



---

# Integrating a Virtual Chassis Fabric into an EVPN-VXLAN Environment



Modified: 2018-12-18

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

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

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

*Integrating a Virtual Chassis Fabric into an EVPN-VXLAN Environment*  
Copyright © 2018 Juniper Networks, Inc. All rights reserved.

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

#### YEAR 2000 NOTICE

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

#### END USER LICENSE AGREEMENT

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

# Table of Contents

<b>Chapter 1</b>	<b>Integrating a Virtual Chassis Fabric into an EVPN-VXLAN Environment . . . . .</b>	<b>5</b>
	About This Network Configuration Example . . . . .	5
	Use Case Overview . . . . .	5
	Need For a Layer 2 Overlay . . . . .	5
	Virtual Chassis Fabric Integration with IP Fabric Architectures . . . . .	6
	Multihoming . . . . .	6
	Technical Overview . . . . .	6
	Virtual Chassis Fabric Overview . . . . .	6
	Understanding VXLAN . . . . .	8
	VXLAN Control Plane Limitations . . . . .	8
	Understanding EVPN . . . . .	9
	Example: Configuring EVPN-VXLAN on a Virtual Chassis Fabric . . . . .	10
	Appendix: Quick Configurations for Configuration Example . . . . .	31



## CHAPTER 1

# Integrating a Virtual Chassis Fabric into an EVPN-VXLAN Environment

- [About This Network Configuration Example on page 5](#)
- [Use Case Overview on page 5](#)
- [Technical Overview on page 6](#)
- [Example: Configuring EVPN-VXLAN on a Virtual Chassis Fabric on page 10](#)
- [Appendix: Quick Configurations for Configuration Example on page 31](#)

## About This Network Configuration Example

---

This network configuration example (NCE) describes how to configure EVPN-VXLAN on QFX5100 devices configured as a Virtual Chassis Fabric (VCF) to enable integration with IP fabric architectures. This configuration is also applicable to QFX5100 standalone devices, as well as QFX5100 devices configured as a Virtual Chassis.

The use case covered in this document illustrates how to stretch a Layer 2 network using EVPN-VXLAN on the QFX5100 and the QFX5100 VCF, and how to multihomed a server to two separate VCFs acting as EVPN PEs to achieve fabric level redundancy.

The NCE begins by providing a brief overview of VCF and EVPN-VXLAN technologies, and then provides detailed step-by-step configuration steps for enabling EVPN-VXLAN on the QFX5100 VCF.

## Use Case Overview

---

### Need For a Layer 2 Overlay

In traditional data centers, Layer 2 technologies are often used to support various design requirements, including:

- Heartbeat exchange and internode communication between high availability clusters
- Server farm growth and physical server placement flexibility
- Virtual machine (VM) migration
- Disaster recovery and geographically dispersed clustering

The Juniper Networks Virtual Chassis Fabric (VCF) provides a low-latency, high-performance Layer 2 fabric architecture for data centers that can be managed as a single device, and meets the requirements of many data center environments.

However, as the design of these data centers evolves to scale out multitenant networks, a new data center architecture is needed that decouples the underlay (physical) network from a tenant overlay network. Using a Layer 3, IP-based underlay coupled with a Virtual Extensible LAN (VXLAN)-Ethernet VPN (EVPN) overlay, data center and cloud operators can deploy much larger networks than are otherwise possible with traditional Layer 2, Ethernet-based architectures. With overlays, endpoints (servers or VMs) can be placed anywhere in the network and remain connected to the same logical Layer 2 network, enabling the virtual topology to be decoupled from the physical topology.

## Virtual Chassis Fabric Integration with IP Fabric Architectures

Starting with Junos OS Release 14.1X53-D40, a VCF can integrate with EVPN-VXLAN environments and act as an EVPN PE device. This enables the entire VCF—with up to 16 top-of-rack (TOR) switches—to appear as a single PE device, with a single VXLAN tunnel endpoint (VTEP). This capability maintains the existing Ethernet fabric for intra-VCF traffic, with all the advantages and features it provides, while also merging it with the larger EVPN-VXLAN IP fabric to provide connectivity with the rest of the data center.

## Multihoming

Server multihoming to redundant TOR devices is another common requirement in data centers. Traditionally, this requirement required proprietary solutions such as multichassis link aggregation (MC-LAG) or Virtual Chassis/VCF. While each solution has its merits, it does require the same vendor across these devices, and in the case of MC-LAG, multihoming is limited to two PE devices.

Ethernet VPN (EVPN), on the other hand, is a standards-based multihoming solution that can scale horizontally across any number of PE devices, and seamlessly integrates into multivendor Layer 3 IP fabrics. With support for active-active endpoint multihoming, EVPN can enable redundant server connections to multiple VCFs to provide fabric level redundancy.

### Related Documentation

- [Example: Configuring EVPN-VXLAN on a Virtual Chassis Fabric on page 10](#)
- [Virtual Chassis Fabric Feature Guide](#)
- [EVPN Control Plane and VXLAN Data Plane Feature Guide for QFX5100 Switches](#)
- [Juniper Networks EVPN Implementation for Next-Generation Data Center Architectures](#)

## Technical Overview

---

### Virtual Chassis Fabric Overview

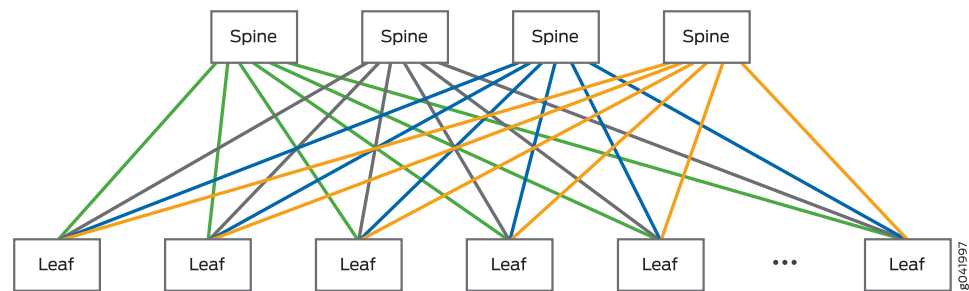
The Juniper Networks Virtual Chassis Fabric (VCF) provides a low-latency, high-performance fabric architecture that can be managed as a single device. VCF is an evolution of the Virtual Chassis feature, which enables you to interconnect multiple devices into a single logical device, into of a fabric architecture. The VCF architecture is

optimized to support small- and medium-sized data centers that contain a mix of 1-Gbps, 10-Gbps, and 40-Gbps Ethernet interfaces.

A VCF is constructed using a spine-and-leaf architecture, where each spine device is connected to each leaf device. A VCF supports up to twenty total devices, with up to four configured as spine devices.

Figure 1 on page 7 illustrates a typical VCF spine-and-leaf architecture.

Figure 1: VCF Spine-and-Leaf Architecture



Note that each spine device must be a QFX5100 device, and in an optimal VCF configuration the leaf devices are also QFX5100s. However, you can also create a mixed VCF using QFX3600, QFX3500, and EX4300 switches as leaf devices.



**NOTE:** For EVPN-VXLAN integration, the VCF must consist of QFX5100 devices only.

A VCF provides the following benefits:

- **Latency**—VCF provides predictable low latency because it uses a fabric architecture that ensures each device is one or two hops away from every other device in the fabric. The weighted algorithm that makes traffic-forwarding decisions in a VCF is designed to avoid congestion and ensures low latency by intelligently forwarding traffic over all paths within the VCF.
- **Resiliency**—The VCF architecture provides a resilient framework because traffic has multiple paths across the fabric. Traffic is, therefore, easily diverted within the fabric when a device or link fails.
- **Flexibility**—You can easily expand the size of your VCF by adding devices to the fabric as your networking needs grow.
- **Investment protection**—In environments that need to expand beyond the capabilities of a traditional QFX5100, QFX3600, QFX3500, or EX4300 Virtual Chassis, a VCF is often a logical upgrade option because it enables the system to evolve by leveraging existing devices.
- **Manageability**—VCF provides multiple features that simplify configuration and management, such as autoprovisioning to add new devices into the fabric after minimal

initial configuration. VCF also leverages many of the existing configuration procedures from a Virtual Chassis.



