configure NLRI parameters for IPv4 for Layer 3 VPNs.

inet6-vpn—Configure NLRI parameters for IPv6 for Layer 3 VPNs.

iso-vpn—Configure NLRI parameters for IS-IS for Layer 3 VPNs.

l2vpn—Configure NLRI parameters for IPv4 for MPLS-based Layer 2 VPNs and VPLS.

labeled-unicast—Configure the family type to be labeled-unicast. This means that the BGP peers are being used only to carry the unicast routes that are being used by labeled-unicast for resolving the labeled-unicast routes. This statement is supported only with **inet** and **inet6**.

multicast—Configure the family type to be multicast. This means that the BGP peers are being used only to carry the unicast routes that are being used by multicast for resolving the multicast routes.

unicast—Configure the family type to be unicast. This means that the BGP peers only carry the unicast routes that are being used for unicast forwarding purposes. The default family type is **unicast**.

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.

Related Documentation

- Configuring IBGP Sessions Between PE Routers in VPNs on page 28
- *Understanding Multiprotocol BGP*
- *autonomous-system*
- *local-as*

ignore-encapsulation-mismatch

Syntax ignore-encapsulation-mismatch;

Hierarchy Level [edit logical-systems *logical-system-name* protocols l2circuit local-switching interface *interface-name*],
[edit logical-systems *logical-system-name* protocols l2circuit neighbor *address* interface *interface-name*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls mesh-group *mesh-group-name* neighbor *neighbor-id*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls neighbor *neighbor-id*],
[edit protocols l2circuit local-switching interface *interface-name*],
[edit protocols l2circuit neighbor *address* interface *interface-name*],
[edit routing-instances *routing-instance-name* protocols evpn interface *interface-name*],
[edit routing-instances *routing-instance-name* protocols vpls mesh-group *mesh-group-name* neighbor *neighbor-id*],
[edit routing-instances *routing-instance-name* protocols vpls neighbor *neighbor-id*]

Release Information Statement introduced in Junos OS Release 9.2.
Statement extended to support local switching in Junos OS Release 10.4.
Statement introduced for EVPNs in Junos OS Release 13.2 for MX 3D Series.

Description Allow a Layer 2 circuit, VPLS, or EVPN to be established even though the encapsulation configured on the CE device interface does not match the encapsulation configured on the Layer 2 circuit, VPLS, or EVPN interface.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration

Related Documentation

- Configuring EVPN Routing Instances on page 33
- *Configuring Interfaces for Layer 2 Circuits*

instance-type

Syntax

instance-type *type*;

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name*],
[edit routing-instances *routing-instance-name*]

Release Information

Statement introduced before Junos OS Release 7.4.
virtual-switch and **layer2-control** options introduced in Junos OS Release 8.4.
Statement introduced in Junos OS Release 9.2 for EX Series switches.
Statement introduced in Junos OS Release 11.3 for the QFX Series.
Statement introduced in Junos OS Release 12.3 for ACX Series routers.
evpn option introduced in Junos OS Release 13.2 for MX 3D Series routers.

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 only) Enable an Ethernet VPN (EVPN) on the routing instance. You cannot configure the **evpn** option under the **[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* instance-type]** hierarchy level.
- **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.

- **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 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.

Related Documentation

- *Example: Using Virtual Routing Instances to Route Among VLANs on EX Series Switches*
- *Configuring Routing Instances on PE Routers in VPNs*
- Configuring EVPN Routing Instances on page 33
- *Configuring Virtual Routing Instances (CLI Procedure)*
- *Configuring Virtual Router Routing Instances*
- *Example: Configuring Filter-Based Forwarding on the Source Address*
- *Example: Configuring Filter-Based Forwarding on Logical Systems*
- *Layer 2 Routing Instance Types*

interface (EVPN Routing Instances)

Syntax

```
interface interface-name {
  ignore-encapsulation-mismatch;
  interface-mac-limit limit {
    packet-action drop;
  }
  no-mac-learning;
  static-mac mac-address;
}
```

Hierarchy Level [edit routing-instances *routing-instance-name* protocols evpn]

Release Information Statement introduced in Junos OS Release 13.2 for EVPNs on MX 3D Series routers.

