JUNIPER NETWORKS | Engineering Simplicity

Junos® OS

OVSDB-VXLAN User Guide for QFX Series Switches (VMware NSX)

Published
2025-12-15

JUNOS

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

Junos® OS OVSDB-VXLAN User Guide for QFX Series Switches (VMware NSX)

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

About This Guide

The Open vSwitch Database (OVSDB) management protocol provides a control plane through which QFX Series switches in the physical underlay can exchange control and statistical information with VMware NSX controllers in the virtual overlay. Virtual Extensible LAN (VXLAN) provides a data plane through which Layer 2 data packets can be tunneled over a Layer 3 transport network. Use this guide to learn how OVSDB-VXLAN is implemented on QFX Series switches and to configure, monitor, and troubleshoot OVSDB-VXLAN on these Juniper Networks devices.

You can also use this guide to learn about and configure manual VXLAN, which enables you to manually create VXLANs on QFX Series switches instead of using a controller. If you use this approach, you must also configure Protocol Independent Multicast (PIM), which enables two QFX Series switches to create VXLAN tunnels between themselves.

1

PART

Overview

CHAPTER 1

OVSDB and VXLAN Overview

IN THIS CHAPTER

Understanding the Junos OS Implementation of OVSDB and VXLAN in a VMware NSX for vSphere Environment

Some Juniper Networks devices support Virtual Extensible LAN (VXLAN) and the Open vSwitch Database (OVSDB) management protocol. (See *OVSDB Support on Juniper Networks Devices*.) Support for VXLAN and OVSDB enables the Juniper Networks devices in a physical network to be integrated into a virtual network.

The implementation of VXLAN and OVSDB on Juniper Networks devices is supported in a VMware NSX for NSX for vSphere environment for the data center. Table 1 on page 3 outlines the components that compose this environment and products that are typically deployed for each component.

Table 1: NSX for vSphere Components and Related Products

Component	Products
Cloud management platform (CMP)	CloudStack OpenStack Custom CMP
Network virtualization platform	NSX for vSphere
Hypervisor	Kernel-based Virtual Machine (KVM) Red Hat VMware ESXi Xen **NOTE**: Juniper Networks supports only KVM and ESXi.
Virtual switch	Open vSwitch (OVS) NSX vSwitch
SDN controller	NSX for vSphere controller
Overlay protocol	VXLAN
Media access control (MAC) learning protocol	OVSDB

Figure 1 on page 4 shows a high-level view of the NSX for vSphere platform architecture, while Figure 2 on page 4 provides a more detailed representation of the components in the virtual and physical networks.

Figure 1: High-Level View of NSX for vSphere Architecture

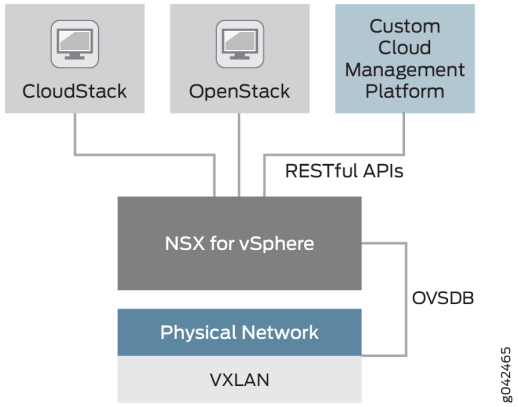
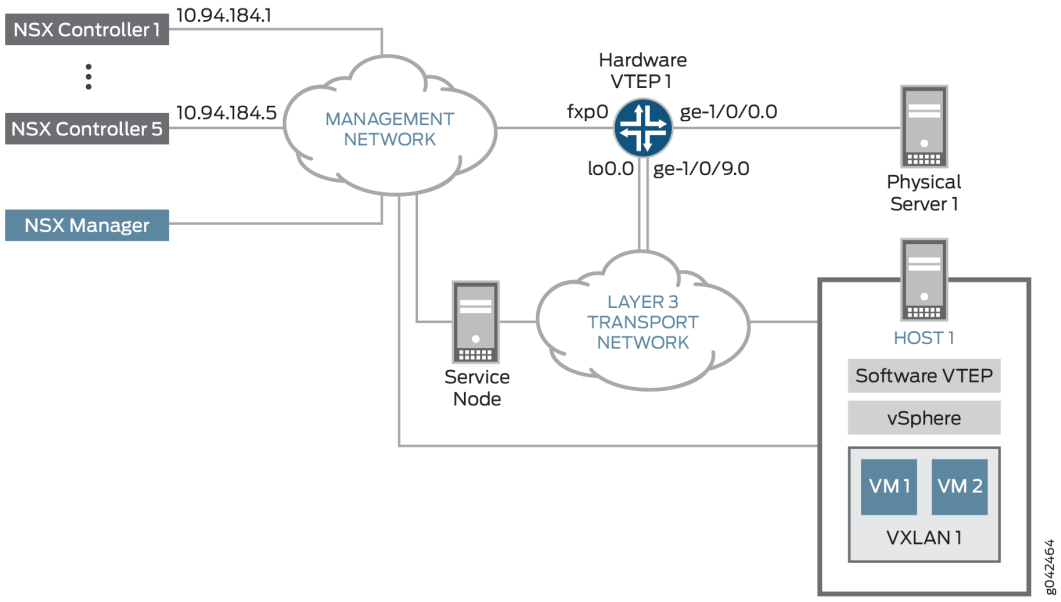


Figure 2: Integration of Juniper Networks Device into NSX for vSphere Environment



In the data center topology shown in Figure 2 on page 4, the physical and virtual servers need to communicate. To facilitate this communication, a Juniper Networks device that supports VXLAN is strategically deployed so that it serves as a *gateway*, which is also known as a hardware *virtual tunnel endpoint (VTEP)*, at the edge of the physical network. Working in conjunction with the software VTEP, which is deployed at the edge of the virtual network, the hardware VTEP encapsulates packets from resources on Physical Server 1 with a VXLAN header, and after the packets traverse the Layer 3 transport network, the software VTEP removes the VXLAN header from the packets and forwards the

packets to the appropriate virtual machines (VMs). In essence, the encapsulation and de-encapsulation of packets by the hardware and software VTEPs enable the components in the physical and virtual networks to coexist without one needing to understand the workings of the other.

The same Juniper Networks device that acts as a hardware VTEP in Figure 2 on page 4 implements OVSDB, which enables this device to learn the MAC addresses of Physical Server 1 and other physical servers, and publish the addresses in the OVSDB schema, which was defined for physical devices. In the virtual network, one or more NSX controllers collect the MAC addresses of Host 1 and other virtual servers, and publish the addresses in the OVSDB schema. Using the OVSDB schema, components in the physical and virtual networks can exchange MAC addresses, as well as statistical information, enabling the components to learn about and reach each other in their respective networks.

RELATED DOCUMENTATION

Understanding the OVSDB Protocol Running on Juniper Networks Devices

OVSDB Schema for Physical Devices

Understanding VXLANs

IN THIS SECTION

Supported VXLAN Standards | 14

Virtual Extensible LAN protocol (VXLAN) technology allows networks to support more VLANs. According to the IEEE 802.1Q standard, traditional VLAN identifiers are 12 bits long—this naming limits networks to 4094 VLANs. The VXLAN protocol overcomes this limitation by using a longer logical network identifier that allows more VLANs and, therefore, more logical network isolation for large networks such as clouds that typically include many virtual machines.

VXLAN Benefits

VXLAN technology allows you to segment your networks (as VLANs do), but it provides benefits that VLANs cannot. Here are the most important benefits of using VXLANs:

- You can theoretically create as many as 16 million VXLANs in an administrative domain (as opposed to 4094 VLANs on a Juniper Networks device).
    - MX Series routers and EX9200 switches support as many as 32,000 VXLANs, 32,000 multicast groups, and 8000 virtual tunnel endpoints (VTEPs). This means that VXLANs based on MX Series routers provide network segmentation at the scale required by cloud builders to support very large numbers of tenants.
    - QFX10000 Series switches support 4000 VXLANs and 2000 remote VTEPs.
    - QFX5100, QFX5110, QFX5200, QFX5210, and EX4600 switches support 4000 VXLANs, 4000 multicast groups, and 2000 remote VTEPs.
    - EX4300-48MP switches support 4000 VXLANs.
- You can enable migration of virtual machines between servers that exist in separate Layer 2 domains by tunneling the traffic over Layer 3 networks. This functionality allows you to dynamically allocate resources within or between data centers without being constrained by Layer 2 boundaries or being forced to create large or geographically stretched Layer 2 domains.

Using VXLANs to create smaller Layer 2 domains that are connected over a Layer 3 network means that you do not need to use Spanning Tree Protocol (STP) to converge the topology but can use more robust routing protocols in the Layer 3 network instead. In the absence of STP, none of your links are blocked, which means you can get full value from all the ports that you purchase. Using routing protocols to connect your Layer 2 domains also allows you to load-balance the traffic to ensure that you get the best use of your available bandwidth. Given the amount of east-west traffic that often flows within or between data centers, maximizing your network performance for that traffic is very important.

The video *Why Use an Overlay Network in a Data Center?* presents a brief overview of the advantages of using VXLANs.

Video: Why Use an Overlay Network in a Data Center?

How Does VXLAN Work?

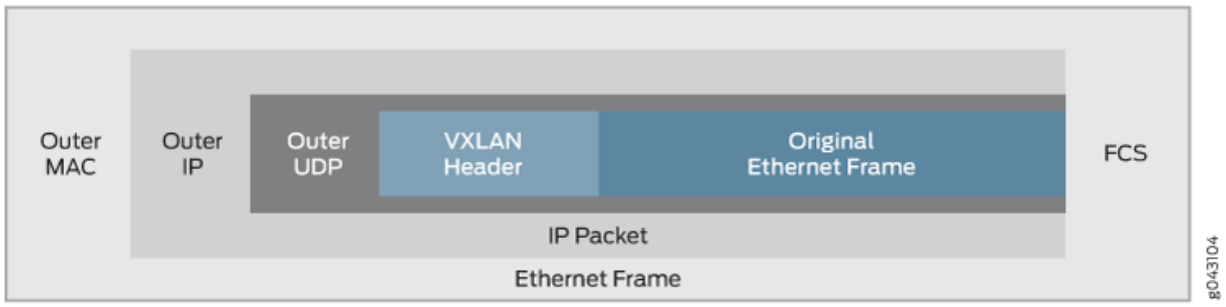
VXLAN is often described as an overlay technology because it allows you to stretch Layer 2 connections over an intervening Layer 3 network by encapsulating (tunneling) Ethernet frames in a VXLAN packet that includes IP addresses. Devices that support VXLANs are called *virtual tunnel endpoints (VTEPs)*—they can be end hosts or network switches or routers. VTEPs encapsulate VXLAN traffic and de-encapsulate that traffic when it leaves the VXLAN tunnel. To encapsulate an Ethernet frame, VTEPs add a number of fields, including the following fields:

- Outer media access control (MAC) destination address (MAC address of the tunnel endpoint VTEP)
- Outer MAC source address (MAC address of the tunnel source VTEP)
- Outer IP destination address (IP address of the tunnel endpoint VTEP)
- Outer IP source address (IP address of the tunnel source VTEP)
- Outer UDP header
- A VXLAN header that includes a 24-bit field—called the *VXLAN network identifier (VNI)*—that is used to uniquely identify the VXLAN. The VNI is similar to a VLAN ID, but having 24 bits allows you to create many more VXLANs than VLANs.

NOTE: Because VXLAN adds 50 to 54 bytes of additional header information to the original Ethernet frame, you might want to increase the MTU of the underlying network. In this case, configure the MTU of the physical interfaces that participate in the VXLAN network, not the MTU of the logical VTEP source interface, which is ignored.

Figure 3 on page 8 shows the VXLAN packet format.

Figure 3: VXLAN Packet Format



VXLAN Implementation Methods

Junos OS supports implementing VXLANs in the following environments:

- Manual VXLAN—In this environment, a Juniper Networks device acts as a transit device for downstream devices acting as VTEPs, or a gateway that provides connectivity for downstream servers that host virtual machines (VMs), which communicate over a Layer 3 network. In this environment, software-defined networking (SDN) controllers are not deployed.

  NOTE: QFX10000 switches do not support manual VXLANs.

- OVSDB-VXLAN—In this environment, SDN controllers use the Open vSwitch Database (OVSDB) management protocol to provide a means through which controllers (such as a VMware NSX or Juniper Networks Contrail controller) and Juniper Networks devices that support OVSDB can communicate.

- EVPN-VXLAN—In this environment, Ethernet VPN (EVPN) is a control plane technology that enables hosts (physical servers and VMs) to be placed anywhere in a network and remain connected to the same logical Layer 2 overlay network, and VXLAN creates the data plane for the Layer 2 overlay network.

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

You can configure the switches to perform all of the following roles:

- (All switches except EX4300-48MP) In an environment without an SDN controller, act as a transit Layer 3 switch for downstream hosts acting as VTEPs. In this configuration, you do not need to configure any VXLAN functionality on the switch. You do need to configure IGMP and PIM so that the switch can form the multicast trees for the VXLAN multicast groups. (See "Manual VXLANs Require PIM " on page 12 for more information.)

- (All switches except EX4300-48MP) In an environment with or without an SDN controller, act as a Layer 2 gateway between virtualized and nonvirtualized networks in the same data center or between data centers. For example, you can use the switch to connect a network that uses VXLANs to one that uses VLANs.
- (EX4300-48MP switches) Act as a Layer 2 gateway between virtualized and nonvirtualized networks in a campus network. For example, you can use the switch to connect a network that uses VXLANs to one that uses VLANs.
- (All switches except EX4300-48MP) Act as a Layer 2 gateway between virtualized networks in the same or different data centers and allow virtual machines to move (VMotion) between those networks and data centers. For example, if you want to allow VMotion between devices in two different networks, you can create the same VLAN in both networks and put both devices on that VLAN. The switches connected to these devices, acting as VTEPs, can map that VLAN to the same VXLAN, and the VXLAN traffic can then be routed between the two networks.
- (QFX5110 and QFX5120 switches with EVPN-VXLAN) Act as a Layer 3 gateway to route traffic between different VXLANs in the same data center.
- (QFX5110 and QFX5120 switches with EVPN-VXLAN) Act as a Layer 3 gateway to route traffic between different VXLANs in different data centers over a WAN or the Internet using standard routing protocols or virtual private LAN service (VPLS) tunnels.

NOTE: If you want a QFX5110 or QFX5120 switch to be a Layer 3 VXLAN gateway in an EVPN-VXLAN environment, you must configure integrated routing and bridging (IRB) interfaces to connect the VXLANs, just as you do if you want to route traffic between VLANs.

Because the additional headers add 50 to 54 bytes, you might need to increase the MTU on a VTEP to accommodate larger packets. For example, if the switch is using the default MTU value of 1514 bytes and you want to forward 1500-byte packets over the VXLAN, you need to increase the MTU to allow for the increased packet size caused by the additional headers.

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

Starting with Junos OS Release 14.1X53-D25 on QFX5100 switches, Junos OS Release 15.1X53-D210 on QFX5110 and QFX5200 switches, Junos OS Release 18.1R1 on QFX5210 switches, and Junos OS Release 18.2R1 on EX4600 switches, you can configure the UDP port used as the destination port for VXLAN traffic. To configure the VXLAN destination port to be something other than the default UDP port of 4789, enter the following statement:

```
set protocols l2-learning destination-udp-port port-number
```

The port you configure will be used for all VXLANs configured on the switch.

NOTE: If you make this change on one switch in a VXLAN, you must make the same change on all the devices that terminate the VXLANs configured on your switch. If you do not do so, traffic will be disrupted for all the VXLANs configured on your switch. When you change the UDP port, the previously learned remote VTEPs and remote MACs are lost and VXLAN traffic is disrupted until the switch relearns the remote VTEPs and remote MACs.

Host Entry Overflow Prevention

IN THIS SECTION

By default, when the host table reaches its capacity, additional host entries overflow into the Longest Prefix Match (LPM) table. This can cause a degradation of routing performance that results from smaller host entries consuming valuable space in the LPM table.

Starting with Junos OS Release 25.2R1, you can prevent host entries from spilling over into the LPM table by configuring the `set forwarding-options no-host-as-lpm` statement. When you enable this option, host entries are blocked from entering the LPM table, and an error message is logged to notify you of the full host table condition. This restriction maintains the LPM table's integrity for larger subnet routes, which are typically more critical for efficient network operation. This configuration requires a PFE restart to clear any existing host entries from the LPM table, ensuring that the table is dedicated solely to subnet routes moving forward. The restart is crucial for maintaining a clean state, as it prevents any residual host entries from affecting table space utilization.

NOTE: Configuring the `set forwarding-options no-host-as-lpm` statement restarts the PFE in standalone devices. Virtual Chassis (VC) devices require a manual restart. Restarting the PFE removes any previously learned host entries from the LPM table.