**NOTE:** For more information on Virtual Chassis Fabric, see the [Virtual Chassis Fabric Feature Guide](#).

## Understanding VXLAN

Network overlays are created by encapsulating traffic and tunneling it over a physical network. A number of tunneling protocols can be used in the data center to create network overlays—the most common protocol is Virtual Extensible LAN (VXLAN). The VXLAN tunneling protocol encapsulates Layer 2 Ethernet frames in Layer 3 UDP packets to enable virtual Layer 2 subnets or segments that can span the underlying (physical) Layer 3 network.

In a VXLAN overlay network, each Layer 2 subnet or segment is uniquely identified by a virtual network identifier (VNI). A VNI enables segmenting of traffic the same way that a VLAN ID segments traffic. As is the case with VLANs, endpoints with the same VNI can communicate directly with each other, whereas endpoints on different VNIs require a router, or gateway.

The entity that performs VXLAN encapsulation and decapsulation is called a VXLAN tunnel endpoint (VTEP). VTEPs typically reside in hypervisor hosts, such as ESXi or KVM hosts, but can also reside in network devices to support bare-metal server (BMS) endpoints. Each VTEP is typically assigned a unique IP address.

## VXLAN Control Plane Limitations

The VXLAN abstraction does not change the flood and learn behavior of the Ethernet protocol, which has inherent limitations in terms of scalability, efficiency, and utilization.

VXLAN can be deployed as a tunneling protocol across a Layer 3 IP fabric data center without a control plane protocol. Two primary methods exist for doing this: VXLAN with a multicast-enabled underlay, and static unicast VXLAN tunnels. While both are viable options for eliminating Layer 2 in the underlay, neither solves the inherent flood-and-learn problem, and both are difficult to scale to large multitenant environments.

The solution instead is to introduce a control plane to minimize flooding and facilitate learning. To facilitate learning, the control plane distributes end host information to Virtual Tunnel End Points (VTEPs) in the same segment.

An extension to Multiprotocol BGP (MP-BGP) addresses the flood and learn problem. MP-BGP allows the network to carry both Layer 2 MAC and Layer 3 IP information at the same time, and having this combined set of information available for forwarding decisions allows optimized routing and switching. This extension that allows BGP to transport both MAC and IP information is called Ethernet VPN (EVPN).





**NOTE:** For more information on VXLAN, see the [EVPN Control Plane and VXLAN Data Plane Feature Guide for QFX5100 Switches](#).

## Understanding EVPN

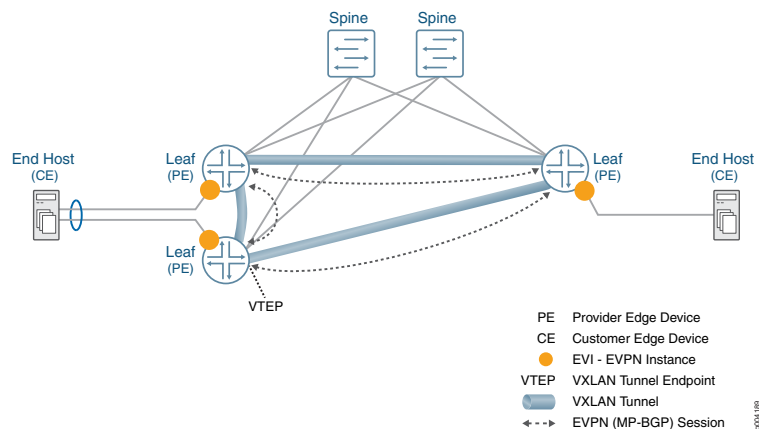
Ethernet VPN (EVPN) is a standards-based protocol that provides virtual multipoint bridged connectivity between different domains over an IP or IP/MPLS backbone network. This control-plane technology uses Multiprotocol BGP (MP-BGP) for MAC and IP address (endpoint) distribution, with MAC addresses being treated as “routes.” As used in data center environments, EVPN enables devices acting as VTEPs to exchange reachability information with each other about their endpoints.

Like other VPN technologies, such as IP VPN and virtual private LAN service (VPLS), EVPN instances (EVIs) are configured on provider edge (PE) routers to maintain logical service separation between customers. PE routers connect to customer edge (CE) devices, which can be routers, switches, or hosts. The PE routers then exchange reachability information using Multiprotocol BGP (MP-BGP), after which encapsulated traffic can be forwarded between them. Because elements of the architecture are common with other VPN technologies, EVPN can be seamlessly introduced and integrated into existing service environments.

In data center environments, PE routers are referred to as leaf devices. These devices connect to upstream devices called spine devices to form a Layer 3 spine-and-leaf architecture, or IP fabric. In these environments, customer edge (CE) devices are typically endpoints such as servers or virtual machines (VMs).

A typical EVPN deployment is shown in [Figure 2 on page 9](#).

**Figure 2: Typical EVPN Environment**



EVPN enables multitenancy, flexible services that can be extended on demand, frequently using compute resources of different physical data centers. In addition, EVPN's technical capabilities include:

- Multihoming—provides multipath forwarding and redundancy through an “all-active” model, allowing an endpoint to connect to two or more leaf devices and forward traffic using all of the links. In the event that an access link or one of the leaf (PE) devices fails, traffic flows from the endpoint (CE device) toward the leaf device using the remaining active link(s). For traffic in the other direction, remote leaf devices update their forwarding tables to send traffic to the remaining active leaf device(s) connected to the multihomed Ethernet segment.
- Aliasing—leverages all-active multihoming to allow a remote leaf device to load-balance traffic across the network toward the endpoint.
- Split horizon—prevents the looping of broadcast, unknown unicast, and multicast (BUM) traffic in a network. With split horizon, a packet is never sent back in the direction from which it was received.
- VM mobility—also known as Virtual Machine Traffic Optimization (VMTO), enables live VMs to be dynamically moved from one data center to another.

Benefits of using EVPNs include:

- Ability to have an active multihomed edge device
- Load balancing across multiple links
- MAC address mobility
- Multitenancy
- Aliasing
- Fast convergence



**NOTE:** For more information on EVPN, see [Juniper Networks EVPN Implementation for Next-Generation Data Center Architectures](#).

---

**Related  
Documentation**

- [Use Case Overview on page 5](#)
- [EVPN-VXLAN Support of Virtual Chassis and Virtual Chassis Fabric](#)
- [Virtual Chassis Fabric Feature Guide](#)
- [EVPN Control Plane and VXLAN Data Plane Feature Guide for QFX5100 Switches](#)
- [Juniper Networks EVPN Implementation for Next-Generation Data Center Architectures](#)

---

## Example: Configuring EVPN-VXLAN on a Virtual Chassis Fabric

---

This section illustrates how to configure EVPN-VXLAN on QFX5100 devices configured as a Virtual Chassis Fabric (VCF). This configuration is also applicable to QFX5100 standalone devices and QFX5100 devices configured as a Virtual Chassis.