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.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring EVPN Routing Instances on page 33
- evpn on page 118
- instance-type on page 125

interface (Routing Instances)

Syntax

interface *interface-name* {
  description *text*;
}

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name*],
[edit routing-instances *routing-instance-name*]

Release Information

Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.2 for EX Series switches.
Statement introduced in Junos OS Release 12.3 for ACX Series routers.
Statement introduced in Junos OS Release 13.2 for MX 3D Series routers.

Description

Interface over which the VPN traffic travels between the PE device and CE device. You configure the interface on the PE device. If the value **vrf** is specified for the **instance-type** statement included in the routing instance configuration, this statement is required.

Options

interface-name—Name of the interface.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- *Configuring Routing Instances on PE Routers in VPNs*
- Configuring EVPN Routing Instances on page 33
- *Example: Configuring MPLS-Based Layer 3 VPNs on EX Series Switches*
- *interface (VPLS Routing Instances)*

interface-mac-limit

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*]

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.

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.

Options

limit—Specify the number of MAC addresses that can be learned from each interface.

Range: 16 through 65,536 MAC addresses

Default: 512 addresses

The remaining statement is explained separately.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring EVPN Routing Instances on page 33
- *Configuring VPLS Routing Instances*
- interface on page 127
- mac-table-size on page 132

label-allocation

Syntax label-allocation per-instance;

Hierarchy Level [edit routing-instances *routing-instance-name* protocols evpn]

Release Information Statement introduced in Junos OS Release 13.2 on MX 3D Series routers.

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.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring EVPN Routing Instances on page 33

mac-statistics

Syntax mac-statistics;

Hierarchy Level [edit bridge-domains *bridge-domain-name* bridge-options],
[edit logical-systems *logical-system-name* bridge-domains *bridge-domain-name* bridge-options],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* bridge-domains *bridge-domain-name* bridge-options],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* switch-options],
[edit logical-systems *logical-system-name* switch-options],
[edit routing-instances *routing-instance-name* bridge-domains *bridge-domain-name* bridge-options],
[edit routing-instances *routing-instance-name* switch-options],
[edit routing-instances *routing-instance-name* protocols evpn],
[edit switch-options],
[edit switch-options],
[edit vlans *vlan-name* switch-options]

Release Information Statement introduced in Junos OS Release 8.4.
Support for the **switch-options** statement added in Junos OS Release 9.2.
Support for top-level configuration for the **virtual-switch** type of routing instance added in Junos OS Release 9.2. In Junos OS Release 9.1 and earlier, the routing instances hierarchy supported this statement only for a VPLS instance or a bridge domain configured within a virtual switch.
Support for logical systems added in Junos OS Release 9.6.
[edit switch-options] and [edit vlans *vlan-name* switch-options] hierarchy levels introduced in Junos OS Release 12.3R2 for EX Series switches.
Support for EVPNs added in Junos OS Release 13.2 for MX 3D Series routers.
[edit switch-options] and [edit vlans *vlan-name* switch-options] hierarchy levels introduced in Junos OS Release 13.2 for the QFX Series.

Description (MX Series routers, EX Series switches, and QFX Series only) For bridge domains or VLANs, enable MAC accounting either for a specific bridge domain or VLAN, or for a set of bridge domains or VLANs associated with a Layer 2 trunk port.