Host entry overflow prevention is particularly beneficial for environments with high host entry demands. The feature supports both IPv4 and IPv6 routes and applies to VXLAN and non-VXLAN environments, making it versatile for various deployment scenarios. Segregating host and subnet routes avoids the degradation of routing performance that results from host entries spilling over to the LPM table. Additionally, the system generates a system log message each time you toggle the CLI option, providing an audit trail for change management and troubleshooting. By leveraging this feature, you can enhance

routing table management, improve routing efficiency, and ensure systematic control over routing table entries.

Please refer to Feature Explorer for a complete list of the products that support the host entry overflow prevention feature.

CLI Commands

To configure the host entry overflow prevention feature, use the following CLI command:

```
set forwarding-options no-host-as-lpm
```

This command enables the feature that blocks host entries from being added to the LPM table when the host table is full, preserving LPM space for larger subnet routes.

To verify the status of the host entry overflow prevention feature, use the command:

```
show forwarding-options no-host-as-lpm
```

This command displays the current configuration status, indicating whether the feature is enabled or disabled.

To display the a summary of the routes in the Packet Forwarding Engine Host and LPM forwarding tables, use the command:

```
show pfe route summary hw
```

By using these commands, you can effectively manage and monitor the routing table entries, ensuring optimal performance and scalability in your network environment.

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

When the switch acting as a VTEP receives a broadcast, unknown unicast, or multicast packet, it performs the following actions on the packet:

1. It de-encapsulates the packet and delivers it to the locally attached hosts.
2. It then adds the VXLAN encapsulation again and sends the packet to the other VTEPs in the VXLAN.

These actions are performed by the loopback interface used as the VXLAN tunnel address and can, therefore, negatively impact the bandwidth available to the VTEP. Starting with Junos OS Release 14.1X53-D30 for QFX5100 switches, Junos OS Release 15.1X53-D210 for QFX5110 and QFX5200 switches, Junos OS Release 18.1R1 for QFX5210 switches, and Junos OS Release 18.2R1 for EX4600 switches, if you know that there are no multicast receivers attached to other VTEPs in the VXLAN that

want traffic for a specific multicast group, you can reduce the processing load on the loopback interface by entering the following statement:

```
set protocols l2-learning disable-vxlan-multicast-transit vxlan-multicast-group multicast-group
```

In this case, no traffic will be forwarded for the specified group but all other multicast traffic will be forwarded. If you do not want to forward any multicast traffic to other VTEPs in the VXLAN, enter the following statement:

```
set protocols l2-learning disable-vxlan-multicast-transit vxlan-multicast-group all
```

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

You can configure an MX Series router, EX9200 switch, or QFX10000 switch to act as a VTEP and perform all of the following roles:

- Act as a Layer 2 gateway between virtualized and nonvirtualized networks in the same data center or between data centers. For example, you can use an MX Series router to connect a network that uses VXLANs to one that uses VLANs.
- Act as a Layer 2 gateway between virtualized networks in the same or different data centers and allow virtual machines to move (VMotion) between those networks and data centers.
- Act as a Layer 3 gateway to route traffic between different VXLANs in the same data center.
- Act as a Layer 3 gateway to route traffic between different VXLANs in different data centers over a WAN or the Internet using standard routing protocols or virtual private LAN service (VPLS) tunnels.

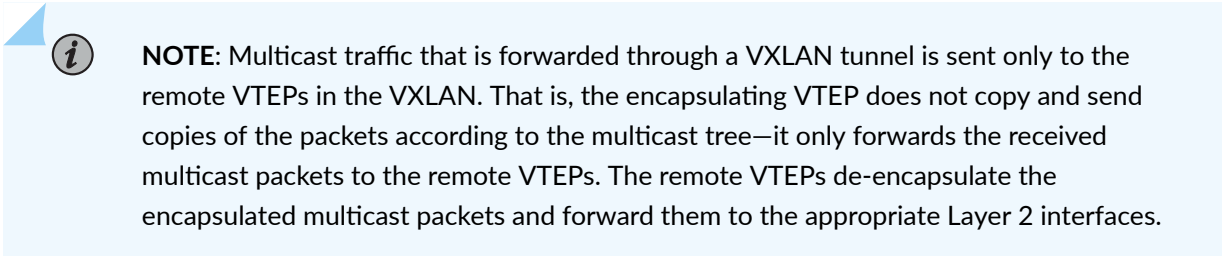
NOTE: If you want one of the devices described in this section to be a VXLAN Layer 3 gateway, you must configure integrated routing and bridging (IRB) interfaces to connect the VXLANs, just as you do if you want to route traffic between VLANs.

Manual VXLANs Require PIM

In an environment with a controller (such as a VMware NSX or Juniper Networks Contrail controller), you can provision VXLANs on a Juniper Networks device. A controller also provides a control plane that VTEPs use to advertise their reachability and learn about the reachability of other VTEPs. You can also manually create VXLANs on Juniper Networks devices instead of using a controller. If you use this approach, you must also configure Protocol Independent Multicast (PIM) on the VTEPs so that they can create VXLAN tunnels between themselves.

You must also configure each VTEP in a given VXLAN to be a member of the same multicast group. (If possible, you should assign a different multicast group address to each VXLAN, although this is not required. Multiple VXLANs can share the same multicast group.) The VTEPs can then forward ARP requests they receive from their connected hosts to the multicast group. The other VTEPs in the group de-encapsulate the VXLAN information, and (assuming they are members of the same VXLAN) they forward the ARP request to their connected hosts. When the target host receives the ARP request, it responds with its MAC address, and its VTEP forwards this ARP reply back to the source VTEP. Through this process, the VTEPs learn the IP addresses of the other VTEPs in the VXLAN and the MAC addresses of the hosts connected to the other VTEPs.

The multicast groups and trees are also used to forward broadcast, unknown unicast, and multicast (BUM) traffic between VTEPs. This prevents BUM traffic from being unnecessarily flooded outside the VXLAN.



NOTE: Multicast traffic that is forwarded through a VXLAN tunnel is sent only to the remote VTEPs in the VXLAN. That is, the encapsulating VTEP does not copy and send copies of the packets according to the multicast tree—it only forwards the received multicast packets to the remote VTEPs. The remote VTEPs de-encapsulate the encapsulated multicast packets and forward them to the appropriate Layer 2 interfaces.

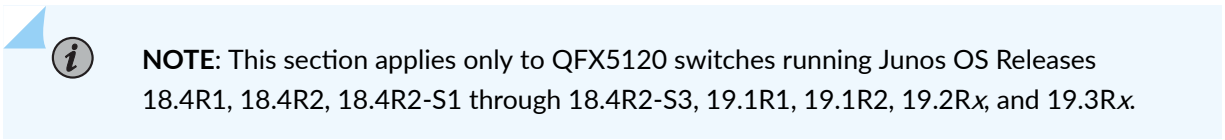
Load Balancing VXLAN Traffic

The Layer 3 routes that form VXLAN tunnels use per-packet load balancing by default, which means that load balancing is implemented if there are ECMP paths to the remote VTEP. This is different from normal routing behavior in which per-packet load balancing is not used by default. (Normal routing uses per-prefix load balancing by default.)

The source port field in the UDP header is used to enable ECMP load balancing of the VXLAN traffic in the Layer 3 network. This field is set to a hash of the inner packet fields, which results in a variable that ECMP can use to distinguish between tunnels (flows).

None of the other fields that flow-based ECMP normally uses are suitable for use with VXLANs. All tunnels between the same two VTEPs have the same outer source and destination IP addresses, and the UDP destination port is set to port 4789 by definition. Therefore, none of these fields provide a sufficient way for ECMP to differentiate flows.

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



NOTE: This section applies only to QFX5120 switches running Junos OS Releases 18.4R1, 18.4R2, 18.4R2-S1 through 18.4R2-S3, 19.1R1, 19.1R2, 19.2R*x*, and 19.3R*x*.

When a QFX5120 switch attempts to tunnel traffic on core-facing Layer 3 tagged interfaces or IRB interfaces, the switch drops the packets. To avoid this issue, you can configure a simple two-term filter-based firewall on the Layer 3 tagged or IRB interface.

NOTE: QFX5120 switches support a maximum of 256 two-term filter-based firewalls.

For example:

```
set interfaces et-0/0/3 unit 0 family inet filter input vxlan100
set firewall family inet filter vxlan100 term 1 from destination-address 192.168.0.1/24 then
accept
set firewall family inet filter vxlan100 term 2 then routing-instance route1
```

Term 1 matches and accepts traffic that is destined for the QFX5210 switch, which is identified by the source VTEP IP address (192.168.0.1/24) assigned to the switch's loopback interface. For term 1, note that when specifying an action, you can alternatively count traffic instead of accepting it.

Term 2 matches and forwards all other data traffic to a routing instance (route 1), which is configured interface et-0/0/3.

In this example, note that interface et-0/0/3 is referenced by routing instance route1. As a result, you must include the `set firewall family inet filter vxlan100 term 2 then routing-instance route1` command. Without this command, the firewall filter will not work properly.

Using ping and traceroute with a VXLAN

On QFX5100 and QFX5110 switches, you can use the `ping` and `traceroute` commands to troubleshoot traffic flow through a VXLAN tunnel by including the `overlay` parameter and various options. You use these options to force the `ping` or `traceroute` packets to follow the same path as data packets through the VXLAN tunnel. In other words, you make the underlay packets (`ping` and `traceroute`) take the same route as the overlay packets (data traffic). See *ping overlay* and *traceroute overlay* for more information.

Supported VXLAN Standards

RFCs and Internet drafts that define standards for VXLAN:

- RFC 7348, *Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks*
- Internet draft draft-ietf-nvo3-vxlan-gpe, *Generic Protocol Extension for VXLAN*

Change History Table

Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
25.2R1	Starting with Junos OS Release, you can prevent host routes from spilling over into the LPM table.
14.1X53-D30	Starting with Junos OS Release 14.1X53-D30 for QFX5100 switches, Junos OS Release 15.1X53-D210 for QFX5110 and QFX5200 switches, Junos OS Release 18.1R1 for QFX5210 switches, and Junos OS Release 18.2R1 for EX4600 switches, if you know that there are no multicast receivers attached to other VTEPs in the VXLAN that want traffic for a specific multicast group, you can reduce the processing load on the loopback interface
14.1X53-D25	Starting with Junos OS Release 14.1X53-D25 on QFX5100 switches, Junos OS Release 15.1X53-D210 on QFX5110 and QFX5200 switches, Junos OS Release 18.1R1 on QFX5210 switches, and Junos OS Release 18.2R1 on EX4600 switches, you can configure the UDP port used as the destination port for VXLAN traffic.

RELATED DOCUMENTATION

Examples: Manually Configuring VXLANs on QFX Series and EX4600 Switches | **101**

Understanding EVPN with VXLAN Data Plane Encapsulation

OVSDB Support on Juniper Networks Devices

mtu (QFX Series)

VXLAN Constraints on EX Series, QFX Series, PTX Series, and ACX Series Devices

IN THIS SECTION

When configuring Virtual Extensible LANs (VXLANs) on QFX Series and EX Series switches, be aware of the constraints described in the following sections. In these sections, “Layer 3 side” refers to a network-facing interface that performs VXLAN encapsulation and de-encapsulation, and “Layer 2 side” refers to a server-facing interface that is a member of a VLAN that is mapped to a VXLAN.

VXLAN Constraints on QFX5xxx, EX4100, EX4100-F, EX4300-48MP, EX4400, and EX4600 Switches

- (QFX5130-32CD and QFX5700 switches) We don't support a centrally routed bridging (CRB) architecture using a collapsed spine model when:
  - An interface uses both enterprise and service provider style configurations at the `edit interfaces` *`interface-name`* hierarchy.
  - Multiple access ports on the same VXLAN bridge domain use a mix of enterprise and service provider styles.

  Enterprise Style Configuration

  ```
  set interfaces interface-name unit 0 family ethernet-switching interface-mode trunk
  set interfaces interface-name unit 0 family ethernet-switching vlan members vlan-name
  ```

  Service Provider Style Configuration

  ```
  set interfaces interface-name vlan-tagging
  set interfaces interface-name encapsulation extended-vlan-bridge
  set interfaces interface-name unit unit vlan-id vlan-id
  ```

  Flexible Ethernet Services Configuration

  ```
  set interfaces interface-name flexible-vlan-tagging
  set interfaces interface-name encapsulation flexible-ethernet-services
  set interfaces interface-name unit unit encapsulation vlan-bridge
  set interfaces interface-name unit unit vlan-id vlan-id
  set interfaces interface-name unit unit family ethernet-switching interface-mode trunk
  set interfaces interface-name unit unit family ethernet-switching vlan members vlan-name
  ```

  Supported scenarios:

  - A non-collapsed spine and no VLAN is configured with a local interface.

- A collapsed spine and some VLAN's on the device are configured without a local interface, while some VLAN's on the device are configured using enterprise style on CE-facing interfaces.
- A collapsed spine and all VLAN's on the device are configured with `flexible-ethernet-services` encapsulation on a physical interface, while multiple logical interfaces are configured with either enterprise or service provider style.

Unsupported scenarios:

- A collapsed spine and any local interface on the device includes all configured VLAN's. The workaround for this scenario is to use `flexible-ethernet-services` encapsulation on the physical interface.
- A collapsed spine where some VLAN's on the device are not configured on any local interface, and some VLAN's are configured using service provider style on CE-facing interfaces. The workaround for this scenario is to configure a `vlan-id`.
- A collapsed spine where some VLAN's on the device are not part of any interface, and some VLAN's are configured using `flexible-ethernet-services` encapsulation on multiple, logical interfaces, using enterprise style. The workaround for this scenario is to configure a `vlan-id`.

- VXLAN support on a Virtual Chassis or Virtual Chassis Fabric (VCF) has the following constraints and recommendations:
  - We support EVPN-VXLAN on an EX4300-48MP Virtual Chassis only in campus networks.
  - Standalone EX4400 switches and EX4400 Virtual Chassis support EVPN-VXLAN. For multihoming use cases, hosts can be multihomed to standalone EX4400 switches, but we don’t support multihoming a host with ESI-LAG interfaces to EX4400 Virtual Chassis.
  - Standalone EX4100 (EX4100 and EX4100-F) switches and EX4100 Virtual Chassis (EX4100 and EX4100-F) support EVPN-VXLAN. For multihoming use cases, hosts can be multihomed to standalone EX4100 switches, but we don't support multihoming a host with ESI-LAG interfaces to EX4100 Virtual Chassis.
  - In data center networks, we support EVPN-VXLAN only on a Virtual Chassis or VCF consisting of all QFX5100 switches, and not on any other mixed or non-mixed Virtual Chassis or VCF. We support VCF in EVPN-VXLAN data center environments running Junos OS releases starting in 14.1X53-D40 and prior to 17.1R1. However, we don't recommend using EVPN-VXLAN on a QFX5100 Virtual Chassis or VCF because the feature support is at parity only with support on standalone QFX5100 switches running Junos OS Release 14.1X53-D40.

    We have deprecated VCF support in general from Junos OS Release 21.4R1 onward.
  - When a QFX5100 Virtual Chassis learns a MAC address on a VXLAN interface, the MAC table entry can possibly take up to 10 to 15 minutes to age out (two to three times the 5-minute default aging interval). This happens when the Virtual Chassis learns a MAC address from an

incoming packet on one Virtual Chassis member switch, and then must forward that packet over the Virtual Chassis port (VCP) links to another member switch in the Virtual Chassis on the way to its destination. The Virtual Chassis marks the MAC address as seen again at the second member switch, so the MAC address might not age out for one or two additional aging intervals beyond the first one. MAC learning can't be turned off on VXLAN interfaces only at the second Virtual Chassis member switch, so you can't avoid the extra delay in this case.

- (QFX5120 switches only) Traffic that is tunneled via a core-facing Layer 3 tagged interface or IRB interface is dropped. To avoid this limitation, you can configure flexible VLAN tagging. For more information, see *Understanding VXLANs*.

- (QFX5110 and QFX5120) If you configure an interface in the enterprise style with flexible Ethernet services encapsulation, the device drops transit Layer 2 VXLAN-encapsulated packets on that interface. To work around this issue, configure the interface in the service provider style instead of using the enterprise style. For more information on enterprise style and service provider style configurations, see *Flexible Ethernet Services Encapsulation*. For an overview of configuring flexible Ethernet services in an EVPN-VXLAN fabric, see *Understanding Flexible Ethernet Services Support With EVPN-VXLAN*.

- (QFX5110 and QFX5120 switches) In EVPN-VXLAN fabrics, we don't support enterprise style, service provider style, and native VLAN configurations on the same physical interface if the native VLAN is the same as one of the VLANs in the service provider style configuration. The native VLAN can be one of the VLANs in the enterprise style configuration. For more information on enterprise style and service provider style configurations, see *Flexible Ethernet Services Encapsulation*.