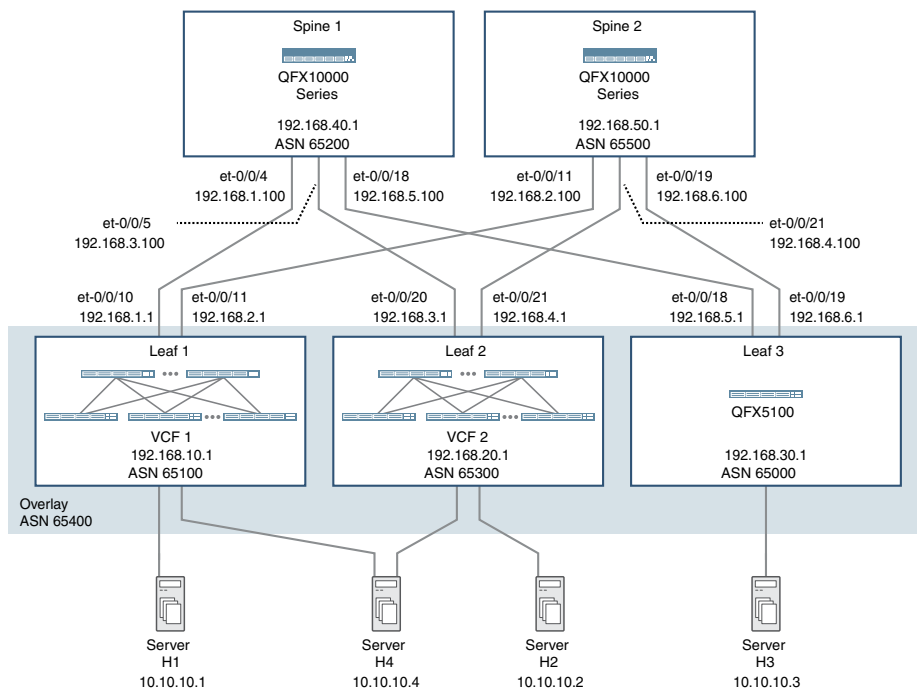
## Requirements

- Two QFX10002-36Q devices running Junos OS Release 15.1X53-D30 or later
- Two QFX5100-based VCFs running Junos OS Release 14.1X53-D40 or later
- One QFX5100 standalone device running Junos OS Release 14.1X53-D30 or later

## Overview & Topology

The topology used in this configuration example is shown in [Figure 3 on page 11](#).

*Figure 3: IP Fabric Topology With VCFs as Leaf Devices*



The topology uses an IP fabric architecture, with QFX10002 switches as spine devices. For this example, they do not participate in the EVPN control plane and are only providing underlay connectivity. The leaf layer includes two Virtual Chassis Fabrics (already configured) using QFX5100 switches, and a standalone QFX5100 switch, each acting as a leaf device, or EVPN PE. Attached to the leaf layer are four servers, H1 through H4. Servers H1, H2, and H3 are single-homed to leaf 1 (VCF-1), leaf 2 (VCF-2), and leaf 3 (the standalone QFX5100), respectively. H4 is multihomed to leaf 1 and leaf 2.

## Configuration

This section covers the CLI configuration steps to setup EVPN-VXLAN on a Virtual Chassis Fabric (VCF) using QFX5100 devices, and on a standalone QFX5100 device.



**NOTE:** This section shows the step-by-step configuration procedures for each device. All-in-one quick configurations for each device are provided in [“Appendix: Quick Configurations for Configuration Example”](#) on page 31.

The configuration is divided into three general steps:

- [Interface and System Configuration on page 12](#)
- [Underlay Configuration on page 15](#)
- [Overlay Configuration on page 17](#)

### Interface and System Configuration

#### Step-by-Step Procedure

#### Leaf 1 (VCF-1)

1. Configure the uplink interfaces towards the spine devices.

```
set interfaces et-0/0/10 unit 0 description "to Spine 1"
set interfaces et-0/0/10 unit 0 family inet address 192.168.1.1/24
set interfaces et-0/0/11 unit 0 description "to Spine 2"
set interfaces et-0/0/11 unit 0 family inet address 192.168.2.1/24
```

2. Configure the loopback interface, router ID, and autonomous system number.

```
set interfaces lo0 unit 0 family inet address 192.168.10.1/32
set routing-options router-id 192.168.10.1
set routing-options autonomous-system 65100
```

3. Configure the interface towards the single-homed server, H1.

```
set interfaces xe-6/0/40 description "to Server H1"
set interfaces xe-6/0/40 encapsulation ethernet-bridge
set interfaces xe-6/0/40 unit 0 family ethernet-switching vlan members vlan10
```

4. Configure the interface towards the multihomed server, H4.

Use the same ESI value on all leaf devices where the server is multihomed.

```
set chassis aggregated-devices ethernet device-count 1
set interfaces et-0/0/15 description "to Server H4"
set interfaces et-0/0/15 ether-options 802.3ad ae0
set interfaces ae0 encapsulation ethernet-bridge
set interfaces ae0 esi 00:01:01:01:01:01:01:01
set interfaces ae0 esi all-active
set interfaces ae0 unit 0 family ethernet-switching vlan members vlan10
```

5. Configure LACP on the AE interface.

Use the same system ID value on all leaf devices where the server is multihomed.

```
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp periodic fast
set interfaces ae0 aggregated-ether-options lacp system-id 00:00:00:01:01:01
```

### Step-by-Step Procedure

#### Leaf 2 (VCF-2)

1. Configure the uplink interfaces towards the spine devices.

```
set interfaces et-0/0/20 unit 0 description "to Spine 1"
set interfaces et-0/0/20 unit 0 family inet address 192.168.3.1/24
set interfaces et-0/0/21 unit 0 description "to Spine 2"
set interfaces et-0/0/21 unit 0 family inet address 192.168.4.1/24
```

2. Configure the loopback interface, router ID, and autonomous system number.

```
set interfaces lo0 unit 0 family inet address 192.168.20.1/32
set routing-options router-id 192.168.20.1
set routing-options autonomous-system 65300
```

3. Configure the interface towards the single-homed server, H2.

```
set interfaces xe-2/0/16:0 description "to Server H2"
set interfaces xe-2/0/16:0 encapsulation ethernet-bridge
set interfaces xe-2/0/16:0 unit 0 family ethernet-switching vlan members vlan10
```

4. Configure the interface towards the multihomed server, H4.

Use the same ESI value on all leaf devices where the server is multihomed.

```
set chassis aggregated-devices ethernet device-count 1
set interfaces et-0/0/17 description "to Server H4"
set interfaces et-0/0/17 ether-options 802.3ad ae0
set interfaces ae0 encapsulation ethernet-bridge
set interfaces ae0 esi 00:01:01:01:01:01:01:01:01:01
set interfaces ae0 esi all-active
set interfaces ae0 unit 0 family ethernet-switching vlan members vlan10
```

5. Configure LACP on the AE interface.

Use the same system ID value on all leaf devices where the server is multihomed.

```
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp periodic fast
set interfaces ae0 aggregated-ether-options lacp system-id 00:00:00:01:01:01
```

**Step-by-Step Procedure**    **Leaf 3 (Standalone QFX5100)**

1. Configure the uplink interfaces towards the spine devices.

```
set interfaces et-0/0/18 unit 0 description "to Spine 1"  
set interfaces et-0/0/18 unit 0 family inet address 192.168.5.1/24  
set interfaces et-0/0/19 unit 0 description "to Spine 2"  
set interfaces et-0/0/19 unit 0 family inet address 192.168.6.1/24
```

2. Configure the loopback interface, router ID, and autonomous system number.

```
set interfaces lo0 unit 0 family inet address 192.168.30.1/32  
set routing-options router-id 192.168.30.1  
set routing-options autonomous-system 65000
```

3. Configure the interface towards the single-homed server, H3.

```
set interfaces xe-0/0/12:0 description "to Server H3"  
set interfaces xe-0/0/12:0 encapsulation ethernet-bridge  
set interfaces xe-0/0/12:0 unit 0 family ethernet-switching vlan members vlan10
```

**Step-by-Step Procedure**    **Spine1 (QFX10K-1)**

1. Configure the downstream interfaces towards the leaf devices.

```
set interfaces et-0/0/4 unit 0 description "to Leaf 1"  
set interfaces et-0/0/4 unit 0 family inet address 192.168.1.100/24  
set interfaces et-0/0/5 unit 0 description "to Leaf 2"  
set interfaces et-0/0/5 unit 0 family inet address 192.168.3.100/24  
set interfaces et-0/0/18 unit 0 description "to Leaf 3"  
set interfaces et-0/0/18 unit 0 family inet address 192.168.5.100/24
```