Default disabled

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- *Layer 2 Learning and Forwarding for Bridge Domains Overview*
- *Layer 2 Learning and Forwarding for VLANs Overview*
- *Layer 2 Learning and Forwarding for Bridge Domains Functioning as Switches with Layer 2 Trunk Ports*
- *Layer 2 Learning and Forwarding for VLANs Acting as a Switch for a Layer 2 Trunk Port*
- Configuring EVPN Routing Instances on page 33

mac-table-size

Syntax

```
mac-table-size size {
  packet-action drop;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls],
[edit routing-instances *routing-instance-name* protocols evpn],
[edit routing-instances *routing-instance-name* protocols vpls]

Release Information

Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 13.2 for EVPNs on MX 3D Series routers.

Description

Specify the size of the MAC address table.

Options

size—Specify the size of the MAC address table.

Range:

- (M Series and T Series routers only) 16 through 65,536 MAC addresses
- (MX Series routers only) 16 through 1,048,575 MAC addresses
- (T4000 routers with Type 5 FPCs only) 16 through 262,143 MAC addresses

NOTE: Before modifying the size of the MAC address table (to 262,143 addresses), you must enable network services mode by including the **enhanced-mode** statement at the **[edit chassis network-services]** hierarchy level and then reboot the router.

Default: 512 MAC addresses

The remaining statement is explained separately.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring EVPN Routing Instances on page 33
- *Configuring VPLS Routing Instances*
- *Configuring Improved VPLS MAC Address Learning on T4000 Routers with Type 5 FPCs*
- *enhanced-mode*
- evpn on page 118

no-mac-learning

Syntax

no-mac-learning;

Hierarchy Level

[edit bridge-domains *bridge-domain-name* bridge-options],
[edit bridge-domains *bridge-domain-name* bridge-options interface *interface-name*],
[edit logical-systems *logical-system-name* bridge-domains *bridge-domain-name* bridge-options],
[edit logical-systems *logical-system-name* bridge-domains *bridge-domain-name* bridge-options interface *interface-name*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* bridge-domains *bridge-domain-name* bridge-options],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* bridge-domains *bridge-domain-name* bridge-options interface *interface-name*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* switch-options],
[edit logical-systems *logical-system-name* switch-options],
[edit bridge-domains *bridge-domain-name* bridge-options interface *interface-name*],
[edit routing-instances *routing-instance-name* bridge-domains *bridge-domain-name* bridge-options],
[edit routing-instances *routing-instance-name* bridge-domains *bridge-domain-name* bridge-options interface *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* switch-options],
[edit switch-options],
[edit switch-options],
[edit switch-options interface *interface-name*],
[set vlans *vlan-name* switch-options]

Release Information

Statement introduced in Junos OS Release 8.4.
Support for the **switch-options** statement added in Junos OS Release 9.2.
Support for top-level configuration for the **virtual-switch** type of routing instance added in Junos OS Release 9.2. In Junos OS Release 9.1 and earlier, the routing instances hierarchy supported this statement only for a VPLS instance or bridge domain configured within a virtual switch.
Support for logical systems added in Junos OS Release 9.6.
[edit switch-options], **[edit switch-options interface *interface-name*]**, **[edit vlans *vlan-name* switch-options]**, and **[edit vlans *vlan-name* switch-options interface *interface-name*]** hierarchy levels introduced in Junos OS Release 12.3 R2 for EX Series switches.
Support for EVPNs added in Junos OS Release 13.2 for MX 3D Series routers.
Hierarchy levels **[edit switch-options interface *interface-name*]** and **[edit vlans *vlan-name* switch-options]** introduced in Junos OS Release 13.2X50-D10 for EX Series switches.

Description

For MX Series routers and EX Series switches, disable MAC learning for a virtual switch, for a bridge domain or VLAN, for a specific logical interface in a bridge domain or VLAN, or for a set of bridge domains or VLANs associated with a Layer 2 trunk port. On platforms that support EVPNs, you can disable MAC learning on an EVPN.

NOTE: When MAC learning is disabled for a VPLS routing instance, traffic is not load-balanced and only one of the equal-cost next hops is used.

Default MAC learning is enabled.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring EVPN Routing Instances on page 33