- (QFX5xxx switches) In an EVPN-VXLAN fabric, you can't configure the *native-vlan-id* statement on the same interface where you enable VLAN translation with the *vlan-rewrite* statement.

- (QFX5100, QFX5110, QFX5200, and QFX5210 switches) We support VXLAN configuration in the default-switch routing instance and in MAC VRF routing instances (`instance-type mac-vrf`).

  (EX4300-48MP and EX4600 switches) We support VXLAN configuration only in the default-switch routing instance.

- (QFX5100, QFX5200, QFX5210, EX4300-48MP, and EX4600 switches) Routing traffic between different VXLANs is not supported.

  NOTE: The following switches support VXLAN routing as of the indicated releases, so this limitation no longer applies:

  - EX4300-48MP switches: Starting in Junos OS Release 19.4R1.
  - QFX5210 switches: Starting in Junos OS Release 21.3R1.

- (QFX5100, QFX5110, QFX5120, EX4600, and EX4650 switches) These switches support only one VTEP next hop on VXLAN underlay IRB interfaces. If you don't want to use multiple egress ports but need more than one VTEP next hop, as a workaround you can do one of the following:
  - Place a router between the switch and the remote VTEPs so only one next hop is between them.
  - Use physical Layer 3 interfaces instead of IRB interfaces for remote VTEP reachability.
- (QFX5110 switches) By default, routing traffic between a VXLAN and a Layer 3 logical interface—for example, an interface configured with the `set interfaces` *`interface-name`* `unit` *`logical-unit-number`* `family inet address` *`ip-address/prefix-length`* command—is not enabled. If this routing functionality is required in your EVPN-VXLAN network, you can perform some additional configuration to make it work. For more information, see *Understanding How to Configure VXLANs and Layer 3 Logical Interfaces to Interoperate*.
- Integrated routing and bridging (IRB) interfaces used in EVPN-VXLAN overlay networks do not support the IS-IS routing protocol.
- (QFX5100, QFX5110, QFX5120, QFX5200, QFX5210, EX4300-48MP, and EX4600 switches) A physical interface cannot be a member of a VLAN and a VXLAN. That is, an interface that performs VXLAN encapsulation and de-encapsulation cannot also be a member of a VLAN. For example, if a VLAN that is mapped to a VXLAN is a member of trunk port xe-0/0/0, any other VLAN that is a member of xe-0/0/0 must also be assigned to a VXLAN.

  NOTE: Starting in Junos OS Releases 18.1R3 and 18.4R1 for QFX5100, QFX5110, QFX5200, and QFX5210 switches and Junos OS Release 19.4R1 for QFX5120 and EX4650 switches, this limitation no longer applies because you can configure flexible Ethernet services on the physical interface. (The same is true for aggregated Ethernet (AE) bundle interfaces.)

  Also, starting in Junos OS Release 20.3R1 on QFX5110 and QFX5120 switches, we support Layer 2 VLAN and VXLAN logical interfaces on the same physical interface using service provider style interface configurations only.

  For more information, see *Understanding Flexible Ethernet Services Support With EVPN-VXLAN*.
- (QFX5110, QFX5120, EX4100, and EX4400 switches) We don't support VXLAN and non-VXLAN logical interfaces on the same physical interface using enterprise style interface configurations.
- (QFX5120, EX4100, EX4300-48MP, EX4400, and EX4650 switches) With 802.1X authentication for VXLAN traffic on an access port, upon authentication, the RADIUS server dynamically switches the traffic from the original VLAN to a dynamic VLAN you configure as a VXLAN-enabled VLAN. A VXLAN-enabled VLAN is a VLAN for which you configure a VXLAN network identifier (VNI) mapping using the `vxlan vni` *`vni`* statement at the `[edit vlans` *`vlan-name`*`]` hierarchy.

When you configure the 802.1X dynamic VLAN and its corresponding VNI mapping, you must also configure the original VLAN as a VXLAN-enabled VLAN with a VNI mapping. If you don't explicitly configure the port as a member of a VLAN, the port uses the default VLAN. In that case, you must configure the default VLAN as a VXLAN-enabled VLAN with a VNI mapping.

Also, in releases before Junos OS Release 22.2R3-S3, when any dynamic VLAN is assigned to a port, that VLAN must already have been statically configured on another port on the device. Starting in Junos OS Release 22.2R3-S3, we no longer have this constraint.

See *802.1X Authentication* and *RADIUS Server Configuration for Authentication* for more on dynamic VLAN assignment with a RADIUS server.

- Multichassis link aggregation groups (MC-LAGs) are not supported with VXLAN.

  NOTE: In an EVPN-VXLAN environment, EVPN multihoming active-active mode is used instead of MC-LAG for redundant connectivity between hosts and leaf devices.

- IP fragmentation and defragmentation are not supported on the Layer 3 side.
- The following features are not supported on the Layer 2 side:
  - (QFX5100, QFX5200, QFX5210, EX4300-48MP, and EX4600 switches) IGMP snooping with EVPN-VXLAN.
  - Redundant trunk groups (RTGs).
  - The ability to shut down a Layer 2 interface or temporarily bring down the interface when a storm control level is exceeded is not supported.
  - We don't support full STP, MSTP, RSTP, or VSTP (xSTP) features with VXLAN. However, you can configure xSTP on edge (access port) for BPDU block-on-edge support. See *BPDU Protection for Spanning-Tree Protocols* for details.
- Access port security features such as the following are not supported with VXLAN:
  - DHCP snooping.
  - Dynamic ARP inspection.
  - MAC limiting and MAC move limiting.

  Some exceptions include:

  - MAC limiting is supported on OVSDB-managed interfaces in an OVSDB-VXLAN environment with Contrail controllers. For more information, see *Features Supported on OVSDB-Managed Interfaces*.

  - On these devices serving as L2 VXLAN gateways in EVPN-VXLAN centrally-routed bridging overlay networks:
    - EX4300 multigigabit switches starting in Junos OS Release 19.4R1
    - EX4400 switches starting in Junos OS Release 21.1R1
    - EX4400 multigigabit switches starting in Junos OS Release 21.2R1
    - EX4100 and EX4100-F switches starting in Junos OS Release 22.3R1
    - EX4100 multigigabit switches starting in Junos OS Release 22.3R1

    we support these access security features on Layer 2 access-side interfaces that are associated with VXLAN-mapped VLANs:
    - DHCPv4 and DHCPv6 snooping
    - Dynamic ARP inspection (DAI)
    - Neighbor discovery inspection (NDI)
    - IPv4 and IPv6 source guard
    - Router advertisement (RA) guard

    We support these features only on single-homed access interface connections.
- Ingress node replication is not supported in the following cases:
  - When PIM is used for the control plane (manual VXLAN).
  - When an SDN controller is used for the control plane (OVSDB-VXLAN).

  Ingress node replication is supported with EVPN-VXLAN.
- PIM-BIDIR and PIM-SSM are not supported with VXLANs.
- If you configure a port-mirroring instance to mirror traffic exiting from an interface that performs VXLAN encapsulation, the source and destination MAC addresses of the mirrored packets are invalid. The original VXLAN traffic is not affected.
- (QFX5110 switches only) VLAN firewall filters are not supported on IRB interfaces on which EVPN-VXLAN is enabled.
- (EX4650 and QFX5000 Series switches) Firewall filter and policer support:
  - (QFX5100 switches) Firewall filters and policers are not supported on transit traffic on which EVPN-VXLAN is enabled. They are supported only in the ingress direction on CE-facing interfaces.

  - (QFX5100 switches) For IRB interfaces in an EVPN-VXLAN one-layer IP fabric, firewall filtering and policing is supported only at the ingress point of non-encapsulated frames routed through the IRB interface.
  - (EX4650, QFX5110, and QFX5120 switches) We support ingress filtering and policing for routed VXLAN interfaces (IRB interfaces) as ingress route Access Control Lists [IRACLs].
  - (QFX5110, QFX5120, and QFX5210 switches) We support Ingress filtering and policing on non-routed VXLAN interfaces as ingress port ACLs [IPACLs]).
  - (QFX5110 and QFX5120 switches) Filtering and policing support on non-routed VXLAN interfaces extends to the egress direction as egress Port ACLs ([EPACLs]).

- (EX4300-48MP switches only) The following styles of interface configuration are not supported:
  - Service provider style, where a physical interface is divided into multiple logical interfaces, each of which is dedicated to a particular customer VLAN. The `extended-vlan-bridge` encapsulation type is configured on the physical interface.
  - Flexible Ethernet services, which is an encapsulation type that enables a physical interface to support both service provider and enterprise styles of interface configuration.

  For more information about these styles of interface configuration, see Flexible Ethernet Services Encapsulation.

- (QFX5100 switches only) Using the `no-arp-suppression` configuration statement, you can turn off the suppression of ARP requests on one or more specified VLANs. However, starting in Junos OS Release 18.4R3, you must turn off this feature on all VLANs. To do so, you can use one of these configuration options:
  - Use a batch command to turn off the feature on all VLANs—`set groups` *`group-name`* `vlans * no-arp-suppression`. With this command, we use the asterisk (*) as a wildcard that specifies all VLANs.
  - Use a command to turn off the feature on each individual VLAN—`set vlans` *`vlan-name`* `no-arp-suppression`.

  > **NOTE**: The `no-arp-suppression` statement is hidden in the Junos OS CLI, but can still be configured when required.

- QFX5120-48Y, QFX5120-32C, and QFX5200 switches support hierarchical equal-cost multipath (ECMP), which allows these switches to perform a two-level route resolution. However, all other QFX5*xxx* switches do not support hierarchical ECMP. As a result, when an EVPN Type-5 packet is encapsulated with a VXLAN header, then de-encapsulated by a QFX5*xxx* switch that does not support hierarchical ECMP, the switch is unable to resolve the two-levels of routes that were in the inner packet. The switch then drops the packet.

- (QFX5100, QFX5110, QFX5120, QFX5200, and QFX5210 switches) When you configure the IRB interfaces on a device that is concurrently supporting configured VLANs on both a centrally-routed bridging overlay and an edge-routed bridging overlay, the IRB MAC address and virtual gateway MAC address on the IRB interface for the centrally-routed bridging overlay must be different from the IRB MAC address and virtual gateway MAC address on the IRB interface for the edge-routed bridging overlay.

- (QFX5110 and QFX5120 switches only) In an EVPN-VXLAN overlay, we do not support running a routing protocol on an IRB interface that is in a default routing instance (default.inet.0). Instead, we recommend including the IRB interface in a routing instance of type `vrf`.

- The following constraints apply to both VXLANs and VLANs.

  - (QFX5*xxx* switches only) When configuring route leaking between virtual routing and forwarding (VRF) instances, you must specify each prefix with a subnet mask that is equal to or longer than /16 for IPv4 prefixes or equal to or longer than /64 for IPv6 prefixes. If a subnet mask that you specify does not meet these parameters, the specified routes will not be leaked.

  - (QFX5120 switches only) By default, QFX5120 switches allocate 5 MB of a shared buffer to absorb bursts of multicast traffic. If a burst of multicast traffic exceeds 5 MB, the switch will drop the packets that the switch receives after the buffer space is exceeded. To prevent the switch from dropping multicast packets when this situation occurs, you can do one of the following:

    - Use the `shared-buffer` configuration statement in the `[edit class-of-service]` hierarchy level to reallocate a higher percentage of the shared buffer for multicast traffic. For more information about fine-tuning a shared buffer, see Understanding CoS Buffer Configuration.

    - On the Juniper Networks switch from which the bursts of multicast traffic are coming, apply a shaper on the egress link.

  - (QFX5*xxx* switches only) If you have enabled storm control on an interface, you might notice a significant difference between the configured and actual traffic rates for the interface. This difference is the result of an internal storm control meter that quantifies the actual traffic rate for the interface in increments of 64 kbps, for example, 64 kbps, 128 kbps, 192 kbps, and so on.

- (QFX5xxx and EX46xx switches) You can't use hardware-assisted inline Bidirectional Forwarding Detection (BFD) with VXLAN encapsulation of BFD packets. Also, if you configure EVPN overlay BGP peerings, use distributed BFD instead of hardware-assisted inline BFD. See *Understanding How BFD Detects Network Failures* for details on the different types of BFD sessions that you can configure.

- (QFX5130-32CD, QFX5130E-32CD, QFX5130-48C, QFX5700, and QFX5700E switches) Starting in Junos OS Evolved Releases 25.2X100-D10 and 25.4R1, we support hardware-assisted inline BFD with 100 x 3 millisecond timers over VXLAN tunnels on the listed platforms only. This support applies to:

  - Type 2 IPv4 or IPv6 L2 and L3 multihop BFD with ECMP or multihomed VTEPs with these requirements:
    - 100 x 3 ms timers
    - Overlay BGP peering between loopbacks
    - BFD configured on the overlay BGP sessions
  - Type 5 IPv4 or IPv6 multihop BFD with ECMP
  - Pure Type 5 routing instances
- (QFX5xxx and EX46xx switches) We don't support configuring and using MPLS and EVPN-VXLAN at the same time on the same device. We have this constraint because Broadcom-based platforms use the same hardware tables to store tunnel and virtual port information used by both the MPLS and VXLAN feature sets.

  Also, you can't use an MPLS underlay with EVPN and a VXLAN overlay; this is a Broadcom hardware limitation.
- (QFX5xxx and EX46xx switches) We don’t support tagged L3 interfaces and L2 bridge domains as subunits of the same interface with an IRB in service provider style configurations.

VXLAN Constraints on QFX10000 Series Switches

- MC-LAGs are not supported with VXLAN.

  NOTE: In an EVPN-VXLAN environment, EVPN multihoming active-active mode is used instead of MC-LAG for redundant connectivity between hosts and leaf devices.
- IP fragmentation is not supported on the Layer 3 side.
- The following features are not supported on the Layer 2 side:
  - IGMP snooping with EVPN-VXLAN in Junos OS Releases before Junos OS Release 17.2R1.
  - STP (any variant).
- Access port security features are not supported with VXLAN. For example, the following features are not supported:
  - DHCP snooping.
  - Dynamic ARP inspection.
  - MAC limiting and MAC move limiting.

- Ingress node replication is not supported when an SDN controller is used for the control plane (OVSDB-VXLAN). Ingress node replication is supported for EVPN-VXLAN.
- QFX10000 switches that are deployed in an EVPN-VXLAN environment do not support an IPv6 physical underlay network.
- When the next-hop database on a QFX10000 switch includes next hops for both the underlay network and the EVPN-VXLAN overlay network, the next hop to a VXLAN peer cannot be an Ethernet segment identifier (ESI) or a virtual tunnel endpoint (VTEP) interface.
- IRB interfaces used in EVPN-VXLAN overlay networks do not support the IS-IS routing protocol.
- VLAN firewall filters applied to IRB interfaces on which EVPN-VXLAN is enabled.
- Filter-based forwarding (FBF) is not supported on IRB interfaces used in an EVPN-VXLAN environment.
- QFX10002, QFX10008, and QFX10016 switches do not support port mirroring and analyzing when EVPN-VXLAN is also configured.
- When you configure the IRB interfaces on a device that is concurrently supporting configured VLANs on both a centrally-routed bridging overlay and an edge-routed bridging overlay, the IRB MAC address and virtual gateway MAC address on the IRB interface for the centrally-routed bridging overlay must be different from the IRB MAC address and virtual gateway MAC address on the IRB interface for the edge-routed bridging overlay.
- In an EVPN-VXLAN overlay, we do not support running a routing protocol on an IRB interface that is in a default routing instance (default.inet.0). Instead, we recommend including the IRB interface in a routing instance of type `vrf`.
- In an EVPN-VXLAN environment, we don't support configuring anycast gateways with the `default-gateway` statement and the `advertise` statement on links participating in the same Ethernet segment (ES).
- You must configure class of service (CoS) rewrite rules to have the differentiated services code point (DSCP) bits copied from the inner VXLAN header to the outer VXLAN header.

VXLAN Constraints on PTX10000 Series Routers

- You must enable tunnel termination globally on PTX10K series devices in EVPN-VXLAN deployments, as follows:

  ```
  set forwarding-options tunnel-termination
  ```

  This option is disabled by default.

- You can't use a firewall filter on these devices to block VXLAN tunnel termination on specific ports, but you can use the following command to block VXLAN tunnel termination for a port:

  `set interfaces` *`logical-interface-name`* `unit` *`n`* `family` *`inet/inet6`* `no-tunnel-termination`

  Adding the no-tunnel-termination option disables tunnel termination for all traffic on the specific port where it is configured.