2. Configure the loopback interface, router ID, and autonomous system number.

```
set interfaces lo0 unit 0 family inet address 192.168.40.1/32  
set routing-options router-id 192.168.40.1  
set routing-options autonomous-system 65200
```

**Step-by-Step Procedure**    **Spine 2 (QFX10k-2)**

1. Configure the downstream interfaces towards the leaf devices.

```
set interfaces et-0/0/11 unit 0 description "to Leaf 1"  
set interfaces et-0/0/11 unit 0 family inet address 192.168.2.100/24  
set interfaces et-0/0/21 unit 0 description "to Leaf 2"  
set interfaces et-0/0/21 unit 0 family inet address 192.168.4.100/24  
set interfaces et-0/0/19 unit 0 description "to Leaf 3"
```

```
set interfaces et-0/0/19 unit 0 family inet address 192.168.6.100/24
```

2. Configure the loopback interface, router ID, and autonomous system number.

```
set interfaces lo0 unit 0 family inet address 192.168.50.1/32
set routing-options router-id 192.168.50.1
set routing-options autonomous-system 65500
```

## Underlay Configuration

**Step-by-Step Procedure** This section includes configuration of EBGP sessions between each of the leaf and spine devices.

### Leaf 1 (VCF-1)

1. Configure an EBGP session with each spine device.

Since each EBGP neighbor belongs to a separate AS, specify the **multipath multiple-as** parameter to enable multipathing.

```
set protocols bgp group underlay type external
set protocols bgp group underlay export directs
set protocols bgp group underlay multipath multiple-as
set protocols bgp group underlay description "to Spine 1/2"
set protocols bgp group underlay neighbor 192.168.1.100 peer-as 65200
set protocols bgp group underlay neighbor 192.168.2.100 peer-as 65500
```

2. Define a policy to export directly connected subnets for underlay connectivity.

```
set policy-options policy-statement directs term 1 from protocol direct
set policy-options policy-statement directs term 1 then accept
```

**Step-by-Step Procedure** Leaf 2 (VCF-2)

1. Configure an EBGP session with each spine device.

```
set protocols bgp group underlay type external
set protocols bgp group underlay export directs
set protocols bgp group underlay multipath multiple-as
set protocols bgp group underlay description "to Spine 1/2"
set protocols bgp group underlay neighbor 192.168.3.100 peer-as 65200
set protocols bgp group underlay neighbor 192.168.4.100 peer-as 65500
```

2. Define a policy to export directly connected subnets for underlay connectivity.

```
set policy-options policy-statement directs term 1 from protocol direct
set policy-options policy-statement directs term 1 then accept
```

**Step-by-Step Procedure**    **Leaf 3 (Standalone QFX5100)**

1. Configure an EBGP session with each spine device.

```
set protocols bgp group underlay type external
set protocols bgp group underlay export directs
set protocols bgp group underlay multipath multiple-as
set protocols bgp group underlay description "to Spine 1/2"
set protocols bgp group underlay neighbor 192.168.5.100 peer-as 65200
set protocols bgp group underlay neighbor 192.168.6.100 peer-as 65500
```

2. Define a policy to export directly connected subnets for underlay connectivity.

```
set policy-options policy-statement directs term 1 from protocol direct
set policy-options policy-statement directs term 1 then accept
```

**Step-by-Step Procedure**    **Spine 1 (QFX10k-1)**

1. Configure an EBGP session with each leaf device.

```
set protocols bgp group underlay type external
set protocols bgp group underlay export directs
set protocols bgp group underlay multipath multiple-as
set protocols bgp group underlay description "to Leaf 1/2/3"
set protocols bgp group underlay neighbor 192.168.1.1 peer-as 65100
set protocols bgp group underlay neighbor 192.168.3.1 peer-as 65300
set protocols bgp group underlay neighbor 192.168.5.1 peer-as 65000
```

2. Define a policy to export directly connected subnets for underlay connectivity.

```
set policy-options policy-statement directs term 1 from protocol direct
set policy-options policy-statement directs term 1 then accept
```

**Step-by-Step Procedure**    **Spine 2 (QFX10k-2)**

1. Configure an EBGP session with each leaf device.

```
set protocols bgp group underlay type external
set protocols bgp group underlay export directs
set protocols bgp group underlay multipath multiple-as
set protocols bgp group underlay description "to Leaf 1/2/3"
set protocols bgp group underlay neighbor 192.168.2.1 peer-as 65100
set protocols bgp group underlay neighbor 192.168.4.1 peer-as 65300
set protocols bgp group underlay neighbor 192.168.6.1 peer-as 65000
```

2. Define a policy to export directly connected subnets for underlay connectivity.



```
set policy-options policy-statement directs term 1 from protocol direct
set policy-options policy-statement directs term 1 then accept
```

## Overlay Configuration

**Step-by-Step Procedure** This section includes configuration of EVPN between the leaf (PE) devices. The overlay is configured as a single AS, using MP-IBGP for peering.

### Leaf 1 (VCF-1)

1. Setup EVPN (MP-IBGP) sessions with the other leaf devices.

```
set protocols bgp group overlay type internal
set protocols bgp group overlay local-address 192.168.10.1
set protocols bgp group overlay family evpn signaling
set protocols bgp group overlay local-as 65400
set protocols bgp group overlay multipath
set protocols bgp group overlay neighbor 192.168.20.1
set protocols bgp group overlay neighbor 192.168.30.1
```

2. Specify the loopback interface as the source address for the VTEP tunnel.

```
set switch-options vtep-source-interface lo0.0
```

3. Specify a route distinguisher to uniquely identify routes sent from this device.

```
set switch-options route-distinguisher 192.168.10.1:1
```

4. Specify the VRF import policy.

```
set switch-options vrf-import LEAF-IN
```

5. Specify the global VRF export policy.

```
set switch-options vrf-target target:9999:9999
```

6. Specify explicit route targets for the relevant VNI(s).

```
set protocols evpn vni-options vni 10 vrf-target export target:1:10
```

7. Set VXLAN as the data plane encapsulation for EVPN.

```
set protocols evpn encapsulation vxlan
```

8. Specify the relevant VNI(s) to be advertised by EVPN.

```
set protocols evpn extended-vni-list 10
```

9. Set the replication mode for multicast traffic.

```
set protocols evpn multicast-mode ingress-replication
```

10. Define the VRF import policy to accept EVPN routes advertised from the other leaf devices.

```
set policy-options policy-statement LEAF-IN term import_leaf_esi from community
  comm-leaf_esi
set policy-options policy-statement LEAF-IN term import_leaf_esi then accept
set policy-options policy-statement LEAF-IN term import_vni10 from community
  com10
set policy-options policy-statement LEAF-IN term import_vni10 then accept
```

11. Define community values.

```
set policy-options community com10 members target:1:10
set policy-options community comm-leaf_esi members target:9999:9999
```

12. Configure load balancing.

```
set policy-options policy-statement loadbalance then load-balance per-packet
set routing-options forwarding-table export loadbalance
```

13. Define vlan10, the VLAN assigned to the server-facing ports, and map it to a VNI.

```
set vlans vlan10 vlan-id 10
set vlans vlan10 vxlan vni 10
set vlans vlan10 vxlan ingress-node-replication
```

### Step-by-Step Procedure

#### Leaf 2 (VCF-2)

1. Setup EVPN (MP-IBGP) sessions with the other leaf devices.

```
set protocols bgp group overlay type internal
set protocols bgp group overlay local-address 192.168.20.1
set protocols bgp group overlay family evpn signaling set protocols
set protocols bgp group overlay local-as 65400
set protocols bgp group overlay multipath
set protocols bgp group overlay neighbor 192.168.10.1
set protocols bgp group overlay neighbor 192.168.30.1
```

2. Specify the loopback interface as the source address for the VTEP tunnel.

```
set switch-options vtep-source-interface lo0.0
```

3. Specify a route distinguisher to uniquely identify routes sent from this device.

```
set switch-options route-distinguisher 192.168.20.1:1
```