- *Layer 2 Learning and Forwarding for Bridge Domains Overview*
- *Layer 2 Learning and Forwarding for VLANs Overview*
- *Layer 2 Learning and Forwarding for Bridge Domains Functioning as Switches with Layer 2 Trunk Ports*
- *Understanding Bridging and VLANs on EX Series Switches*
- *Understanding Q-in-Q Tunneling on EX Series Switches*

packet-action

Syntax packet-action *action*;

Hierarchy Level
[edit bridge-domains *bridge-domain-name* bridge-options interface *interface-name* interface-mac-limit *limit*],
[edit bridge-domains *bridge-domain-name* bridge-options interface-mac-limit *limit*],
[edit logical-systems *logical-system-name* bridge-domains *bridge-domain-name* bridge-options interface *interface-name* interface-mac-limit *limit*],
[edit logical-systems *logical-system-name* bridge-domains *bridge-domain-name* bridge-options interface-mac-limit *limit*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* bridge-domains *bridge-domain-name* bridge-options interface *interface-name* interface-mac-limit *limit*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* bridge-domains *bridge-domain-name* bridge-options interface-mac-limit *limit*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* switch-options interface *interface-name* interface-mac-limit *limit*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* switch-options interface-mac-limit *limit*],
[edit logical-systems *logical-system-name* switch-options interface-mac-limit *limit*],
[edit protocols l2-learning global-mac-limit *limit*],
[edit routing-instances *routing-instance-name* bridge-domains *bridge-domain-name* bridge-options interface *interface-name* interface-mac-limit *limit*],
[edit routing-instances *routing-instance-name* bridge-domains *bridge-domain-name* bridge-options interface-mac-limit *limit*],
[edit routing-instances *routing-instance-name* protocols evpn interface-mac-limit],
[edit routing-instances *routing-instance-name* protocols evpn interface *interface-name* interface-mac-limit],
[edit routing-instances *routing-instance-name* protocols evpn mac-table-size *limit*],
[edit routing-instances *routing-instance-name* switch-options interface *interface-name* interface-mac-limit *limit*],
[edit routing-instances *routing-instance-name* switch-options interface-mac-limit *limit*],
[edit switch-options interface-mac-limit *limit*],
[edit switch-options interface *interface-name* interface-mac-limit *limit*],
[edit switch-options interface-mac-limit *limit*],
[edit switch-options interface *interface-name* interface-mac-limit *limit*],
[edit switch-options interface-mac-limit *limit*],
[edit switch-options mac-table-size *limit*],
[edit switch-options interface *interface-name* interface-mac-limit *limit*],
[edit vlans *vlan-name* switch-options interface *interface-name* interface-mac-limit *limit*],
[edit vlans *vlan-name* switch-options interface-mac-limit *limit*],
[edit vlans *vlan-name* switch-options mac-table-size *limit*]
[edit vlans *vlan-name* switch-options interface-mac-limit *limit*],
[edit vlans *vlan-name* switch-options interface *interface-name* interface-mac-limit *limit*],
[edit vlans *vlan-name* switch-options mac-table-size *limit*]

Release Information
Statement introduced in Junos OS Release 8.4.
Support for the **switch-options** statement added in Junos OS Release 9.2.
Support for top-level configuration for the **virtual-switch** type of routing instance added in Junos OS Release 9.2. In Junos OS Release 9.1 and earlier, the routing instances hierarchy supported this statement only for a VPLS instance or a bridge domain configured within a virtual switch.

Support for logical systems added in Junos OS Release 9.6.
[edit switch-options interface *interface-name* interface-mac-limit *limit*], [edit switch-options interface-mac-limit *limit*], [edit switch-options mac-table-size *limit*], [edit vlans *vlan-name* switch-options interface *interface-name* interface-mac-limit *limit*], [edit vlans *vlan-name* switch-options interface-mac-limit *limit*], and [edit vlans *vlan-name* switch-options mac-table-size *limit*] hierarchy levels introduced in Junos OS Release 12.3R2 for EX Series switches.
Support for EVPNs introduced in Junos OS Release 13.2 on MX Series 3D Universal Edge Routers.
Support at the **[edit switch-options interface *interface-name* interface-mac-limit *limit*]** hierarchy level and hierarchy levels under **[edit vlans *vlan-name*]** introduced in Junos OS Release 13.2X50-D10 for EX Series switches and Junos OS Release 13.2 for the QFX Series.