VXLAN Constraints on ACX Series Routers

- Multihoming peers within a DC should be from a similar product family. We do not recommend mixing ACX series with QFX series, PTX series, or MX series for multihoming.
- (ACX 7xxx routers) Networks with both MAC and IP scale should configure `set system packet-forwarding-options hw-db-profile cloud-metro`. The default is `lean-edge`, which is intended for IP scale.
- (ACX 7xxx routers) Load balancing is not enabled by default. Shown here is one sample configuration. Configure only those options necessary for your deployment.

```
set forwarding-options hash-key family inet layer-3
set forwarding-options hash-key family inet layer-4
set forwarding-options hash-key family inet6 layer-3
set forwarding-options hash-key family inet6 layer-4
set forwarding-options hash-key family multiservice source-mac
set forwarding-options hash-key family multiservice destination-mac
set policy-options policy-statement <statement name> then load-balance per-packet
```

- (ACX 7xxx routers) Configure `set system packet-forwarding-options system-profile vxlan-extended` to support an EVPN-VXLAN IPv6 underlay.
- (ACX 7xxx routers) Configure `set system packet-forwarding-options system-profile vxlan-stitching` to support EVPN-VLXAN to EVPN-VXLAN stitching, and EVPN-VXLAN to EVPN-MPLS stitching with `no-control-word` support.
- (ACX 7xxx routers) Configure `set system packet-forwarding-options system-profile vxlan-stitching control-word` to support control word functions in a stitched EVPN-VXLAN to EVPN-MPLS network. Refer to *vxlan-stitching* for additional information regarding control word support.
- (ACX 7xxx routers) One user defined Virtual Gateway Address (VGA) IPv4 MAC (`virtual-gateway-v4-mac`), and one user defined VGA IPv6 MAC (`virtual-gateway-v6-mac`) will be supported system wide.
- (ACX 7xxx routers) Configure `set system packet-forwarding-options no-ip-tos-rewrite` to propagate DSCP information from payload to VXLAN router header during VXLAN encapsulation.

- (ACX 7xxx routers) ESI configuration in EVPN multihoming mode is supported per physical interface device, or per LAG interface only.
- (ACX 7xxx routers) IRB interfaces are not supported as EVPN-VXLAN underlay networks.
- (ACX 7xxx routers) For EVPN-VXLAN to EVPN-MPLS stitching, devices which do not support forwarding based on bridge-domain labels will not be compatible with ACX series devices as DC peer GW's.

VXLAN Constraints on All Devices

- If you configure multiple sub-units on a port with an ESI, we don't support the disable operation (`set interfaces` *`logical-interface-name`* `unit` *`n`* `disable`) on those logical interfaces.

RELATED DOCUMENTATION

Understanding VXLANs

Examples: Manually Configuring VXLANs on QFX Series and EX4600 Switches | **101**

Manually Configuring VXLANs on QFX Series and EX4600 Switches | **98**

Understanding the OVSDB Protocol Running on Juniper Networks Devices

The Juniper Networks Junos OS implementation of the Open vSwitch Database (OVSDB) management protocol provides a means through which Juniper Networks devices that support OVSDB can communicate with software-defined networking (SDN) controllers. Juniper Networks devices exchange control and statistical information with the SDN controllers, thereby enabling *virtual machine (VM)* traffic from the entities in a virtualized network to be forwarded to entities in a physical network and vice versa.

The Junos OS implementation of OVSDB includes an OVSDB server and an OVSDB client, both of which run on each Juniper Networks device that supports OVSDB.

The OVSDB server on a Juniper Networks device can communicate with an OVSDB client on an SDN controller. To establish a connection between a Juniper Networks device and an SDN controller, you must specify information about the SDN controller (IP address) and the connection (port over which the connection occurs and the communication protocol to be used) on each Juniper Networks device. After the configuration is successfully committed, the connection is established between the management port of the Juniper Networks device and the SDN controller port that you specify in the Junos OS configuration.

The OVSDB server stores and maintains an OVSDB database schema, which is defined for physical devices. This schema contains control and statistical information provided by the OVSDB client on the Juniper Networks devices and on SDN controllers. This information is stored in various tables in the schema. The OVSDB client monitors the schema for additions, deletions, and modifications to this information, and the information is used for various purposes, such as learning the media access control (MAC) addresses of virtual hosts and physical servers.

The schema provides a means through which the Juniper Networks devices and the SDN controllers can exchange information. For example, the Juniper Networks devices capture MAC routes to entities in the physical network and push this information to a table in the schema so that SDN controllers with connections to these Juniper Networks devices can access the MAC routes. Conversely, SDN controllers capture MAC routes to entities in the virtualized network and push this information to a table in the schema so that Juniper Networks devices with connections to the SDN controllers can access the MAC routes.

Some of the OVSDB table names include the words *local* or *remote*, for example, *unicast MACs local table* and *unicast MACs remote table*. Information in *local* tables is learned by a Juniper Networks device that functions as a hardware *virtual tunnel endpoint (VTEP)*, while information in *remote* tables is learned from other software or hardware VTEPs.

OVSDB Support on Juniper Networks Devices

The following Juniper Networks devices support the *Open vSwitch Database (OVSDB)* management protocol:

- EX9200 Line of Ethernet Switches
- MX80, MX104, MX240, MX480, MX960, MX2010, and MX2020 Universal Routing Platforms
- QFX Series Switches

Starting with Junos OS Release 14.1X53-D30 for QFX5100 switches, 15.1X53-D20 for QFX10002 switches, 15.1X53-D30 for QFX10008 switches, 15.1X53-D60 for QFX10016 switches, 15.1X53-D210 for QFX5110 and QFX5200 switches, 16.1R1 for EX9200 switches and MX routers, and 18.1R1 for QFX5210 switches, the OVSDB software (jsdn) package is included in the Junos OS software (jinstall) package. As a result, if you have one of the listed releases or a later release, you no longer need to Install the separate jsdn package on the Juniper Networks devices.

Change History Table

Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
14.1X53-D30	Starting with Junos OS Release 14.1X53-D30 for QFX5100 switches, 15.1X53-D20 for QFX10002 switches, 15.1X53-D30 for QFX10008 switches, 15.1X53-D60 for QFX10016 switches, 15.1X53-D210 for QFX5110 and QFX5200 switches, 16.1R1 for EX9200 switches and MX routers, and 18.1R1 for QFX5210 switches, the OVSDB software (jsdn) package is included in the Junos OS software (jinstall) package. As a result, if you have one of the listed releases or a later release, you no longer need to Install the separate jsdn package on the Juniper Networks devices.

OVSDB Schema for Physical Devices

An Open vSwitch Database (OVSDB) server runs on a Juniper Networks device that supports the OVSDB management protocol. When this device is connected to one or more SDN controllers, the connections provide a means through which the Juniper Networks device and the SDN controllers can communicate.

Juniper Networks devices that support OVSDB and SDN controllers exchange control and statistical data. This data is stored in the OVSDB database schema defined for physical devices. The schema resides in the OVSDB server. The schema includes several tables. Juniper Networks devices and SDN controllers, both of which have OVSDB clients, can add rows to the tables as well as monitor the tables for the addition, deletion, and modification of rows.

For example, the OVSDB client on a Juniper Networks device and an SDN controller can collect MAC routes learned by entities in the physical or virtualized networks, respectively, and publish the routes to the appropriate table in the schema. By using the MAC routes and other information provided in the table, Juniper Networks devices in the physical network and entities in the virtualized network can determine where to forward *virtual machine (VM)* traffic.

Some of the OVSDB table names include the words *local* or *remote*—for example, the *unicast MACs local table* and the *unicast MACs remote table*. Information in *local* tables is learned by a Juniper Networks device that functions as a hardware virtual tunnel endpoint (VTEP), whereas information in *remote* tables is learned by other software or hardware VTEPs.

Table 2 on page 30 describes the tables in the schema, the physical or virtual entity that is the source of the data provided in the table, and the command that you can enter in the CLI of the Juniper Networks device to get similar information.

Table 2: OVSDB Schema Tables

Table Name	Description	Source of Information	Command
Global table	Includes the top-level configuration for the Juniper Networks device.	Juniper Networks device	–
Manager table	Includes information about each SDN controller that is connected to the Juniper Networks device.	Juniper Networks device	`show ovsdb controller`
Physical switch table	Includes information about a Juniper Networks device that functions as a hardware VTEP. This table includes information only for the device on which the table resides.	Juniper Networks device	–
Physical port table	Includes information about OVSDB-managed interfaces.	Juniper Networks device	`show ovsdb interface`
Logical switch table	Includes the following information: • Logical switches, which you configured in a VMware NSX environment, or virtual networks, which you configured in a Contrail environment. • The equivalent VXLANs, which were configured on the Juniper Networks device.	• SDN controller • Juniper Networks device	`show ovsdb logical-switch`

Table 2: OVSDB Schema Tables *(Continued)*

Table Name	Description	Source of Information	Command
Logical binding statistics table	Includes statistics for OVSDB-managed interfaces.	Juniper Networks device	`show ovsdb statistics interface`
Physical locator table	Includes information about Juniper Networks devices configured as hardware VTEPs, software VTEPs, and service nodes in an NSX environment.	Juniper Networks device	`show ovsdb virtual-tunnel-end-point`
Physical locator set table	Includes a list of software VTEPs, service nodes, or top-of-rack service nodes (TSNs) for a logical switch.	Juniper Networks device	–
Unicast MACs remote table	Reachability information, including unicast MAC addresses, for entities in the virtualized network.	SDN controller	`show ovsdb mac`
Unicast MACs local table	Reachability information, including unicast MAC addresses, for entities in the physical network.	Juniper Networks device	`show ovsdb mac`
Multicast MACs remote table	Includes only one row. In this row, the MAC column includes the keyword `unknown dst` along with a list of software VTEPs, service nodes, or TSNs, which handle multicast traffic.	SDN controller	`show ovsdb mac`

Table 2: OVSDB Schema Tables *(Continued)*

Table Name	Description	Source of Information	Command
Multicast MACs local table	Includes one row for each logical switch. In this row, the MAC column includes the keyword `unknown dst` and a list of hardware VTEPs, which are identified by the IP address assigned to the hardware VTEP loopback interface (lo0). These hardware VTEPs can terminate or originate a VXLAN tunnel. The Multicast MACs local table is introduced in Junos OS Release 14.1X53-D25 for QFX5100 switches and in Junos OS Release 14.2R4 for MX Series routers and EX9200 switches. For all other QFX switches that support OVSDB, this table is present when OVSDB support is introduced.	Juniper Networks device	`show ovsdb mac`

Table 2: OVSDB Schema Tables *(Continued)*

Table Name	Description	Source of Information	Command
Tunnel table	**NOTE**: Only the Juniper Networks switches that support OVSDB with BFD in turn support this table. Includes information about tunnels through which BFD control messages are transmitted between the hardware VTEP and entities that replicate and forward BUM packets (software VTEPs and service nodes) within an OVSDB-managed VXLAN. Using BFD, the hardware VTEP can determine which replicators are reachable.	Juniper Networks device	`show ovsdb tunnels`

Change History Table

Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
14.1X53-D25	The Multicast MACs local table is introduced in Junos OS Release 14.1X53-D25 for QFX5100 switches and in Junos OS Release 14.2R4 for MX Series routers and EX9200 switches. For all other QFX switches that support OVSDB, this table is present when OVSDB support is introduced.

RELATED DOCUMENTATION

Understanding the OVSDB Protocol Running on Juniper Networks Devices

Understanding How to Set Up OVSDB Connections on a Juniper Networks Device

Understanding How Layer 2 BUM and Layer 3 Routed Multicast Traffic Are Handled with OVSDB

The Juniper Networks Junos OS implementation of the Open vSwitch Database (OVSDB) management protocol provides a means through which software-defined networking (SDN) controllers and Juniper Networks devices that support OVSDB can communicate.

This topic explains how a Juniper Networks device with Virtual Extensible LAN (VXLAN) and OVSDB management protocol capabilities handles the following types of traffic:

- (This scenario applies to all Juniper Networks devices that support VXLAN and OVSDB.) Layer 2 *broadcast, unknown unicast, and multicast (BUM)* traffic that originates in an OVSDB-managed VXLAN and is forwarded to interfaces within the same VXLAN.

  > **NOTE**: You must explicitly configure the replication of unknown unicast traffic in a Contrail environment.

- (This scenario applies only to Juniper Networks devices that can function as a Layer 3 VXLAN gateway in an OVSDB-VXLAN environment.) Layer 3 multicast traffic that is received by an *integrated routing and bridging (IRB)* interface in an OVSDB-managed VXLAN and is forwarded to interfaces in another OVSDB-managed VXLAN.

By default, Layer 2 BUM traffic that originates in an OVSDB-managed VXLAN is handled by one or more software *virtual tunnel endpoints (VTEPs)*, service nodes, or top-of-rack service nodes (TSNs) in the same VXLAN. (In this topic, software VTEPs, service nodes, and TSNs are known collectively as *replicators*.) The table for remote multicast media access control (MAC) addresses in the OVSDB schema for physical devices contains only one entry that has the keyword `unknown-dst` as the MAC string and a list of replicators.

Given the previously described table entry, Layer 2 BUM traffic received on an interface in the OVSDB-managed VXLAN is forwarded to one of the replicators. The replicator to which a BUM packet is forwarded is determined by the Juniper Networks device on which the OVSDB-managed VXLAN is configured. On receiving the BUM packet, the entity replicates the packet and forwards the replicas to all interfaces within the VXLAN.

Instead of using replicators, you can optionally enable ingress node replication to handle Layer 2 BUM traffic on Juniper Networks devices that support OVSDB.

> **NOTE**: Ingress node replication is supported on all Juniper Networks devices that support OVSDB except the QFX Series switches.

With ingress node replication enabled, on receiving a Layer 2 BUM packet on an interface in an OVSDB-managed VXLAN, the Juniper Networks device replicates the packet and then forwards the replicas to all software VTEPs included in the unicast MACs remote table in the OVSDB schema. The software VTEPs then forward the replicas to all *virtual machines (VMs)*, except service VMs, or nodes, on the same host.

NOTE: When Juniper Networks devices replicate Layer 2 BUM packets to a large number of remote software VTEPs, the performance of the Juniper Networks devices can be impacted.

On IRB interfaces that forward Layer 3 multicast traffic from one OVSDB-managed VXLAN to another, ingress node replication is automatically implemented. With ingress node replication, the Juniper Networks device replicates a Layer 3 multicast packet and then the IRB interface forwards the replicas to all hardware and software VTEPs, but not to service nodes, in the other OVSDB-managed VXLAN. For the routing of Layer 3 multicast traffic from one OVSDB-managed VXLAN to another, ingress node replication is the only option and does not need to be configured.

RELATED DOCUMENTATION

Configuring OVSDB-Managed VXLANs

Understanding BFD in a VMware NSX Environment with OVSDB and VXLAN | 35

Understanding BFD in a VMware NSX Environment with OVSDB and VXLAN

Within a Virtual Extensible LAN (VXLAN) managed by the Open vSwitch Database (OVSDB) protocol, by default, Layer 2 broadcast, unknown unicast, and multicast (BUM) traffic is replicated and forwarded by one or more software virtual tunnel endpoints (VTEPs) or service nodes in the same VXLAN. (In this topic, software VTEPs and service nodes are known collectively as *replicators*.)

To prevent a Juniper Networks switch that functions as a hardware VTEP in a VMware NSX environment from forwarding BUM packets to a non-functional replicator, starting in Junos OS Release 14.1X53-D40 for QFX5100 switches and 18.1R1 for QFX5110, QFX5200, and QFX5210 switches, these switches can use the Bidirectional Forwarding Detection (BFD) protocol.

By exchanging BFD control messages with replicators at regular intervals, the hardware VTEP can monitor the replicators to ensure that they are functioning and are, therefore, reachable. If the replicator does not respond to three BFD control messages, the BFD status of the replicators is considered to be down. The hardware VTEP does not forward BUM packets to a replicator with a BFD status of down.

To monitor the status of the replicators, the hardware VTEP, NSX controllers, and replicators must all be BFD-capable. In addition, the NSX controller must enable BFD on the hardware VTEP and replicators. When the NSX controller has enabled BFD on the hardware VTEP and replicators, their BFD status is considered to be enabled. If the NSX controller cannot enable BFD on these entities (for example, the entity is not BFD-capable), their BFD status is considered to be disabled.

With the BFD protocol enabled on a hardware VTEP, upon receipt of a BUM packet on an OVSDB-managed interface, the hardware VTEP can choose one of the functioning replicators to handle the packet.

When the hardware VTEP and a replicator exchange BFD control messages, the packets are encapsulated and transported through a VXLAN tunnel. The hardware VTEP and the NSX controller to which it is connected collect Information about this tunnel, such as source and destination IP addresses, the status of the BFD protocol, the status of the tunnel, and so on. The hardware VTEP and NSX controller push their collected information to the tunnel table in the OVSDB schema.