4. Specify the VRF import policy.

```
set switch-options vrf-import LEAF-IN
```

5. Specify the global VRF export policy.

```
set switch-options vrf-target target:9999:9999
```

6. Specify explicit route targets for the relevant VNI(s).

```
set protocols evpn vni-options vni 10 vrf-target export target:1:10
```

7. Set VXLAN as the data plane encapsulation for EVPN.

```
set protocols evpn encapsulation vxlan
```

8. Specify the relevant VNI(s) to be advertised by EVPN.

```
set protocols evpn extended-vni-list 10
```

9. Set the replication mode for multicast traffic.

```
set protocols evpn multicast-mode ingress-replication
```

10. Define the VRF import policy to accept EVPN routes advertised from the other leaf devices.

```
set policy-options policy-statement LEAF-IN term import_leaf_esi from community  
  comm-leaf_esi  
set policy-options policy-statement LEAF-IN term import_leaf_esi then accept  
set policy-options policy-statement LEAF-IN term import_vni10 from community  
  com10  
set policy-options policy-statement LEAF-IN term import_vni10 then accept
```

11. Define community values.

```
set policy-options community com10 members target:1:10  
set policy-options community comm-leaf_esi members target:9999:9999
```

12. Configure load balancing.

```
set policy-options policy-statement loadbalance then load-balance per-packet
set routing-options forwarding-table export loadbalance
```

13. Define vlan10, the VLAN assigned to the server-facing ports, and map it to a VNI.

```
set vlans vlan10 vlan-id 10
set vlans vlan10 vxlan vni 10
set vlans vlan10 vxlan ingress-node-replication
```

### Step-by-Step Procedure

#### Leaf 3 (Standalone QFX5100)

1. Setup EVPN (MP-IBGP) sessions with the other leaf devices.

```
set protocols bgp group overlay type internal
set protocols bgp group overlay local-address 192.168.30.1
set protocols bgp group overlay family evpn signaling
set protocols bgp group overlay local-as 65400
set protocols bgp group overlay multipath
set protocols bgp group overlay neighbor 192.168.20.1
set protocols bgp group overlay neighbor 192.168.10.1
```

2. Specify the loopback interface as the source address for the VTEP tunnel.

```
set switch-options vtep-source-interface lo0.0
```

3. Specify a route distinguisher to uniquely identify routes sent from this device.

```
set switch-options route-distinguisher 192.168.30.1:1
```

4. Specify the VRF import policy.

```
set switch-options vrf-import LEAF-IN
```

5. Specify the global VRF export policy.

```
set switch-options vrf-target target:9999:9999
```

6. Specify explicit route targets for the relevant VNI(s).

```
set protocols evpn vni-options vni 10 vrf-target export target:1:10
```

7. Set VXLAN as the data plane encapsulation for EVPN.

```
set protocols evpn encapsulation vxlan
```

8. Specify the relevant VNI(s) to be advertised by EVPN.

```
set protocols evpn extended-vni-list 10
```

9. Set the replication mode for multicast traffic.

```
set protocols evpn multicast-mode ingress-replication
```

10. Define the VRF import policy to accept EVPN routes advertised from the other leaf devices.

```
set policy-options policy-statement LEAF-IN term import_leaf_esi from community
  comm-leaf_esi
set policy-options policy-statement LEAF-IN term import_leaf_esi then accept
set policy-options policy-statement LEAF-IN term import_vni10 from community
  com10
set policy-options policy-statement LEAF-IN term import_vni10 then accept
```

11. Define community values.

```
set policy-options community com10 members target:1:10
set policy-options community comm-leaf_esi members target:9999:9999
```

12. Configure load balancing.

```
set policy-options policy-statement loadbalance then load-balance per-packet
set routing-options forwarding-table export loadbalance
```

13. Define vlan10, the VLAN assigned to the server-facing ports, and map it to a VNI.

```
set vlans vlan10 vlan-id 10
set vlans vlan10 vxlan vni 10
set vlans vlan10 vxlan ingress-node-replication
```

## Verification

Confirm that the configuration is working properly.

- [Leaf 1 \(VCF-1\): Verifying BGP Sessions on page 22](#)
- [Leaf 1 \(VCF-1\): Verifying EVPN Database Information on page 22](#)
- [Leaf 1 \(VCF-1\): Verifying Local Switching Table Information on page 23](#)
- [Leaf 2 \(VCF-2\): Verifying BGP Sessions on page 24](#)

- Leaf 2 (VCF-2): Verifying EVPN Database Information on page 25
- Leaf 2 (VCF-2): Verifying Local Switching Table Information on page 26
- Leaf 3 (Standalone QFX5100): Verifying BGP Sessions on page 27
- Leaf 3 (Standalone QFX5100): Verifying EVPN Database Information on page 27
- Leaf 3 (Standalone QFX5100): Verifying Local Switching Table Information on page 28
- Verifying Reachability Between End Hosts on page 29

### Leaf 1 (VCF-1): Verifying BGP Sessions

**Purpose** Verify the state of the BGP sessions with the spine devices and other leaf devices.

**Action** Verify that EBGp sessions are established with the spine devices (192.168.1.100 and 192.168.2.100), and IBGP sessions are established with the other leaf devices (192.168.20.1 and 192.168.30.1).

```
{master:0}
user@VCF-1> show bgp summary
Groups: 2 Peers: 4 Down peers: 0
Table Tot Paths Act Paths Suppressed History Damp State Pending
bgp.evpn.0
      7      7      0      0      0      0
inet.0
      34     17      0      0      0      0
Peer AS InPkt OutPkt OutQ Flaps Last Up/Dwn
State|#Active/Received/Accepted/Damped...
192.168.1.100 65200 5142 5124 0 0 1d 14:16:13
Establ
  inet.0: 9/17/17/0
192.168.2.100 65500 5150 5126 0 0 1d 14:16:12
Establ
  inet.0: 8/17/17/0
192.168.20.1 65400 5072 5062 0 0 1d 14:16:09
Establ
  bgp.evpn.0: 5/5/5/0
  default-switch.evpn.0: 4/4/4/0
  __default_evpn__.evpn.0: 1/1/1/0
192.168.30.1 65400 5057 5061 0 0 1d 14:16:01
Establ
  bgp.evpn.0: 2/2/2/0
  default-switch.evpn.0: 2/2/2/0
  __default_evpn__.evpn.0: 0/0/0/0
```

**Meaning** The EBGp underlay sessions are established with the spine devices to form the underlay layer and enable connectivity for the leaf devices. The IBGP sessions are established with the other leaf devices using MP-IBGP with EVPN signaling to form the overlay layer and enable the exchange of EVPN routes.

### Leaf 1 (VCF-1): Verifying EVPN Database Information

**Purpose** Verify that the EVPN database has been populated correctly.



vlan10	00:10:94:00:00:08	D	xe-6/0/40.0	
vlan10	00:10:94:00:00:09	D	vtep.32769	192.168.20.1
vlan10	00:10:94:00:00:0a	D	vtep.32770	192.168.30.1
vlan10	30:7c:5e:8e:f2:50	DL	ae0.0	

**Meaning** The output above confirms that the local switching table is correctly learning and installing MAC addresses for all endpoints. It also shows the relationship between MAC addresses, VLANs they are associated to (in this case, vlan10), and their next-hop interface.

The first and fourth entries show that packets destined for servers H1 and H4 will exit from local interfaces xe-6/0/40 and ae0, respectively.

The second and third entries show that the next hop for packets destined to remote servers H2 and H3 is a VTEP interface. This indicates that to reach these end hosts, the switch will need to encapsulate the packets and send them through the VXLAN tunnel to the appropriate remote leaf (PE) device.