Description Specify the action taken when packets with new source MAC addresses are received after the MAC address limit is reached. If this statement is not configured, packets with new source MAC addresses are forwarded by default.

NOTE: On a QFX Series Virtual Chassis, if you include the **shutdown** option at the **[edit vlans *vlan-name* switch-options interface *interface-name* interface-mac-limit packet-action]** hierarchy level and issue the **commit** operation, the system generates a commit error. The system does not generate an error if you include the **shutdown** option at the **[edit switch-options interface *interface-name* interface-mac-limit packet-action]** hierarchy level.

Default Disabled. The default is for packets for new source MAC addresses to be forwarded after the MAC address limit is reached.

Options

drop—Drop packets with new source MAC addresses, and do not learn the new source MAC addresses.

drop-and-log—(EX Series switches and QFX Series only) Drop packets with new source MAC addresses, and generate an alarm, an SNMP trap, or a system log entry.

log—(EX Series switches and QFX Series only) Hold packets with new source MAC addresses, and generate an alarm, an SNMP trap, or a system log entry.

none—(EX Series switches and QFX Series only) Forward packets with new source MAC addresses, and learn the new source MAC address.

shutdown—(EX Series switches and QFX Series only) Disable the specified interface, and generate an alarm, an SNMP trap, or a system log entry.

Required Privilege Level
routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring EVPN Routing Instances on page 33
- *Configuring MAC Limiting (CLI Procedure)*
- *Configuring Persistent MAC Learning (CLI Procedure)*
- *Layer 2 Learning and Forwarding for Bridge Domains Overview*
- *Layer 2 Learning and Forwarding for VLANs Overview*
- *Layer 2 Learning and Forwarding for Bridge Domains Functioning as Switches with Layer 2 Trunk Ports*
- *Layer 2 Learning and Forwarding for VLANs Overview*
- *Layer 2 Learning and Forwarding for VLANs Acting as a Switch for a Layer 2 Trunk Port*

routing-instances

Syntax routing-instances *routing-instance-name* { ... }

Hierarchy Level [edit],
[edit logical-systems *logical-system-name*]

Release Information Statement introduced before Junos OS Release 7.4.

Description Configure an additional routing entity for a router. You can create multiple instances of BGP, IS-IS, OSPF, OSPF version 3 (OSPFv3), and RIP for a router.

Default Routing instances are disabled for the router.

Options ***routing-instance-name***—Name of the routing instance, a maximum of 31 characters. 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.

Related Documentation

- Configuring EVPN Routing Instances on page 33
- *Configuring Routing Instances on PE Routers in VPNs*

static-mac

Syntax

static-mac *mac-address* {
  vlan-id *number*;
}

Hierarchy Level

[edit bridge-domains *bridge-domain-name* bridge-options interface *interface-name*],
[edit logical-systems *logical-system-name* bridge-domains *bridge-domain-name* bridge-options interface *interface-name*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* bridge-domains *bridge-domain-name* bridge-options interface *interface-name*],
[edit routing-instances *routing-instance-name* bridge-domains *bridge-domain-name* bridge-options interface *interface-name*],
[edit routing-instances *routing-instance-name* protocols evpn interface *interface-name*]
[edit vlans *vlan-name* switch-options interface *interface-name*]

Release Information

Statement introduced in Junos OS Release 8.4.
Support for logical systems added in Junos OS Release 9.6.
[edit vlans *vlan-name* switch-options interface *interface name*] hierarchy level introduced in Junos OS Release 12.3R2 for EX Series switches.
Support for EVPNs added in Junos OS Release 13.2 for MX 3D Series routers. The **vlan-id** option is not available for EVPNs.
[edit vlans *vlan-name* switch-options interface *interface name*] hierarchy level introduced in Junos OS Release 13.2 for the QFX Series.

Description

Configure a static MAC address for a logical interface in a bridge domain or VLAN.

The **vlan-id** option can be specified for **static-macs** only if **vlan-id all** is configured for the bridging domain or VLAN.

Options

mac-address—MAC address

vlan-id *number*—(Optional) VLAN identifier to associate with static MAC address.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring EVPN Routing Instances on page 33
- *Layer 2 Learning and Forwarding for Bridge Domains Overview*
- *Layer 2 Learning and Forwarding for VLANs Overview*

traceoptions (Protocols EVPN)

Syntax