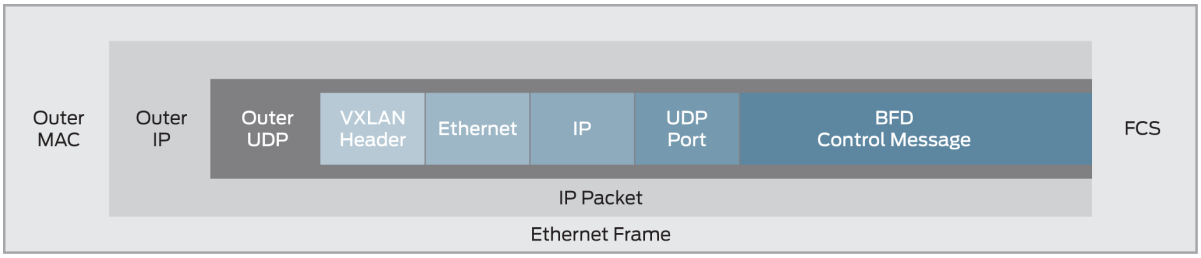
To view the tunnel information collected by the hardware VTEP, you can use the `show ovsdb tunnels` command. This command can help you to determine when there is an issue with the BFD protocol or a replicator.

The VXLAN packet format for BFD control messages is different from the format used for data packets. Moving from the inner part of the packet to the outer, the differences are as follows:

- Addition of UDP port header—UDP port 3784.
- Addition of IP header—Source and destination IP addresses of the devices at the end of each tunnel.
- Addition of Ethernet header—Source and destination MAC addresses of the devices at the end of each tunnel.
- VXLAN—VXLAN network identifier (VNI) of 0.

Figure 4 on page 36 shows the VXLAN packet format for BFD control messages.

Figure 4: VXLAN Packet Format for BFD Control Messages



Change History Table

Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
14.1X53-D40	To prevent a Juniper Networks switch that functions as a hardware VTEP in a VMware NSX environment from forwarding BUM packets to a non-functional replicator, starting in Junos OS Release 14.1X53-D40 for QFX5100 switches and 18.1R1 for QFX5110, QFX5200, and QFX5210 switches, these switches can use the Bidirectional Forwarding Detection (BFD) protocol.

RELATED DOCUMENTATION

Understanding How Layer 2 BUM and Layer 3 Routed Multicast Traffic Are Handled with OVSDB | 34

OVSDB Schema for Physical Devices | 29

show ovsdb tunnels

Understanding Overlay ping and traceroute Packet Support

IN THIS SECTION

In a virtualized overlay network, existing ping and traceroute mechanisms do not provide enough information to determine whether or not connectivity is established throughout the network. The existing `ping` and `traceroute` commands can only verify the basic connectivity between two endpoints in the underlying physical network, but not in the overlay network. For example, you can issue the existing `ping` command on a Juniper Networks device that functions as a virtual tunnel endpoint (VTEP) to another Juniper Networks devices that also functions as a VTEP in a Virtual Extensible LAN (VXLAN) overlay. In this situation, the ping output might indicate that the connection between the source and destination VTEPs is up and running despite the fact that one of the endpoints (physical servers upon which applications directly run) is not reachable.

Starting with Junos OS Release 14.1X53-D30 for QFX5100 switches, Release 16.1 for EX9200 switches, and Release 16.2 for MX Series routers, overlay ping and traceroute are introduced as troubleshooting tools for overlay networks.

For ping and traceroute mechanisms to work in overlay networks, the ping and traceroute packets, also referred to collectively as Operations, Administration, and Management (OAM) packets, must be encapsulated with the same VXLAN UDP headers (outer headers) as the data packets forwarded over the overlay segment. This implementation ensures that transit nodes forward the OAM packets in the same way as a data packet for that particular overlay segment.

If any connectivity issues arise for a particular data flow, the overlay OAM packet corresponding to the flow would experience the same connectivity issues as the data packet for that flow.

When using ping overlay and traceroute overlay, keep the following in mind:

- The only tunnel type supported is VXLAN tunnels.
- The VTEPs in the overlay network that send and receive the overlay ping packets must be Juniper Networks devices that support overlay ping and traceroute.

Overlay ping and traceroute Functionality

Overlay ping and traceroute packets are sent as User Datagram Protocol (UDP) echo requests and replies and are encapsulated in the VXLAN header. VTEPs, which initiate and terminate overlay tunnels, send and receive overlay OAM packets. Overlay ping and traceroute are supported only in VXLAN overlay networks in which the sending and receiving VTEPs are both Juniper Networks devices.

The overlay ping functionality validates both the data plane and the MAC address and IP address of the VTEPs. This additional validation is different from the more commonly known IP ping functionality where the actual destination replies to the echo request without the overlay segment context.

While tracing a route in a VXLAN overlay network, Juniper Networks devices that are along the route that support overlay traceroute additionally provide a timestamp. Third-party devices and Juniper Networks devices that do not support overlay traceroute do not provide this timestamp.

Overlay OAM Packet Format for UDP Payloads

The format of overlay OAM packets depends on the type of payload that is carried in the tunnel. In the case of VXLAN tunnels, the inner packet is a Layer 2 packet.

NOTE: Only Layer 2 UDP payloads are supported.

Figure 5 on page 39 shows complete headers on a VXLAN-encapsulated overlay OAM packet.

Figure 5: VXLAN-Encapsulated Overlay OAM Packet

- Outer Ethernet header—Contains the source MAC (SMAC) and destination MAC (DMAC) addresses of directly connected nodes in the physical network. These addresses change at every hop.
- Outer IP header—Contains the source and destination IP addresses of the Juniper Networks devices that function as the VTEPs that initiate and terminate the tunnel.
- Outer UDP header—Contains the source port associated with the flow entropy and destination port. The source port is an internally calculated hash value. The destination port is the standard UDP port (4789) used for VXLAN.
- VXLAN header—Contains the VXLAN Network Identifier (VNI) or the segment ID of the VXLAN, and new router alert (RA) flag bits.
- Inner Ethernet header—Contains a control MAC address (00-00-5E-90-*xx-xx*) for both the SMAC and DMAC. This address is not forwarded out of the VTEP. Alternatively, the SMAC can be set to a non-control MAC address. However, if a non-control MAC address is used, the VTEP must not learn the SMAC from the overlay OAM packets.
- Inner IP header—Contains the source IP address that can be set to the IP address of the endpoint or source VTEP. The destination IP address can be set to the 127/8 address, which ensures that the overlay OAM packet is not forwarded out of the ports of the Juniper Networks device that is configured as a VTEP.
- Inner UDP header—Contains a new reserved value used in the destination port field in the inner UDP header. This value identifies the incoming UDP packet as an overlay OAM packet.
- Inner UDP payload—Contains all of the overlay OAM-specific message format and type, length, and value (TLV) definitions.

  The Inner UDP payload format is as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Message Type  | Reply mode    | Return Code   | Return Subcode|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Originator Handle                                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Sequence Number                                               |
```

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| TimeStamp Sent (seconds)                                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| TimeStamp Sent (microseconds)                                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| TimeStamp Received (seconds)                                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| TimeStamp Received (microseconds)                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| TLVs ...                                                      |
.                                                               .
.                                                               .
.                                                               .
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The OAM-specific message type is one of the following:

```
Value What it means
----- -------------
1     Echo Request
2     Echo Reply

Reply Mode Values:-
Value What it means
----- --------------------------------
1     Do not reply
2     Reply via an IPv4/IPv6 UDP Packet
3     Reply via Overlay Segment
```

The TLV definition for VXLAN ping is as follows:

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Type = 1(VXLAN ping IPv4)   | Length                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| VXLAN VNI                                     | Reserved      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| IPv4 Sender Address                                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Multiple Routing Instance Support

Starting in Junos OS Release 19.3R1, you can use the `ping overlay` and `traceroute overlay` commands to verify connectivity and detect fault in a static VxLAN tunnel with multiple routing instances. The ping and traceroute packets created for the `ping overlay` and `traceroute overlay` commands follow the same underlay network path as the data packets. This allow you to verify the connectivity between two VTEPs in the overlay VxLAN tunnel. The devices that are configured as the source and destination VTEP must both be running a Junos OS release that supports multiple routing instance, but the transit devices do not.

Change History Table

Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
19.3R1	Starting in Junos OS Release 19.3R1, you can use the `ping overlay` and `traceroute overlay` commands to verify connectivity and detect fault in a static VxLAN tunnel with multiple routing instances.
14.1X53-D30	Starting with Junos OS Release 14.1X53-D30 for QFX5100 switches, Release 16.1 for EX9200 switches, and Release 16.2 for MX Series routers, overlay ping and traceroute are introduced as troubleshooting tools for overlay networks.

RELATED DOCUMENTATION

Example: Troubleshoot a VXLAN Overlay Network with Overlay Ping and Overlay Traceroute on QFX Series Switches | **126**

ping overlay

traceroute overlay

PIM NSR and Unified ISSU Support for VXLAN Overview

Protocol Independent Multicast (PIM) nonstop active routing (NSR) support for Virtual Extensible LAN (VXLAN) is supported on MX Series routers.

The Layer 2 address learning daemon (l2ald) passes VXLAN parameters (VXLAN multicast group addresses and the source interface for a VXLAN tunnel [`vtep-source-interface`]) to the routing protocol process on the primary Routing Engine. The routing protocol process forms PIM joins with the multicast routes through the pseudo-VXLAN interface based on these configuration details.

Because the l2ald daemon does not run on the backup Routing Engine, the configured parameters are not available to the routing protocol process in the backup Routing Engine when NSR is enabled. The PIM NSR mirroring mechanism provides the VXLAN configuration details to the backup Routing Engine, which enables creation of the required states. The routing protocol process matches the multicast routes on the backup Routing Engine with PIM states, which maintains the multicast routes in the Forwarding state.

In response to Routing Engine switchover, the multicast routes remain in the Forwarding state on the new primary Routing Engine. This prevents traffic loss during Routing Engine switchover. When the l2ald process becomes active, it refreshes VXLAN configuration parameters to PIM.

NOTE: For this feature, NSR support is available for VXLAN in PIM sparse mode.

This feature does not introduce any new CLI commands. You can issue the following `show` commands on the backup Routing Engine to monitor the PIM joins and multicast routes on the backup Routing Engine:

- `show pim join extensive`
- `show multicast route extensive`

Unified ISSU Support

Starting in Junos OS Release 17.2 R1, unified in-service software upgrade is supported for VXLAN using PIM on MX Series routers. ISSU enables you to upgrade your Junos OS software on your MX Series router with no disruption on the control plane and with minimal disruption of traffic. Unified ISSU is supported only on dual Routing Engine platforms. The graceful Routing Engine switchover (GRES) and nonstop active routing (NSR) features must both be enabled. Unified ISSU allows you to eliminate network downtime, reduce operating costs, and deliver higher levels of services. See Getting Started with Unified In-Service Software Upgrade.

NOTE: Unified ISSU is not supported on the QFX series switches.

To enable GRES, include the `graceful-switchover` statement at the `[edit chassis redundancy]` hierarchy level.

To enable NSR, include the `nonstop-routing` statement at the `[edit routing-options]` hierarchy level and the `commit synchronize` statement at the `[edit system]` hierarchy level.

Change History Table

Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
17.2R1	Starting in Junos OS Release 17.2 R1, unified in-service software upgrade is supported for VXLAN using PIM on MX Series routers.
16.2R1	Starting in Junos OS Release 16.2R1, Protocol Independent Multicast (PIM) nonstop active routing (NSR) support for Virtual Extensible LAN (VXLAN) is supported on MX Series routers.

RELATED DOCUMENTATION

- *show pim join*
- *show multicast route*
- Understanding Graceful Routing Engine Switchover

2

PART

Configuring OVSDB and VXLAN

CHAPTER 2

Configuring OVSDB-Managed VXLANs with an SDN Controller

IN THIS CHAPTER

OVSDB and VXLAN Configuration Workflows for VMware NSX Environment

IN THIS SECTION

The workflow that you use to configure Open vSwitch Database (OVSDB) and Virtual Extensible LAN (VXLAN) in a VMware NSX environment depends on the Juniper Networks device that you are configuring. This topic provides more information about the following workflows:

OVSDB and VXLAN Configuration Workflow for QFX Series Switches

Table 3 on page 46 provides a high-level workflow of the tasks that you must perform to configure OVSDB and VXLAN on QFX Series switches. You must perform the tasks in Table 3 on page 46 for each Juniper Networks switch that you plan to deploy in an OVSDB environment. In general, the successful completion of a task in this workflow depends on the successful completion of the previous task, so it is important to adhere to the task sequence provided in Table 3 on page 46.

Table 3: OVSDB and VXLAN Configuration Workflow for QFX Series Switches

Sequence	Task	For More Information
1	Create and install a Secure Sockets Layer (SSL) key and certificate.	*Creating and Installing an SSL Key and Certificate on a Juniper Networks Device for Connection with SDN Controllers*.
2	Enter the `set switch-options ovsdb-managed` configuration mode command on the Juniper Networks switch.	–
3	Explicitly configure a connection to at least one VMware NSX controller.	"Setting Up OVSDB on Juniper Networks Devices That Support the Dynamic Configuration of VXLANs" on page 53.
4	Specify that each physical interface associated with a VXLAN is to be managed by OVSDB.	"Setting Up OVSDB on Juniper Networks Devices That Support the Dynamic Configuration of VXLANs" on page 53.
5	Configure a logical switch for each OVSDB-managed VXLAN that you plan to implement.	See the VMware documentation that accompanies NSX Manager or the NSX API.

Table 3: OVSDB and VXLAN Configuration Workflow for QFX Series Switches *(Continued)*

Sequence	Task	For More Information
6	• For each Juniper Network switch on which OVSDB-managed VXLANs and interfaces are configured, create a gateway. • For each OVSDB-managed interface that you configure, create a gateway service. • For each logical interface that you plan to implement for a VXLAN, configure a logical switch port. **NOTE**: On QFX Series switches, when multiple logical interfaces are bound to an OVSDB-managed physical interface, keep in mind that all of the logical interfaces must be either access interfaces that handle untagged packets or trunk interfaces that handle tagged packets. An OVSDB-managed physical interface does not support a mix of access and trunk interfaces.	For general information about configuring gateways, gateway services, and logical switch ports, see the VMware documentation that accompanies NSX Manager or the NSX API. For key NSX Manager configuration details that help you configure gateways, gateway services, and logical switch ports so they function properly with their physical counterparts, see *VMware NSX Configuration for Juniper Networks Devices Functioning as Virtual Tunnel Endpoints*.
7	Configure the loopback interface (lo0) on the Juniper Networks switch for VXLAN by entering the following configuration mode commands: • `set interfaces lo0 unit 0 family inet address` *`ip-address`* `primary` • `set switch-options vtep-source-Interface lo0.0`	–

After you successfully complete task 6 in Table 3 on page 46, the Juniper Networks switch dynamically creates a VXLAN for each logical switch that you configured in task 5. The Juniper Networks switch also dynamically creates and associates interfaces with each VXLAN. The dynamically created interface configuration is based on the gateway service and logical switch ports that you configured in task 6. For more information, see "Understanding Dynamically Configured VXLANs in an OVSDB Environment" on page 55.

For OVSDB-VXLAN scenarios in which Juniper Networks switches are commonly deployed, see the following topics:

- "Example: Setting Up a VXLAN Layer 2 Gateway and OVSDB Connections in a VMware NSX Environment (Trunk Interfaces Supporting Untagged Packets)" on page 70
- "Example: Setting Up a VXLAN Layer 2 Gateway and OVSDB Connections in a VMware NSX Environment (Trunk Interfaces Supporting Tagged Packets)" on page 81

OVSDB and VXLAN Configuration Workflow for MX Series Routers and EX9200 Switches

Table 4 on page 48 provides a high-level workflow of the tasks that you must perform to configure OVSDB and VXLAN on MX Series routers and EX9200 switches. You must perform the tasks inTable 4 on page 48 for each Juniper Networks device that you plan to deploy in an OVSDB environment. In general, the successful completion of a task in this workflow depends on the successful completion of the previous task, so it is important to adhere to the task sequence provided in Table 4 on page 48.

Table 4: OVSDB and VXLAN Configuration Workflow for MX Series Routers and EX9200 Switches

Sequence	Task	For More Information
1	Create and install an SSL key and certificate.	*Creating and Installing an SSL Key and Certificate on a Juniper Networks Device for Connection with SDN Controllers*.
2	Explicitly configure a connection to at least one NSX controller.	*Setting Up the OVSDB Protocol on Juniper Networks Devices that Support Manual Configuration of VXLANs*.
3	Specify that each physical interface associated with a VXLAN is to be managed by OVSDB.	*Setting Up the OVSDB Protocol on Juniper Networks Devices that Support Manual Configuration of VXLANs*.
4	Configure a logical switch for each OVSDB-managed VXLAN that you plan to implement.	See the VMware documentation that accompanies NSX Manager or the NSX API.
5	Configure OVSDB-managed VXLANs.	*Configuring OVSDB-Managed VXLANs*.

Table 4: OVSDB and VXLAN Configuration Workflow for MX Series Routers and EX9200 Switches *(Continued)*

Sequence	Task	For More Information
6	For each Juniper Network device on which OVSDB-managed VXLANs and interfaces will be configured, create a gateway. For each OVSDB-managed interface that you configure, create a gateway service. For each logical interface that you plan to implement for a VXLAN, configure a logical switch port.	For general information about configuring gateways, gateway services, and logical switch ports, see the VMware documentation that accompanies NSX Manager or the NSX API. For key NSX Manager configuration details that help you configure gateways, gateway services, and logical switch ports, so that they function properly with their physical counterparts, see *VMware NSX Configuration for Juniper Networks Devices Functioning as Virtual Tunnel Endpoints*.
7	Configure the loopback interface (lo0) on the Juniper Networks device for VXLAN by entering the following configuration mode commands: • `set interfaces lo0 unit 0 family inet address` *`ip-address`* `primary` • `set switch-options vtep-source-Interface lo0.0`	–

For OVSDB-VXLAN scenarios in which these Juniper Networks devices are commonly deployed, see the following topics:

- *Example: Setting Up Inter-VXLAN Unicast Routing and OVSDB Connections in a Data Center*
- *Example: Setting Up Inter-VXLAN Unicast and Multicast Routing and OVSDB Connections in a Data Center*
- *Example: Configuring VXLAN to VPLS Stitching with OVSDB*

RELATED DOCUMENTATION

> *Verifying That a Logical Switch and Corresponding Junos OS OVSDB-Managed VXLAN Are Working Properly*

Installing OVSDB on Juniper Networks Devices

NOTE: The Open vSwitch Database (OVSDB) software is included in the **jsdn** package. For some Juniper Networks devices, the **jsdn** package is included in the Junos OS software (**jinstall**) package. On these Juniper Networks devices, you do not need to install the separate **jsdn** package, which means that you can skip the task described in this topic. For information about which devices do not require installation of the separate **jsdn** package, see *OVSDB Support on Juniper Networks Devices*.

If the **jsdn** package for your Juniper Networks device is not included in the **jinstall** package, you must copy a separate **jsdn** package to the Juniper Networks device and then install the package. The package name uses the following format:

jsdn-*packageID-release*

where:

- *packageID* identifies the package that must run on each Juniper Networks device.
- *release* identifies the release; for example, 16.2. The **jsdn** package release and the **jinstall** release running on the device must be the same.

To install the **jsdn** package on a Juniper Networks device:

1. Download the software package to the Juniper Networks device.
2. If an older **jsdn** package already exists on the Juniper Networks device, remove the package by issuing the `request system software delete` operational mode command.