## Leaf 2 (VCF-2): Verifying BGP Sessions

**Purpose** Verify the state of the BGP sessions with the spine devices and other leaf devices.

**Action** Verify that EBGP sessions are established with the spine devices (192.168.3.100 and 192.168.4.100), and IBGP sessions are established with the other leaf devices (192.168.10.1 and 192.168.30.1).

```
{master:0}
user@VCF-2> show bgp summary
Groups: 2 Peers: 4 Down peers: 0
Table          Tot Paths  Act Paths  Suppressed    History  Damp  State    Pending
inet.0
                35         18         0             0         0     0         0
bgp.evpn.0
                8          8          0             0         0     0         0
Peer           AS         InPkt     OutPkt     OutQ     Flaps  Last Up/Dwn
State|#Active/Received/Accepted/Damped...
192.168.3.100  65200     22372     22061      0         0 6d 16:52:27
Establ
  inet.0: 9/17/17/0
192.168.4.100  65500     22387     22100      0         0 6d 16:52:27
Establ
  inet.0: 9/18/18/0
192.168.10.1   65400     5067      5078       0         62 1d 14:18:14
Establ
  bgp.evpn.0: 6/6/6/0
  default-switch.evpn.0: 5/5/5/0
  __default_evpn__.evpn.0: 1/1/1/0
192.168.30.1   65400     21255     21252      0         0 6d 16:52:19
Establ
  bgp.evpn.0: 2/2/2/0
  default-switch.evpn.0: 2/2/2/0
  __default_evpn__.evpn.0: 0/0/0/0
```





VNIs they are associated to (in this case, VNI 10), and the leaf (PE) devices from which they are learned.

In the top output above, the first and third entries are MAC addresses for servers H1 (00:10:94:00:00:08) and H3 (00:10:94:00:00:0a), which have been learned from leaf 1 (192.168.10.1) and leaf 3 (192.168.30.1), respectively.

The second entry shows that server H2's MAC address (00:10:94:00:00:09) has been learned from local interface xe-2/0/16:0.

The fourth entry shows server H4's MAC address (30:7c:5e:8e:f2:50) and confirms it is a multihomed host, as evidenced by the presence of the Ethernet Segment Identifier (ESI). This entry can also be seen on leaf 1 above. Further, the lower output above shows that server H4 has been learned from local interface ae0.

### Leaf 2 (VCF-2): Verifying Local Switching Table Information

**Purpose** Verify that the local switching table has been populated correctly.

**Action** Verify that the local switching table is installing MAC address information for locally attached hosts, and receiving advertisements from the other leaf devices with information about remote hosts.

```
{master:0}
user@VCF-2> show ethernet-switching table

MAC flags (S - static MAC, D - dynamic MAC, L - locally learned, P - Persistent
static
          SE - statistics enabled, NM - non configured MAC, R - remote PE MAC,
0 - ovsdb MAC)

Ethernet switching table : 4 entries, 4 learned
Routing instance : default-switch
  Vlan      MAC          MAC      Logical    Active
  name      address      flags    interface  source
  vlan10    00:10:94:00:00:08  D        vtep.32770  192.168.10.1
  vlan10    00:10:94:00:00:09  D        xe-2/0/16:0.0
  vlan10    00:10:94:00:00:0a  D        vtep.32769  192.168.30.1
  vlan10    30:7c:5e:8e:f2:50  DR       ae0.0
```

**Meaning** The output above confirms that the local switching table is correctly learning and installing MAC addresses for all endpoints. It also shows the relationship between MAC addresses, VLANs they are associated to (in this case, vlan10), and their next-hop interface.

The second and fourth entries show that packets destined for servers H2 and H4 will exit from local interfaces xe-2/0/16:0 and ae0, respectively.

The first and third entries show that the next hop for packets destined to remote servers H1 and H3 is a VTEP interface. This indicates that to reach these end hosts, the switch will need to encapsulate the packets and send them through the VXLAN tunnel to the appropriate remote leaf (PE) device.

### Leaf 3 (Standalone QFX5100): Verifying BGP Sessions

**Purpose** Verify the state of the BGP sessions with the spine devices and other leaf devices.

**Action** Verify that EBGP sessions are established with the spine devices (192.168.5.100 and 192.168.6.100), and IBGP sessions are established with the other leaf devices (192.168.10.1 and 192.168.20.1).

```
{master:0}
user@QFX5100> show bgp summary
Groups: 2 Peers: 4 Down peers: 0
Table          Tot Paths  Act Paths  Suppressed    History  Damp State    Pending
bgp.evpn.0
              9           9           0             0         0         0         0
inet.0
              28          19           0             0         0         0         0
Peer          AS      InPkt    OutPkt    OutQ    Flaps  Last Up/Dwn
State|#Active/Received/Accepted/Damped...
192.168.5.100 65200   22453   22070     0       0 6d 17:05:39
Establ
  inet.0: 10/15/15/0
192.168.6.100 65500   22465   22132     0       0 6d 17:05:44
Establ
  inet.0: 9/13/13/0
192.168.10.1  65400   5095    5092     0       62 1d 14:31:20
Establ
  bgp.evpn.0: 5/5/5/0
  default-switch.evpn.0: 5/5/5/0
  __default_evpn__.evpn.0: 0/0/0/0
192.168.20.1  65400   21282   21286     0       0 6d 17:05:22
Establ
  bgp.evpn.0: 4/4/4/0
  default-switch.evpn.0: 4/4/4/0
  __default_evpn__.evpn.0: 0/0/0/0
```

**Meaning** The EBGP underlay sessions are established with the spine devices to form the underlay layer and enable connectivity for the leaf devices. The IBGP sessions are established with the other leaf devices using MP-IBGP with EVPN signaling to form the overlay layer and enable the exchange of EVPN routes.

### Leaf 3 (Standalone QFX5100): Verifying EVPN Database Information

**Purpose** Verify that the EVPN database has been populated correctly.

**Action** Verify that the EVPN database is installing MAC address information for locally attached hosts, and receiving advertisements from the other leaf devices with information about remote hosts.

```
{master:0}
user@QFX5100> show evpn database
```



**Action** Verify that the local switching table is installing MAC address information for locally attached hosts, and receiving advertisements from the other leaf devices with information about remote hosts.

```
{master:0}
user@QFX5100> show ethernet-switching table

MAC flags (S - static MAC, D - dynamic MAC, L - locally learned, P - Persistent
static
          SE - statistics enabled, NM - non configured MAC, R - remote PE MAC,
O - ovsdb MAC)

Ethernet switching table : 3 entries, 3 learned
Routing instance : default-switch
  Vlan          MAC          MAC          Logical      Active
  name          address      flags        interface    source
  vlan10       00:10:94:00:00:08  D          vtep.32768   192.168.10.1
  vlan10       00:10:94:00:00:09  D          vtep.32769   192.168.20.1
  vlan10       00:10:94:00:00:0a  D          xe-0/0/12:0.0
```

**Meaning** The output above confirms that the local switching table is correctly learning and installing MAC addresses for all endpoints. It also shows the relationship between MAC addresses, VLANs they are associated to (in this case, vlan10), and their next-hop interface.

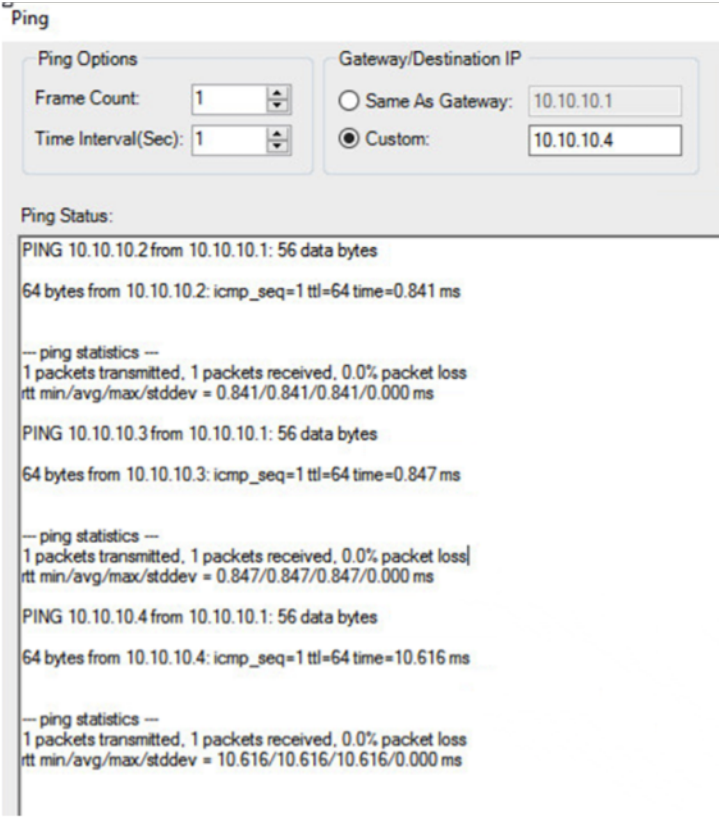
The third entry shows that packets destined for server H3 will exit from local interface xe-0/0/12:0.