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

Hierarchy Level [edit routing-instances *routing-instance-name* protocols evpn]

Release Information Statement introduced in Junos OS Release 13.2 for MX 3D Series routers.

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: ***x*k** to specify kilobytes, ***x*m** to specify megabytes, or ***x*g** 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.

Related Documentation

- Tracing EVPN Traffic and Operations on page 35

vlan-id (routing instance)

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*]

Release Information Statement introduced in Junos OS Release 13.2.

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.

Related Documentation

- Configuring EVPN Routing Instances on page 33

PART 3

Administration

CHAPTER 5

EVPN Operational Commands

show evpn instance

Syntax

show evpn instance
<brief | extensive>
<backup-forwarder>
<designated-forwarder>
<esi *esi*>
<*instance-name*>
<neighbor *neighbor-address*>

Release Information Command introduced in Junos OS Release 14.1.

Description Show Ethernet VPN (EVPN) routing instance information.

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 forwarder (non-DFs) for the Ethernet segment.

designated-forwarder—(Optional) Display IP address of the designated forwarder for the Ethernet segment.

esi ***esi***—(Optional) Display brief information about the routing instance associated with this ESI value.

instance-name—(Optional) Display information for the specified routing instance.

neighbor ***neighbor-address***—(Optional) Display IP address of the EVPN neighbor.

Required Privilege Level view

List of Sample Output

show evpn instance brief on page 148
show evpn instance extensive on page 148
show evpn instance instance-name esi esi-value backup-forwarder on page 150
show evpn instance instance-name esi esi-value designated-forwarder on page 150

Output Fields Table 3 on page 146 lists the output fields for the **show evpn instance** command. Output fields are listed in the approximate order in which they appear.

Table 3: 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**

Table 3: show evpn instance Output Fields *(continued)*

Field Name	Field Description	Level of Output
IRB intfs	Statistics on number of IRB interfaces for each routing instance: • **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**
Route Distinguisher	Unique route distinguisher associated with this routing instance.	**extensive**
Per-instance MAC route label	Label of MAC route for each routing instance.	**extensive**
Per-instance multicast route label	Label of multicast route for each routing instance.	**extensive**
Total MAC addresses	Total number of MAC addresses received for each routing instance.	**extensive**
Default gateway MAC addresses	Number of MAC addresses serving as a default gateway in the routing instance.	**extensive**
Number of local interfaces	Number of 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**
Mode	Mode of operation for each routing instance: • **single-homed**—Default mode and does not require Ethernet segment values to be configured. • **single-active**—EVPN active-standy 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**

Table 3: show evpn instance Output Fields *(continued)*

Field Name	Field Description	Level of Output
L3 context	Names of routing instances that have the Layer 3 routes installed for an EVPN IRB interface, typically a VRF routing instance.	**extensive**
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	IP address of all the backup designated forwarders (BDFs) or non-DF routers for the Ethernet segment. NOTE: Immediately after an evpn interface esi value is changed and the new configuration is committed, the **Designated forwarder** information will change to **DF not elected yet** and the Backup forwarder information will not be displayed until after the election is complete.	**extensive**

Sample Output

show evpn instance brief

```
user@host> show evpn instance brief
                          Intfs       IRB intfs        MH      MAC addresses
Instance                       Total    Up  Total    Up  Nbrs  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 extensive

```
user@host> 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           00:00:00:00:00:00:00:00:00:00  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
```

show evpn instance instance-name esi esi-value backup-forwarder

```
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 instance-name esi esi-value designated-forwarder

```
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
```

PART 4

Index

Index

Symbols

A

B

C

D

E

F

I