The first and second entries show that the next hop for packets destined to remote servers H1 and H2 is a VTEP interface. This indicates that to reach these end hosts, the switch will need to encapsulate the packets and send them through the VXLAN tunnel to the appropriate remote leaf (PE) device.

### Verifying Reachability Between End Hosts

**Purpose** Verify that there is intrasubnet connectivity between the end host devices.

**Action** Verify that server H1 can ping servers H2 (10.10.10.2), H3 (10.10.10.3), and H4 (10.10.10.4).



```
Ping
Ping Options
Frame Count: 1
Time Interval(Sec): 1
Gateway/Destination IP
Same As Gateway: 10.10.10.1
Custom: 10.10.10.4
Ping Status:
PING 10.10.10.2 from 10.10.10.1: 56 data bytes
64 bytes from 10.10.10.2: icmp_seq=1 ttl=64 time=0.841 ms
-- ping statistics --
1 packets transmitted, 1 packets received, 0.0% packet loss
rtt min/avg/max/stddev = 0.841/0.841/0.841/0.000 ms
PING 10.10.10.3 from 10.10.10.1: 56 data bytes
64 bytes from 10.10.10.3: icmp_seq=1 ttl=64 time=0.847 ms
-- ping statistics --
1 packets transmitted, 1 packets received, 0.0% packet loss
rtt min/avg/max/stddev = 0.847/0.847/0.847/0.000 ms
PING 10.10.10.4 from 10.10.10.1: 56 data bytes
64 bytes from 10.10.10.4: icmp_seq=1 ttl=64 time=10.616 ms
-- ping statistics --
1 packets transmitted, 1 packets received, 0.0% packet loss
rtt min/avg/max/stddev = 10.616/10.616/10.616/0.000 ms
```

**Meaning** The servers have reachability to one another.

**Related Documentation**

- [Use Case Overview on page 5](#)
- [Technical Overview on page 6](#)
- [EVPN-VXLAN Support of Virtual Chassis and Virtual Chassis Fabric](#)
- [Virtual Chassis Fabric Feature Guide](#)
- [EVPN Control Plane and VXLAN Data Plane Feature Guide for QFX5100 Switches](#)
- [Juniper Networks EVPN Implementation for Next-Generation Data Center Architectures](#)

## Appendix: Quick Configurations for Configuration Example

This section includes all-in-one quick configurations for each device in the configuration example.

### Interface and System Configuration

#### CLI Quick Configuration

##### Leaf 1 (VCF-1)

```
set interfaces et-0/0/10 unit 0 description "to Spine 1"
set interfaces et-0/0/10 unit 0 family inet address 192.168.1.1/24
set interfaces et-0/0/10 unit 0 description "to Spine 2"
set interfaces et-0/0/11 unit 0 family inet address 192.168.2.1/24
set interfaces lo0 unit 0 family inet address 192.168.10.1/32
set routing-options router-id 192.168.10.1
set routing-options autonomous-system 65100
set interfaces xe-6/0/40 description "to Server H1"
set interfaces xe-6/0/40 encapsulation ethernet-bridge
set interfaces xe-6/0/40 unit 0 family ethernet-switching vlan members vlan10
set chassis aggregated-devices ethernet device-count 1
set interfaces et-0/0/15 description "to Server H4"
set interfaces et-0/0/15 ether-options 802.3ad ae0
set interfaces ae0 encapsulation ethernet-bridge
set interfaces ae0 esi 00:01:01:01:01:01:01:01:01
set interfaces ae0 esi all-active
set interfaces ae0 unit 0 family ethernet-switching vlan members vlan10
set interfaces ae0 aggregated-ether-options lACP active
set interfaces ae0 aggregated-ether-options lACP periodic fast
set interfaces ae0 aggregated-ether-options lACP system-id 00:00:00:01:01:01
```

##### Leaf 2 (VCF-2)

```
set interfaces et-0/0/20 unit 0 description "to Spine 1"
set interfaces et-0/0/20 unit 0 family inet address 192.168.3.1/24
set interfaces et-0/0/21 unit 0 description "to Spine 2"
set interfaces et-0/0/21 unit 0 family inet address 192.168.4.1/24
set interfaces lo0 unit 0 family inet address 192.168.20.1/32
set routing-options router-id 192.168.20.1
set routing-options autonomous-system 65300
set interfaces xe-2/0/16:0 description "to Server H2"
set interfaces xe-2/0/16:0 encapsulation ethernet-bridge
set interfaces xe-2/0/16:0 unit 0 family ethernet-switching vlan members vlan10
set chassis aggregated-devices ethernet device-count 1
set interfaces et-0/0/17 description "to Server H4"
set interfaces et-0/0/17 ether-options 802.3ad ae0
set interfaces ae0 encapsulation ethernet-bridge
set interfaces ae0 esi 00:01:01:01:01:01:01:01:01
set interfaces ae0 esi all-active
set interfaces ae0 unit 0 family ethernet-switching vlan members vlan10
set interfaces ae0 aggregated-ether-options lACP active
set interfaces ae0 aggregated-ether-options lACP periodic fast
```

```
set interfaces ae0 aggregated-ether-options lacp system-id 00:00:00:01:01:01
```

### Leaf 3 (Standalone QFX5100)

```
set interfaces et-0/0/18 unit 0 description "to Spine 1"
set interfaces et-0/0/18 unit 0 family inet address 192.168.5.1/24
set interfaces et-0/0/19 unit 0 description "to Spine 2"
set interfaces et-0/0/19 unit 0 family inet address 192.168.6.1/24
set interfaces lo0 unit 0 family inet address 192.168.30.1/32
set routing-options router-id 192.168.30.1
set routing-options autonomous-system 65000
set interfaces xe-0/0/12:0 description "to Server H3"
set interfaces xe-0/0/12:0 encapsulation ethernet-bridge
set interfaces xe-0/0/12:0 unit 0 family ethernet-switching vlan members vlan10
```

### Spine 1 (QFX10K-1)

```
set interfaces et-0/0/4 unit 0 description "to Leaf 1"
set interfaces et-0/0/4 unit 0 family inet address 192.168.1.100/24
set interfaces et-0/0/5 unit 0 description "to Leaf 2"
set interfaces et-0/0/5 unit 0 family inet address 192.168.3.100/24
set interfaces et-0/0/18 unit 0 description "to Leaf 3"
set interfaces et-0/0/18 unit 0 family inet address 192.168.5.100/24
set interfaces lo0 unit 0 family inet address 192.168.40.1/32
set routing-options router-id 192.168.40.1
set routing-options autonomous-system 65200
```

### Spine 2 (QFX10k-2)

```
set interfaces et-0/0/11 unit 0 description "to Leaf 1"
set interfaces et-0/0/11 unit 0 family inet address 192.168.2.100/24
set interfaces et-0/0/21 unit 0 description "to Leaf 2"
set interfaces et-0/0/21 unit 0 family inet address 192.168.4.100/24
set interfaces et-0/0/19 unit 0 description "to Leaf 3"
set interfaces et-0/0/19 unit 0 family inet address 192.168.6.100/24
set interfaces lo0 unit 0 family inet address 192.168.50.1/32
set routing-options router-id 192.168.50.1
set routing-options autonomous-system 65500
```

## Underlay Configuration

### CLI Quick Configuration

This section includes configuration of EBGP sessions between each of the leaf and spine devices.

#### Leaf 1 (VCF-1)

```
set protocols bgp group underlay type external
set protocols bgp group underlay export directs
set protocols bgp group underlay multipath multiple-as
set protocols bgp group underlay description "to Spine 1/2"
set protocols bgp group underlay neighbor 192.168.1.100 peer-as 65200
set protocols bgp group underlay neighbor 192.168.2.100 peer-as 65500
```



```
set policy-options policy-statement directs term 1 from protocol direct
set policy-options policy-statement directs term 1 then accept
```

#### Leaf 2 (VCF-2)

```
set protocols bgp group underlay type external
set protocols bgp group underlay export directs
set protocols bgp group underlay multipath multiple-as
set protocols bgp group underlay description "to Spine 1/2"
set protocols bgp group underlay neighbor 192.168.3.100 peer-as 65200
set protocols bgp group underlay neighbor 192.168.4.100 peer-as 65500
set policy-options policy-statement directs term 1 from protocol direct
set policy-options policy-statement directs term 1 then accept
```

#### Leaf 3 (Standalone QFX5100)

```
set protocols bgp group underlay type external
set protocols bgp group underlay export directs
set protocols bgp group underlay multipath multiple-as
set protocols bgp group underlay description "to Spine 1/2"
set protocols bgp group underlay neighbor 192.168.5.100 peer-as 65200
set protocols bgp group underlay neighbor 192.168.6.100 peer-as 65500
set policy-options policy-statement directs term 1 from protocol direct
set policy-options policy-statement directs term 1 then accept
```

#### Spine 1 (QFX10K-1)

```
set protocols bgp group underlay type external
set protocols bgp group underlay export directs
set protocols bgp group underlay multipath multiple-as
set protocols bgp group underlay description "to Leaf 1/2/3"
set protocols bgp group underlay neighbor 192.168.1.1 peer-as 65100
set protocols bgp group underlay neighbor 192.168.3.1 peer-as 65300
set protocols bgp group underlay neighbor 192.168.5.1 peer-as 65000
set policy-options policy-statement directs term 1 from protocol direct
set policy-options policy-statement directs term 1 then accept
```

#### Spine 2 (QFX10k-2)

```
set protocols bgp group underlay type external
set protocols bgp group underlay export directs
set protocols bgp group underlay multipath multiple-as
set protocols bgp group underlay description "to Leaf 1/2/3"
set protocols bgp group underlay neighbor 192.168.2.1 peer-as 65100
set protocols bgp group underlay neighbor 192.168.4.1 peer-as 65300
set protocols bgp group underlay neighbor 192.168.6.1 peer-as 65000
set policy-options policy-statement directs term 1 from protocol direct
set policy-options policy-statement directs term 1 then accept
```

## Overlay Configuration

**CLI Quick Configuration** This section includes configuration of EVPN between the leaf (PE) devices. The overlay is configured as a single AS, using MP-IBGP for peering.

### Leaf 1 (VCF-1)

```
set protocols bgp group overlay type internal
set protocols bgp group overlay local-address 192.168.10.1
set protocols bgp group overlay family evpn signaling
set protocols bgp group overlay local-as 65400
set protocols bgp group overlay multipath
set protocols bgp group overlay neighbor 192.168.20.1
set protocols bgp group overlay neighbor 192.168.30.1
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 192.168.10.1:1
set switch-options vrf-import LEAF-IN
set switch-options vrf-target target:9999:9999
set protocols evpn vni-options vni 10 vrf-target export target:1:10
set protocols evpn encapsulation vxlan
set protocols evpn extended-vni-list 10
set protocols evpn multicast-mode ingress-replication
set policy-options policy-statement LEAF-IN term import_leaf_esi from community
  comm-leaf_esi
set policy-options policy-statement LEAF-IN term import_leaf_esi then accept
set policy-options policy-statement LEAF-IN term import_vni10 from community com10
set policy-options policy-statement LEAF-IN term import_vni10 then accept
set policy-options community com10 members target:1:10
set policy-options community comm-leaf_esi members target:9999:9999
set policy-options policy-statement loadbalance then load-balance per-packet
set routing-options forwarding-table export loadbalance
set vlans vlan10 vlan-id 10
set vlans vlan10 vxlan vni 10
set vlans vlan10 vxlan ingress-node-replication
```

### Leaf 2 (VCF-2)

```
set protocols bgp group overlay type internal
set protocols bgp group overlay local-address 192.168.20.1
set protocols bgp group overlay family evpn signaling
set protocols bgp group overlay local-as 65400
set protocols bgp group overlay multipath
set protocols bgp group overlay neighbor 192.168.10.1
set protocols bgp group overlay neighbor 192.168.30.1
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 192.168.20.1:1
set switch-options vrf-import LEAF-IN
set switch-options vrf-target target:9999:9999
set protocols evpn vni-options vni 10 vrf-target export target:1:10
set protocols evpn encapsulation vxlan
set protocols evpn extended-vni-list 10
set protocols evpn multicast-mode ingress-replication
```

```

set policy-options policy-statement LEAF-IN term import_leaf_esi from community
  comm-leaf_esi
set policy-options policy-statement LEAF-IN term import_leaf_esi then accept
set policy-options policy-statement LEAF-IN term import_vni10 from community com10
set policy-options policy-statement LEAF-IN term import_vni10 then accept
set policy-options community com10 members target:1:10
set policy-options community comm-leaf_esi members target:9999:9999
set policy-options policy-statement loadbalance then load-balance per-packet
set routing-options forwarding-table export loadbalance
set vlans vlan10 vlan-id 10
set vlans vlan10 vxlan vni 10
set vlans vlan10 vxlan ingress-node-replication

```

### Leaf 3 (Standalone QFX5100)

```

set protocols bgp group overlay type internal
set protocols bgp group overlay local-address 192.168.30.1
set protocols bgp group overlay family evpn signaling
set protocols bgp group overlay local-as 65400
set protocols bgp group overlay multipath
set protocols bgp group overlay neighbor 192.168.20.1
set protocols bgp group overlay neighbor 192.168.10.1
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 192.168.30.1:1
set switch-options vrf-import LEAF-IN
set switch-options vrf-target target:9999:9999
set protocols evpn vni-options vni 10 vrf-target export target:1:10
set protocols evpn encapsulation vxlan
set protocols evpn extended-vni-list 10
set protocols evpn multicast-mode ingress-replication
set policy-options policy-statement LEAF-IN term import_leaf_esi from community
  comm-leaf_esi
set policy-options policy-statement LEAF-IN term import_leaf_esi then accept
set policy-options policy-statement LEAF-IN term import_vni10 from community com10
set policy-options policy-statement LEAF-IN term import_vni10 then accept
set policy-options community com10 members target:1:10
set policy-options community comm-leaf_esi members target:9999:9999
set policy-options policy-statement loadbalance then load-balance per-packet
set routing-options forwarding-table export loadbalance
set vlans vlan10 vlan-id 10
set vlans vlan10 vxlan vni 10
set vlans vlan10 vxlan ingress-node-replication

```

#### Related Documentation

- [Use Case Overview on page 5](#)
- [Technical Overview on page 6](#)
- [Example: Configuring EVPN-VXLAN on a Virtual Chassis Fabric on page 10](#)
- [EVPN-VXLAN Support of Virtual Chassis and Virtual Chassis Fabric](#)
- [Virtual Chassis Fabric Feature Guide](#)
- [EVPN Control Plane and VXLAN Data Plane Feature Guide for QFX5100 Switches](#)

- [Juniper Networks EVPN Implementation for Next-Generation Data Center Architectures](#)