```
user@device> request system software delete existing-ovsdb-package
```

3. Install the new **jsdn** package by using the `request system software add` operational mode command.

```
user@device> request system software add path-to-ovsdb-package
```

RELATED DOCUMENTATION

Understanding the OVSDB Protocol Running on Juniper Networks Devices

OVSDB and VXLAN Configuration Workflows for VMware NSX Environment

Understanding How to Set Up OVSDB Connections on a Juniper Networks Device

The Juniper Networks Junos OS implementation of the Open vSwitch Database (OVSDB) management protocol provides a means through which Juniper Networks devices that support OVSDB can communicate with software-defined networking (SDN) controllers. A Juniper Networks device exchanges control and statistical data with each SDN controller to which it is connected.

You can connect a Juniper Networks device to more than one SDN controller for redundancy.

In a VMware NSX environment, one cluster of NSX controllers typically includes three or five controllers. To implement the OVSDB management protocol on a Juniper Networks device, you must explicitly configure a connection to one SDN controller, using the Junos OS CLI. If the SDN controller to which you explicitly configure a connection is in a cluster, the controller pushes information about other controllers in the same cluster to the device, and the device establishes connections with the other controllers. However, you can also explicitly configure connections with the other controllers in the cluster, using the Junos OS CLI.

To implement the OVSDB management protocol on a Juniper Networks device in a Contrail environment, you must configure a connection to a Contrail controller, using the Junos OS CLI.

Connections to all SDN controllers are made on the management interface of the Juniper Networks device. To set up a connection between a Juniper Networks device and an SDN controller, you need to configure the following parameters on the Juniper Networks device:

- IP address of the SDN controller.
- The protocol that secures the connection. *Secure Sockets Layer (SSL)* is the supported protocol.

  NOTE: The SSL connection requires a private key and certificates, which must be stored in the **/var/db/certs** directory of the Juniper Networks device. See *Creating and Installing an SSL Key and Certificate on a Juniper Networks Device for Connection with SDN Controllers*.

- Number of the port over which the connection is made. The port number of the default port is 6632.

Optionally, you can configure the following connection timers on the Juniper Networks device:

- Inactivity probe duration—The maximum amount of time, in milliseconds, that the connection can be inactive before an inactivity probe is sent. The default value is 0 milliseconds, which means that an inactivity probe is never sent.

- Maximum backoff duration—If an attempt to connect to an SDN controller fails, the maximum amount of time, in milliseconds, before the device can make the next attempt. The default value is 1000 milliseconds.

RELATED DOCUMENTATION

Setting Up the OVSDB Protocol on Juniper Networks Devices that Support Manual Configuration of VXLANs

Setting Up OVSDB on Juniper Networks Devices That Support the Dynamic Configuration of VXLANs | 53

Creating and Installing an SSL Key and Certificate on a Juniper Networks Device for Connection with SDN Controllers

To secure a connection between a Juniper Networks device that supports the Open vSwitch Database (OVSDB) management protocol and one or more software-defined networking (SDN) controllers, the following Secure Sockets Layer (SSL) files must be present in the **/var/db/certs** directory on the device:

- **vtep-privkey.pem**
- **vtep-cert.pem**
- **ca-cert.pem**

You must create the **vtep-privkey.pem** and **vtep-cert.pem** files for the device and then install the two files in the **/var/db/certs** directory on the device.

Upon initial connection between a Juniper Networks device with OVSDB implemented and an SDN controller, the **ca-cert.pem** file is automatically generated and then installed in the **/var/db/certs** directory on the device.

NOTE: The situation at your particular site determines the possible methods that you can use to create the **vtep-privkey.pem** and **vtep-cert.pem** files and install them in the Juniper Networks device. Instead of providing procedures for all possible situations, this topic provides a procedure for one common scenario.

The procedure provided in this topic uses the OpenFlow public key infrastructure (PKI) management utility ovs-pki on a Linux computer to initialize a PKI and create the **vtep-privkey.pem** and **vtep-cert.pem** files. (If you have an existing PKI on your Linux computer, you can skip the step to initialize a

new one.) By default, the utility initializes the PKI and places these files in the **/usr/local/share/openvswitch/pki** directory of the Linux computer.

To create and install an SSL key and certificate on a Juniper Networks device:

1. Initialize a PKI if one does not already exist on your Linux computer.

```
# ovs-pki init
```

2. On the same Linux computer on which the PKI exists, create a new key and certificate for the Juniper Networks device.

```
# ovs-pki req+sign vtep
```

3. Copy only the **vtep-privkey.pem** and **vtep-cert.pem** files from the Linux computer to the **/var/db/certs** directory on the Juniper Networks device.

RELATED DOCUMENTATION

> *Understanding How to Set Up OVSDB Connections on a Juniper Networks Device*

Setting Up OVSDB on Juniper Networks Devices That Support the Dynamic Configuration of VXLANs

To implement the Open vSwitch Database (OVSDB) management protocol on a Juniper Networks device, you must configure a connection between the Juniper Networks device and a software-defined networking (SDN) controller using the Junos OS CLI.

All SDN controller connections are made on the management interface of the Juniper Networks device. This connection is secured by using the Secure Sockets Layer (SSL) protocol. The default port number for the connection is 6632.

You must also specify that each physical interface that is connected to a physical server is managed by OVSDB. By performing this configuration, you essentially disable the Juniper Networks device from learning about other Juniper Networks devices that function as hardware virtual tunnel endpoints (VTEPs) and the MAC addresses learned by the hardware VTEPs. Instead, this configuration enables OVSDB to learn about these elements.

Before setting up OVSDB on a Juniper Networks device, you must do the following:

- Create an SSL private key and certificate, if they do not already exist, and install them in the **/var/db/certs** directory of the Juniper Networks device. See *Creating and Installing an SSL Key and Certificate on a Juniper Networks Device for Connection with SDN Controllers*.

To set up OVSDB on a Juniper Networks device:

1. Specify the IP address of the SDN controller.

```
[edit protocols ovsdb]
user@host# set controller  ip-address
```

2. Specify SSL as the protocol that secures the connection between the Juniper Networks device and the SDN controller.

```
[edit protocols ovsdb]
user@host# set controller  ip-address protocol  ssl
```

3. Set the number of the port over which the connection to the SDN controller is made.

```
[edit protocols ovsdb]
user@host# set controller  ip-address protocol  ssl port number
```

4. (Optional) Specify (in milliseconds) how long the connection can be inactive before an inactivity probe is sent.

```
[edit protocols ovsdb]
user@host# set controller  ip-address inactivity-probe-duration milliseconds
```

5. (Optional) Specify (in milliseconds) how long the device must wait before it can try to connect to the SDN controller again if the previous attempt failed.

```
[edit protocols ovsdb]
user@host# set controller  ip-address maximum-backoff-duration milliseconds
```

6. (Optional) Repeat Steps 1 through 5 to configure a connection to an additional SDN controller in the NSX environment.

7. Specify that each physical interface that is connected to a physical server is managed by OVSDB.

```
[edit protocols ovsdb]
user@host# set interfaces interface-name
```

   When specifying the *interface-name*, you do not need to include a logical unit number.

8. Complete the remaining configuration tasks, which are described in *OVSDB and VXLAN Configuration Workflows for VMware NSX Environment*).

Understanding Dynamically Configured VXLANs in an OVSDB Environment

IN THIS SECTION

NOTE: This topic applies only to QFX Series switches, which support the dynamic configuration of Open vSwitch Database (OVSDB)-managed Virtual Extensible LANs (VXLANs). Although the configuration of OVSDB-managed VXLANs is automated on these switches, there are tasks that you must perform before and after the dynamic configuration.

On all other Juniper Networks devices that support OVSDB and VXLAN, you must manually configure OVSDB-managed VXLANs using the Junos OS CLI. For more information about manually configuring OVSDB-managed VXLANs, see *Configuring OVSDB-Managed VXLANs*.

The Juniper Networks Junos OS implementation of the OVSDB management protocol provides a means through which Juniper Networks devices that support OVSDB can communicate with software-defined networking (SDN) controllers. Support for OVSDB enables the devices in a physical network to be integrated into a virtualized network.

In a Junos OS environment, the concept of an OVSDB-managed Layer 2 broadcast domain in which data flows are limited to that domain is known as a *VXLAN*. The term used for the same concept in other OVSDB environments depends on the environment:

- In an NSX environment, the same concept is known as a *logical switch*.
- In a Contrail environment, the same concept is known as a *virtual network*.

Understanding the terminology used in the different environments will help you to better understand the workflow associated with the dynamic configuration of OVSDB-managed VXLANs, including tasks that you must perform before and after the dynamic configuration.

The following sections describe the dynamic configuration of OVSDB-managed VXLANs:

Performing Tasks Before and After the Dynamic Configuration of OVSDB-Managed VXLANs

Although the configuration of OVSDB-managed VXLANs is automated, there are some tasks that you must perform before and after the dynamic configuration. Table 5 on page 56 includes a sequentially ordered workflow of tasks and events for the dynamic configuration of OVSDB-managed VXLANs in an NSX environment, while Table 6 on page 59 includes the equivalent information for a Contrail environment. Your familiarity with these workflows will ensure that the dynamic configuration of OVSDB-managed VXLANs is properly implemented.

In Table 5 on page 56, the NSX controller and Juniper Networks switch handle the events described in workflow numbers 4, 6, and 7. You must perform the tasks described in workflow numbers 1, 2, 3, 5, and 8. If you perform a task in a different order than that outlined in Table 5 on page 56, the dynamic configuration might not work or the dynamically configured OVSDB-managed VXLAN might not become functional.

Table 5: Workflow of Tasks and Events for the Dynamic Configuration of OVSDB-Managed VXLANs in an NSX Environment

Workflow Number	Task or Event	How Task or Event Is Handled	More Information About Task or Event
1	Enable the Juniper Networks switch to dynamically configure an OVSDB-managed VXLAN.	You must manually enable this capability by entering the `set switch-options ovsdb-managed` configuration mode command on the switch.	–

Table 5: Workflow of Tasks and Events for the Dynamic Configuration of OVSDB-Managed VXLANs in an NSX Environment *(Continued)*

Workflow Number	Task or Event	How Task or Event Is Handled	More Information About Task or Event
2	On the Juniper Networks switch, configure each physical interface that is connected to a physical server so that the interface is managed by OVSDB.	For each physical interface, you must manually enter the `set protocols ovsdb interfaces` *`interface-name`* configuration mode command.	When entering the interface name, you do not need to include a logical unit number.
3	For each OVSDB-managed VXLAN that you want to implement, configure a logical switch.	You must manually configure the logical switch by using NSX Manager or the NSX API. See the documentation that accompanies NSX Manager or the NSX API.	A universally unique identifier (UUID) for the logical switch is dynamically generated.
4	Relevant information about the logical switch is pushed to the Juniper Networks switch.	The NSX controller pushes relevant information to the logical switch table in the OVSDB schema for physical devices. This schema resides in the Juniper Networks switch.	–

Table 5: Workflow of Tasks and Events for the Dynamic Configuration of OVSDB-Managed VXLANs in an NSX Environment *(Continued)*

Workflow Number	Task or Event	How Task or Event Is Handled	More Information About Task or Event
5	Create the following entities: • For each Juniper Networks switch that you deploy as a hardware VTEP, you create a gateway. • For each OVSDB-managed interface that you configured in workflow number 2, you create a gateway service. • For each interface that you plan to implement for a VXLAN, configure a logical switch port.	You must manually configure these entities by using NSX Manager or the NSX API. See the documentation that accompanies NSX Manager or the NSX API. Also see *VMware NSX Configuration for Juniper Networks Devices Functioning as Virtual Tunnel Endpoints*.	–
6	Relevant information about the gateway service and logical switch port are pushed to the Juniper Networks switch.	The NSX controller pushes this information to the Juniper Networks switch.	–
7	A corresponding VXLAN is dynamically created. Based on the gateway service and logical switch port configured in NSX Manager or the NSX API, one or more interfaces are also created and associated with the VXLAN.	The Juniper Networks switch dynamically creates the VXLAN and interface configuration.	For the name of the VXLAN, the Juniper Networks switch uses the UUID of the logical switch.

Table 5: Workflow of Tasks and Events for the Dynamic Configuration of OVSDB-Managed VXLANs in an NSX Environment *(Continued)*

Workflow Number	Task or Event	How Task or Event Is Handled	More Information About Task or Event
8	(Recommended) Verify that the logical switch, corresponding VXLAN, and associated interfaces are configured properly and are operational.	You can enter the `show ovsdb logical-switch` operational mode command on the Juniper Networks switch. In the output, check the Flags field for the logical switches that you configured as described in workflow number 3 to ensure that it displays Created by both.	If the output of the `show ovsdb logical-switch` operational mode command does not include the Created by both state, see *Troubleshooting a Nonoperational Logical Switch and Corresponding Junos OS OVSDB-Managed VXLAN*.

In Table 6 on page 59, the Contrail controller and Juniper Networks switch handle the events described in workflow numbers 5, 8, and 9. You must perform all other tasks described in the table. If you perform a task in a different order than that outlined in Table 6 on page 59, the dynamic configuration might not work or the dynamically configured OVSDB-managed VXLAN might not become functional.

NOTE: Although you can perform the Contrail configurations outlined in Table 6 on page 59 in the Contrail Web user interface or in the Contrail REST API, Table 6 on page 59 only describes how to perform tasks in the Contrail Web user interface.

Table 6: Workflow of Tasks and Events for the Dynamic Configuration of OVSDB-Managed VXLANs in a Contrail Environment

Workflow Number	Task or Event	How Task or Event Is Handled	More Information About Task or Event
1	On the Juniper Networks switch, configure a unique hostname for the switch.	You must manually enter the `set system host-name` *`host-name`* configuration mode command on the switch.	If implementing a virtual chassis, be aware that all members of the virtual chassis must have the same hostname.

Table 6: Workflow of Tasks and Events for the Dynamic Configuration of OVSDB-Managed VXLANs in a Contrail Environment *(Continued)*

Workflow Number	Task or Event	How Task or Event Is Handled	More Information About Task or Event
2	Enable the Juniper Networks switch to dynamically configure an OVSDB-managed VXLAN.	You must manually enable this capability by entering the `set switch-options ovsdb-managed` configuration mode command on the switch.	–
3	On the Juniper Networks switch, configure each physical interface that is connected to a physical server so that the interface is managed by OVSDB.	For each physical interface, you must manually enter the `set protocols ovsdb interfaces` *`interface-name`* configuration mode command.	When entering the interface name, you do not need to include a logical unit number.
4	For each OVSDB-managed VXLAN that you want to implement, configure a virtual network in the Contrail Web user interface.	You must manually configure the virtual network by navigating to Configure > Networking > Networks. See Creating a Virtual Network.	See *Contrail Configuration for Juniper Networks Devices That Function as Hardware VTEPs*.
5	Relevant information about the virtual network is pushed to the Juniper Networks switch.	The Contrail controller pushes relevant information to the logical switch table in the OVSDB schema for physical devices. This schema resides in the Juniper Networks switch.	–

Table 6: Workflow of Tasks and Events for the Dynamic Configuration of OVSDB-Managed VXLANs in a Contrail Environment *(Continued)*

Workflow Number	Task or Event	How Task or Event Is Handled	More Information About Task or Event
6	For each interface that you plan to implement for a VXLAN, configure a logical interface.	In the Contrail Web user interface, you must manually configure the logical interface by navigating to Configure > Physical Devices > Interfaces. For information about configuring a logical interface, see Using TOR Switches and OVSDB to Extend the Contrail Cluster to Other Instances.	See *Contrail Configuration for Juniper Networks Devices That Function as Hardware VTEPs*.
7	For each Juniper Networks switch that you deploy as a hardware VTEP, you create a physical router.	In the Contrail Web user interface, you must manually configure the physical router by navigating to Configure > Physical Devices > Physical Routers. For information about configuring a physical router, see Using TOR Switches and OVSDB to Extend the Contrail Cluster to Other Instances.	See *Contrail Configuration for Juniper Networks Devices That Function as Hardware VTEPs*.
8	Relevant information about the logical interfaces is pushed to the Juniper Networks switch.	The Contrail controller pushes this information to the Juniper Networks switch.	–

Table 6: Workflow of Tasks and Events for the Dynamic Configuration of OVSDB-Managed VXLANs in a Contrail Environment *(Continued)*

Workflow Number	Task or Event	How Task or Event Is Handled	More Information About Task or Event
9	A corresponding VXLAN is dynamically created. Based on the logical interface configured in the Contrail Web user interface, one or more interfaces are also created and associated with the VXLAN.	The Juniper Networks switch dynamically creates the VXLAN and interface configurations.	For the name of the VXLAN, the Juniper Networks switch uses the prefix “Contrail-” and the UUID of the virtual network.
10	(Recommended) Verify that the virtual network, corresponding VXLAN, and interfaces are configured properly and are operational.	You can enter the `show ovsdb logical-switch` operational mode command on the Juniper Networks switch. In the output, check the Flags field for the virtual network that you configured as described in workflow number 4 to ensure that it displays Created by both.	If the output of the `show ovsdb logical-switch` operational mode command does not include the Created by both state, see *Troubleshooting a Nonoperational Logical Switch and Corresponding Junos OS OVSDB-Managed VXLAN*.

What the Juniper Networks Switch Actually Creates Dynamically

When a Juniper Networks switch creates a VXLAN, it sets up a configuration similar to the following sample:

```
set vlans 28805c1d-0122-495d-85df-19abd647d772 vxlan vni 100
```

Note the following meanings for this sample configuration:

- The name of the VXLAN is 28805c1d-0122-495d-85df-19abd647d772. The UUID of the logical switch, which was configured in NSX Manager or in the NSX API, is 28805c1d-0122-495d-85df-19abd647d772. For a VXLAN created in a Contrail environment, the name would be preceded by “Contrail-”.

- For the *virtual network identifier (VNI)*, the Juniper Networks switch uses either the VNI specified in the logical switch configuration (NSX) or the VXLAN identifier specified in the virtual network configuration (Contrail). In this example, VNI 100 is used. If the Juniper Networks switch detects that VNI 100 is a duplicate of a VNI from a VXLAN configured by manually using the `set vlans vlan-name vxlan vni (1 - 16777214)` command in the Junos OS CLI, the switch deletes the manually configured VXLAN. Or, if the Juniper Networks switch detects that VNI 100 is specified in the dynamically configured VXLAN, but for some reason, the VNI is no longer in the equivalent logical switch or virtual network configuration, the Juniper Networks switch deletes VNI 100 from the VXLAN.

If you need to modify or delete an OVSDB-managed VXLAN that was dynamically configured by the Juniper Networks switch, you must modify or delete either the corresponding logical switch configuration (NSX), or the corresponding virtual network configuration (Contrail). After you modify or delete the configuration, the SDN controller pushes the update to the Juniper Networks switch, and the switch modifies or deletes its configuration accordingly.

Depending on either the gateway service and logical switch ports configuration (NSX), or the logical interface configuration (Contrail), the Juniper Networks switch dynamically creates and associates one or more interfaces with the VXLAN. The configuration generated by the switch depends on whether an interface must support untagged or tagged packets. The following sections provide information about the configuration that the switch dynamically generates for each interface:

- "Dynamic Association of a Trunk Interface Supporting Untagged Packets to a Dynamically Created VXLAN" on page 63
- "Dynamic Association of a Trunk Interface Supporting Tagged Packets to a Dynamically Created VXLAN" on page 64

Dynamic Association of a Trunk Interface Supporting Untagged Packets to a Dynamically Created VXLAN

To determine the type of interface to create and associate with an OVSDB-managed VXLAN, the Juniper Networks switch uses the VLAN ID that you specified when configuring either the logical switch port (NSX), or the logical interface (Contrail). If you specified **0** as the VLAN ID, the switch dynamically configures a trunk interface that can handle untagged packets. (If you specified a valid non-zero VLAN ID, the switch creates a trunk interface that handles tagged packets.)

After the SDN controller pushes either the NSX or Contrail configurations to the Juniper Networks switch, the switch dynamically creates a configuration similar to the following:

```
set interfaces ge-1/0/0 flexible-vlan-tagging
set interfaces ge-1/0/0 native-vlan-id 4094
set interfaces ge-1/0/0 encapsulation extended-vlan-bridge
```

```
set interfaces ge-1/0/0 unit 0 vlan-id 4094
set vlans 28805c1d-0122-495d-85df-19abd647d772 interface ge-1/0/0.0
```

This sample configuration sets up physical interface ge-1/0/0 as a trunk interface. It also configures a native VLAN with an ID of 4094 and specifies that logical interface ge-1/0/0.0 is a member of the native VLAN. As a result, logical interface ge-1/0/0.0 handles incoming untagged packets.

NOTE: We reserve VLAN ID 4094 for native VLANs in an OVSDB environment. As a result, when you create either a logical switch port (NSX) or a logical interface (Contrail), if you specify VLAN ID 4094, the Juniper Networks switch does not dynamically configure a corresponding interface. Also, a system log error message is generated.

Instead of dynamically configuring physical interface ge-1/0/0 as an access interface, which typically handles untagged packets, the Juniper Networks switch configures it as a trunk interface. The intent of this configuration is to support the division of physical interface ge-1/0/0 into multiple logical interfaces, some of which are associated with VXLANs that handle untagged packets and some of which are associated with VXLANs that handle tagged packets.

The sample configuration also creates logical interface ge-1/0/0.0 and associates this interface with VXLAN 28805c1d-0122-495d-85df-19abd647d772.

Dynamic Association of a Trunk Interface Supporting Tagged Packets to a Dynamically Created VXLAN

Starting with Junos OS Release 14.1X53-D15 for QFX5100 switches, 15.1X53-D10 for QFX10002 switches, 15.1X53-D30 for QFX10008 switches, 15.1X53-D60 for QFX10016 switches, 15.1X53-D210 for QFX5110 and QFX5200 switches, and 18.1R1 for QFX5210 switches, the dynamic configuration of trunk interfaces and their association with an OVSDB-managed VXLAN is supported.

In a network that is divided into multiple VXLANs, each VXLAN has a VLAN ID associated with it. Packets associated with a particular VXLAN include the corresponding tag. In this situation, the interface that connects the Juniper Networks switch to a physical server in an OVSDB environment is a trunk interface that handles only tagged packets.

To determine the type of interface to create and associate with an OVSDB-managed VXLAN, the Juniper Networks switch uses the VLAN ID that you specified when configuring either the logical switch port (NSX), or the logical interface (Contrail). If you specified a valid VLAN ID other than 0 in either configuration, the switch creates a trunk interface that can handle tagged packets. (If you specified **0** as the VLAN ID, the switch creates a trunk interface that handles untagged packets.)

After the SDN controller pushes the NSX or Contrail configuration to the Juniper Networks switch, the switch dynamically creates a configuration similar to the following:

```
set interfaces ge-1/0/0 flexible-vlan-tagging
set interfaces ge-1/0/0 encapsulation extended-vlan-bridge
set interfaces ge-1/0/0 unit 10 vlan-id 10
set vlans 28805c1d-0122-495d-85df-19abd647d772 interfaces ge-1/0/0.10
```

The sample configuration sets up physical interface ge-1/0/0 as a trunk interface. It also configures a VLAN with an ID of 10 and specifies that interface ge-1/0/0.10 is a member of the VLAN. With the configuration of VLAN 10, logical interface ge-1/0/0.10 accepts incoming packets with a VLAN tag of 10 and adds a tag of 100 to each packet. Adding a tag of 100 identifies the packets as received by the VXLAN 28805c1d-0122-495d-85df-19abd647d772, which has a VNI of 100. This configuration also associates the trunk interface with VXLAN 28805c1d-0122-495d-85df-19abd647d772.

Change History Table

Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
14.1X53-D15	Starting with Junos OS Release 14.1X53-D15 for QFX5100 switches, 15.1X53-D10 for QFX10002 switches, 15.1X53-D30 for QFX10008 switches, 15.1X53-D60 for QFX10016 switches, 15.1X53-D210 for QFX5110 and QFX5200 switches, and 18.1R1 for QFX5210 switches, the dynamic configuration of trunk interfaces and their association with an OVSDB-managed VXLAN is supported.

RELATED DOCUMENTATION

Understanding the OVSDB Protocol Running on Juniper Networks Devices

show ovsdb logical-switch

VMware NSX Configuration for Juniper Networks Devices Functioning as Virtual Tunnel Endpoints

IN THIS SECTION

When implementing the Open vSwitch Database (OVSDB) management protocol and Virtual Extensible LANs (VXLANs) on a Juniper Networks device, you must perform the following tasks in VMware NSX Manager or in the NSX API:

- For each Juniper Networks device on which OVSDB-managed VXLANs and physical interfaces are configured, you must create an NSX-equivalent entity, which is known as a *gateway*.
- For each OVSDB-managed physical interface that you configure on a Juniper Networks device, you must configure a gateway service—for example, a VTEP Layer 2 gateway service.
- For each logical interface that you want to implement for a VXLAN, you must configure a logical switch port.

The configurations described in this topic enable connectivity between physical servers in the physical network and virtual machines (VMs) in the virtual network.

This topic provides a high-level summary of the tasks that you must perform to create a gateway, gateway service, and logical switch ports. Although you can create these virtual entities either in NSX Manager or in the NSX API, this topic only describes how to perform the tasks in NSX Manager. Also, this topic does not include a complete procedure for each task. Rather, it includes key NSX Manager configuration details for ensuring the correct configuration of the virtual entities so that they function properly with the physical entities.

For complete information about performing the tasks described in this topic, see the documentation that accompanies NSX Manager.

This topic describes the following tasks:

Creating a Gateway

In NSX Manager, you must create a gateway for each Juniper Networks device on which OVSDB-managed VXLANs and physical interfaces are configured. Table 7 on page 67 provides a summary of key configuration fields in NSX Manager and how to configure them when creating a gateway.

Table 7: Key Configurations to Create a Gateway in NSX Manager

NSX Manager Configuration Page or Dialog Box	NSX Manager Configuration Field	How to Configure
Type	Transport Node Type	Select **Gateway**.
Properties	VTEP Enabled	Select **VTEP Enabled**.
Credential	Type	Select **Management Address**.
Credential	Management Address	Specify the management IP address of the Juniper Networks device.
Connections/Create Transport Connector	Transport Type	Select **VXLAN**.
Connections/Create Transport Connector	Transport Zone UUID	Select the UUID of an existing transport zone, or create a new transport zone.
Connections/Create Transport Connector	IP Address	Specify the IP address of the loopback interface (lo0) of the Juniper Networks device.

Creating a Gateway Service

In NSX Manager, you must create a gateway service for each OVSDB-managed physical interface that you configure on a Juniper Networks device. Creating a gateway service essentially does the following for each OVSDB-managed physical interface:

- Specifies a gateway service—for example, a VTEP Layer 2 gateway service.
- Binds the interface to a gateway that you created in "Creating a Gateway" on page 67.

Before you start this task, you must complete the following configurations:

- A gateway for the Juniper Networks device on which the OVSDB-managed physical interfaces are configured. See "Creating a Gateway" on page 67.
- The OVSDB-managed physical interfaces on the Juniper Networks device. For information about configuring OVSDB-managed interfaces on Juniper Networks devices that support the dynamic configuration of VXLANs, see "Setting Up OVSDB on Juniper Networks Devices That Support the Dynamic Configuration of VXLANs" on page 53. For information about configuring OVSDB-managed interfaces on Juniper Networks devices that support the manual configuration of VXLANs, see *Setting Up the OVSDB Protocol on Juniper Networks Devices that Support Manual Configuration of VXLANs*.

Table 8 on page 68 provides a summary of key configuration fields in NSX Manager and how to configure them when creating a gateway service.

Table 8: Key Configurations to Create a Gateway Service in NSX Manager

NSX Manager Configuration Page or Dialog Box	NSX Manager Configuration Field	How to Configure
Type	Gateway Service Type	Select **VTEP L2 Gateway Service**.
Transport Nodes/Edit Gateway	Transport Node	Select the gateway that you created for the Juniper Networks device.
Transport Nodes/Edit Gateway	Port ID	Select an OVSDB-managed physical interface configured on the Juniper Networks device.

Creating a Logical Switch Port

In NSX Manager, you must create a logical switch port for each logical interface that you plan to implement for a VXLAN. Creating the logical switch port essentially does the following for each logical interface:

- Binds the logical switch port to a logical switch that you created in NSX Manager or in the NSX API.
- Binds the logical interface to a gateway service that you configured in "Creating a Gateway Service" on page 67.

Before you start this task, you must complete the following configurations:

- A logical switch with which this logical port is associated. For information about configuring a logical switch, see the VMware documentation that accompanies NSX Manager or the NSX API.

- A gateway service that specifies the OVSDB-managed physical interface with which the logical interface is associated. See "Creating a Gateway Service" on page 67.

Table 9 on page 69 provides a summary of key configuration fields in NSX Manager and how to configure them when creating a logical switch port.

Table 9: Key Configurations to Create a Logical Switch Port in NSX Manager

NSX Manager Configuration Page or Dialog Box	NSX Manager Configuration Field	How to Configure
Logical Switch	Logical Switch UUID	Select the UUID of a logical switch.
Attachment	Attachment Type	Select **VTEP L2 Gateway**.
Attachment	VTEP L2 Gateway Service UUID	Select the UUID of a gateway service.
Attachment	VLAN	Select **0** to specify that the port handles untagged packets. Select **1** through **4000** to specify that the port handles tagged packets. **NOTE**: VLAN ID 4094 is reserved for a native VLAN in an OVSDB environment. Specifying this VLAN ID results in an error message. Do not specify this VLAN ID or any VLAN ID not in the accepted range.

RELATED DOCUMENTATION

OVSDB and VXLAN Configuration Workflows for VMware NSX Environment

Example: Setting Up a VXLAN Layer 2 Gateway and OVSDB Connections in a VMware NSX Environment (Trunk Interfaces Supporting Untagged Packets)

IN THIS SECTION

In a physical network, a Juniper Networks device that supports Virtual Extensible LAN (VXLAN) can function as a hardware virtual tunnel endpoint (VTEP). In this role, the Juniper Networks device encapsulates Layer 2 Ethernet frames received from software applications that run directly on a physical server in VXLAN packets. The VXLAN packets are tunneled over a Layer 3 transport network. Upon receipt of the VXLAN packets, software VTEPs in the virtual network de-encapsulate the packets and forward the packets to virtual machines (VMs).

In this VXLAN environment, you can also include VMware NSX controllers and implement the Open vSwitch Database (OVSDB) management protocol on the Juniper Networks device that functions as a hardware VTEP. The Junos OS implementation of OVSDB provides a means through which VMware NSX controllers and Juniper Networks devices can exchange MAC addresses of entities in the physical and virtual networks. This exchange of MAC addresses enables the Juniper Networks device that functions as a hardware VTEP to forward traffic to software VTEPs in the virtual network and software VTEPs in the virtual network to forward traffic to the Juniper Networks device in the physical network.

This example explains how to configure a Juniper Networks device that supports VXLAN as a hardware VTEP. (The VTEP serves as a Layer 2 gateway.) This example also explains how to configure this device with an OVSDB connection to an NSX controller.

In this example, only one VXLAN is deployed. Given this scenario, the packets exchanged between an application running on a physical server and a VM in the VXLAN are untagged. As a result, the QFX Series switch dynamically configures a logical trunk interface for the connection between the physical server and the switch, as well as a native VLAN. The native VLAN enables the trunk interface to handle the untagged packets.

Requirements

This example includes the following hardware and software components:

- A physical server on which software applications directly run.
- A QFX10002 switch running Junos OS software 15.1X53-D30 or later.
- On the QFX Series switch, physical interface ge-1/0/0 provides a connection to physical server 1.
- A cluster of five NSX controllers. (In this example, you explicitly configure a connection with one NSX controller.)
- NSX Manager.
- A service node that handles the replication and forwarding of Layer 2 broadcast, unknown unicast, and multicast (BUM) traffic within the VXLAN used in this example.
- A host that includes VMs managed by a hypervisor, which includes a software VTEP.

Before you begin:

- Create an SSL private key and certificate, and install them in the **/var/db/certs** directory of the QFX Series switch. See "Creating and Installing an SSL Key and Certificate on a Juniper Networks Device for Connection with SDN Controllers" on page 52.
- Using NSX Manager, specify the IP address of the service node.

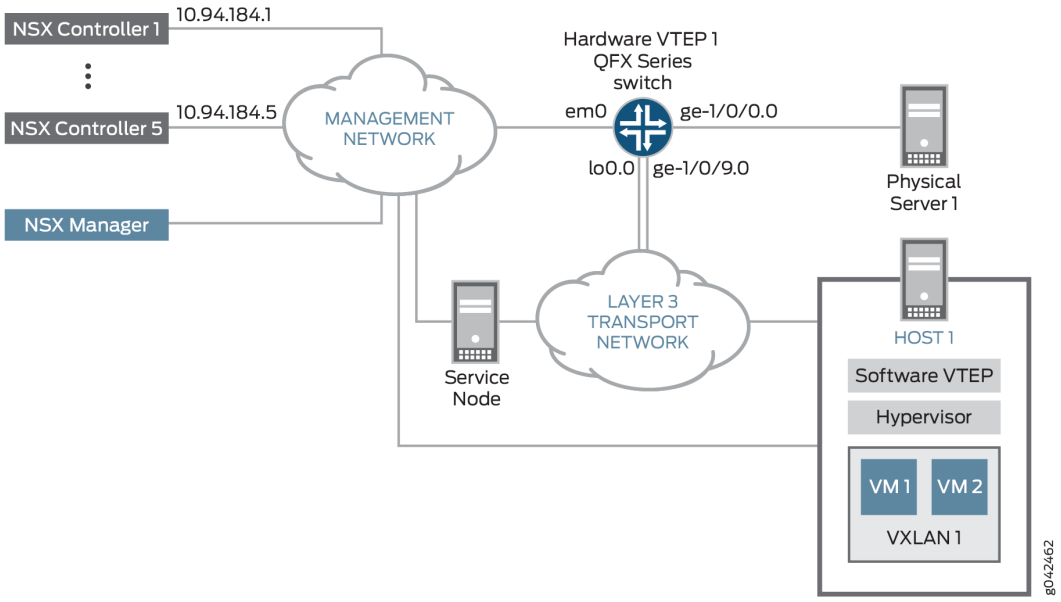
For information about using NSX Manager, see the documentation that accompanies these VMware products.

Overview and Topology

IN THIS SECTION

Figure 6 on page 72 shows a topology in which a software application running directly on physical server 1 in the physical network needs to communicate with virtual machine VM 1 in VXLAN 1 and vice versa.

Figure 6: VXLAN-OVSDB Layer 2 Gateway Topology



To establish communication between the software application on physical server 1 and VM 1 in VXLAN 1, a connection with an NSX controller is explicitly configured on the management interface of the QFX Series switch by using the Junos OS CLI.

Also, some entities in the VXLAN-OVSDB topology must be configured in both NSX Manager and on the QFX Series switch. Table 10 on page 72 provides a summary of the entities that must be configured and where they must be configured.

Table 10: NSX Manager and Junos OS Entities That Must Be Configured

Entities	What Must Be Configured in NSX Manager	What Must Be Configured on a QFX Series Switch
VXLAN 1	Logical switch for VXLAN 1	VXLAN 1 **NOTE**: The QFX Series switch dynamically configures this VXLAN.
Physical interface (ge-1/0/0) between physical server 1 and QFX Series switch	A gateway service. For gateway service type, select VTEP L2 Gateway service.	OVSDB management. Specify that interface ge-1/0/0 is managed by OVSDB.

Table 10: NSX Manager and Junos OS Entities That Must Be Configured *(Continued)*

Entities	What Must Be Configured in NSX Manager	What Must Be Configured on a QFX Series Switch
One logical interface (ge-1/0/0.0) associated with VXLAN 1	One logical switch port for VXLAN 1. For this port, specify VLAN number 0. **NOTE**: A VLAN number of 0 indicates that the port must handle untagged packets.	One logical interface (ge-1/0/0.0) for VXLAN 1. **NOTE**: The QFX Series switch dynamically configures this logical interface.
QFX Series switch (hardware VTEP 1)	Gateway	–

In NSX Manager, a logical switch for VXLAN 1 is configured. In this configuration, a VXLAN network identifier (VNI) of 100 is specified. Also, the universally unique identifier (UUID) that NSX Manager assigns to the logical switch is 28805c1d-0122-495d-85df-19abd647d772. Based on this configuration, the QFX Series switch dynamically creates the following configuration for a Junos OS-equivalent VXLAN:

```
set vlans 28805c1d-0122-495d-85df-19abd647d772 vxlan vni 100
```

Based on the gateway service and logical switch port configuration (VLAN number 0) in NSX Manager, the QFX Series switch dynamically creates the following configuration for a Junos OS-equivalent interface:

```
set interfaces ge-1/0/0 flexible-vlan-tagging
set interfaces ge-1/0/0 native-vlan-id 4094
set interfaces ge-1/0/0 encapsulation extended-vlan-bridge
set interfaces ge-1/0/0 unit 0 vlan-id 4094
set vlans 28805c1d-0122-495d-85df-19abd647d772 interface ge-1/0/0.0
```

This configuration sets physical interface ge-1/0/0 as a trunk interface. It also configures a native VLAN with an ID of 4094. The configuration creates logical interface ge-1/0/0.0 and specifies that it is a member of the native VLAN. As a result, logical interface ge-1/0/0.0 handles incoming untagged packets.

The configuration also associates logical interface ge-1/0/0.0 with VXLAN 28805c1d-0122-495d-85df-19abd647d772.

Table 11 on page 74 provides a summary of the VXLAN-OVSDB topology components that are configured on the QFX Series switch and the configuration settings for each component.

Topology

Table 11: Components of the Topology for Setting Up a VXLAN Layer 2 Gateway and OVSDB Connections

Component	Setting
NSX controller	IP address: 10.94.184.1
OVSDB-managed physical interface	Interface name: ge-1/0/0 Native VLAN ID: 4094
Logical interface	**NOTE**: The QFX Series switch dynamically creates this logical interface configuration, which is based on the gateway service configuration and logical switch port configuration in NSX Manager. Therefore, no manual configuration is required. Interface name: ge-1/0/0.0 Interface type: trunk Member of native VLAN 4094 Associated with VXLAN 28805c1d-0122-495d-85df-19abd647d772
OVSDB-managed VXLAN	**NOTE**: The QFX Series switch dynamically creates this VXLAN configuration, which is based on the logical switch configuration in NSX Manager. Therefore, no manual configuration is required. For VXLAN 1: VXLAN name: 28805c1d-0122-495d-85df-19abd647d772 VNI: 100

Table 11: Components of the Topology for Setting Up a VXLAN Layer 2 Gateway and OVSDB Connections *(Continued)*

Component	Setting
OVSDB tracing operations	Filename: /var/log/ovsdb File size: 10 MB Flag: All
Hardware VTEP source identifier	Source interface: loopback (lo0.0) Source IP address: 10.17.17.17/32
Handling of Layer 2 BUM traffic in VXLAN 28805c1d-0122-495d-85df-19abd647d772	Service node **NOTE**: By default, one or more service nodes handle Layer 2 BUM traffic within a VXLAN; therefore, no manual configuration is required.

Non-OVSDB and Non-VXLAN Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [edit] hierarchy level, and then enter commit from configuration mode.

```
set interfaces ge-1/0/9 unit 0 family inet address 10.40.40.1/24
set routing-options static route 10.19.19.19/32 next-hop 10.40.40.2
set routing-options router-id 10.17.17.17
```

```
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols ospf area 0.0.0.0 interface ge-1/0/9.0
```

Procedure

Step-by-Step Procedure

To configure the Layer 3 network over which the packets exchanged between the physical server and VMs are tunneled:

1. Configure the Layer 3 interface.

```
[edit interfaces]
user@switch# set ge-1/0/9 unit 0 family inet address 10.40.40.1/24
```

2. Set the routing options.

```
[edit routing-options]
user@switch# set static route 10.19.19.19/32 next-hop 10.40.40.2
user@switch# set router-id 10.17.17.17
```

3. Configure the routing protocol.

```
[edit protocols]
user@switch# set ospf area 0.0.0.0 interface lo0.0
user@switch# set ospf area 0.0.0.0 interface ge-1/0/9.0
```

OVSDB and VXLAN Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [edit] hierarchy level, and then enter commit from configuration mode.

```
set switch-options ovsdb-managed
set protocols ovsdb controller 10.94.184.1
set protocols ovsdb interfaces ge-1/0/0
set protocols ovsdb traceoptions file ovsdb
set protocols ovsdb traceoptions file size 10m
set protocols ovsdb traceoptions flag all
set interfaces lo0 unit 0 family inet address 10.17.17.17/32 primary
set interfaces lo0 unit 0 family inet address 10.17.17.17/32 preferred
set switch-options vtep-source-interface lo0.0
```

Procedure

Step-by-Step Procedure

To configure the QFX Series switch as a hardware VTEP with an OVSDB connection to an NSX controller:

1. Enable the QFX Series switch to dynamically configure OVSDB-managed VXLANs and associated interfaces.

```
[edit switch-options]
user@switch# ovsdb-managed
```

2. Explicitly configure a connection with an NSX controller.

```
[edit protocols]
user@switch# set ovsdb controller  10.94.184.1
```

3. Specify that the interface between hardware VTEP 1 and physical server 1 is managed by OVSDB.

```
[edit protocols]
user@switch# set ovsdb interfaces ge-1/0/0
```

4. Set up OVSDB tracing operations.

```
[edit protocols]
user@switch# set ovsdb traceoptions file ovsdb
user@switch# set ovsdb traceoptions file size 10m
user@switch# set ovsdb traceoptions flag all
```

5. Specify an IP address for the loopback interface. This IP address serves as the source IP address in the outer header of any VXLAN-encapsulated packet.

```
[edit interfaces]
user@switch# set lo0 unit 0 family inet address 10.17.17.17/32 primary
user@switch# set lo0 unit 0 family inet address 10.17.17.17/32 preferred
```

6. Set the loopback interface as the interface that identifies hardware VTEP 1.

```
[edit switch-options]
user@switch# set vtep-source-interface lo0.0
```

7. In NSX Manager, configure a logical switch for VXLAN 1. See the VMware documentation that accompanies NSX Manager.

8. In NSX Manager, configure a gateway for the QFX Series switch, and configure a gateway service and logical switch port for the logical interface (ge-1/0/0.0). See "VMware NSX Configuration for Juniper Networks Devices Functioning as Virtual Tunnel Endpoints" on page 66.

Verification

IN THIS SECTION

Confirm that the configuration is working properly:

Verifying the Logical Switch Configuration

Purpose

Verify that the configuration of the logical switch with the UUID of 28805c1d-0122-495d-85df-19abd647d772 is present in the OVSDB schema for physical devices and that the Flags field of the `show ovsdb logical switch` output displays `Created by both`.

Action

From operational mode, enter the `show ovsdb logical-switch` command.

```
user@switch> show ovsdb logical-switch
Logical switch information:
Logical Switch Name: 28805c1d-0122-495d-85df-19abd647d772
Flags: Created by both
VNI: 100
Num of Remote MAC: 1
Num of Local MAC: 0
```

Meaning

The output verifies that the configuration for the logical switch is present. The `Created by both` state indicates that the logical switch was configured in NSX Manager, and that the QFX Series switch dynamically created the corresponding VXLAN. In this state, the logical switch and the VXLAN are operational.

If the state of the logical switch is something other than `Created by both`, see "Troubleshooting a Nonoperational Logical Switch and Corresponding Junos OS OVSDB-Managed VXLAN" on page 120.

Verifying the MAC Address of VM 1

Purpose

Verify that the MAC address of VM 1 is present in the OVSDB schema.

Action

From operational mode, enter the `show ovsdb mac remote` command.

```
user@switch> show ovsdb mac remote
Logical Switch Name: 28805c1d-0122-495d-85df-19abd647d772
  Mac                   IP                Encapsulation      Vtep
  Address               Address                              Address
  a8:59:5e:f6:38:90     0.0.0.0           Vxlan over Ipv4    10.17.17.17
```

Meaning

The output shows that the MAC address for VM 1 is present and is associated with the logical switch with the UUID of 28805c1d-0122-495d-85df-19abd647d772. Given that the MAC address is present, VM 1 is reachable through the QFX Series switch, which functions as a hardware VTEP.

Verifying the NSX Controller Connection

Purpose

Verify that the connection with the NSX controller is up.

Action

From operational mode, enter the `show ovsdb controller` command to verify that the controller connection state is `up`.

```
user@switch> show ovsdb controller
VTEP controller information:
Controller IP address: 10.94.184.1
Controller protocol: ssl
Controller port: 6632
Controller connection: up
Controller seconds-since-connect: 542325
Controller seconds-since-disconnect: 542346
Controller connection status: active
```

Meaning

The output shows that the connection state of the NSX controller is up, in addition to other information about the controller. The `up` state of the NSX controller indicates that OVSDB is enabled on the QFX Series switch.

Verifying the OVSDB-Managed Interface

Purpose

Verify that interface ge-1/0/0.0 is managed by OVSDB.

Action

From operational mode, enter the `show ovsdb interface` command to verify that interface ge-1/0/0.0 is managed by OVSDB.

```
user@switch> show ovsdb interface
Interface  VLAN ID Bridge-domain
ge-1/0/0   0       28805c1d-0122-495d-85df-19abd647d772
```

Meaning

The output shows that interface ge-1/0/0 is managed by OVSDB. It also indicates that the interface is associated with VXLAN 28805c1d-0122-495d-85df-19abd647d772, which has a VLAN ID of 0.

Example: Setting Up a VXLAN Layer 2 Gateway and OVSDB Connections in a VMware NSX Environment (Trunk Interfaces Supporting Tagged Packets)

IN THIS SECTION

In a physical network, a Juniper Networks device that supports Virtual Extensible LAN (VXLAN) can function as a hardware virtual tunnel endpoint (VTEP). In this role, the Juniper Networks device encapsulates Layer 2 Ethernet frames received from software applications that run directly on a physical server in VXLAN packets. The VXLAN packets are tunneled over a Layer 3 transport network. Upon receipt of the VXLAN packets, software VTEPs in the virtual network de-encapsulate the packets and forward the packets to virtual machines (VMs).

In this VXLAN environment, you can also include VMware NSX controllers and implement the Open vSwitch Database (OVSDB) management protocol on the Juniper Networks device that functions as a hardware VTEP. The Junos OS implementation of OVSDB provides a means through which VMware NSX controllers and Juniper Networks devices can exchange MAC addresses of entities in the physical and virtual networks. This exchange of MAC addresses enables the Juniper Networks device that functions as a hardware VTEP to forward traffic to software VTEPs in the virtual network and software VTEPs in the virtual network to forward traffic to the Juniper Networks device in the physical network.

This example explains how to configure a Juniper Networks device that supports VXLAN as a hardware VTEP. (The VTEP serves as a Layer 2 gateway.) This example also explains how to configure this device with an OVSDB connection to an NSX controller.

Starting with Junos OS Release 14.1X53-D15 for QFX5100 switches, 15.1X53-D10 for QFX10002 switches, 15.1X53-D30 for QFX10008 switches, 15.1X53-D60 for QFX10016 switches, 15.1X53-D210 for QFX5110 and QFX5200 switches, and 18.1R1 for QFX5210 switches, the dynamic configuration of trunk interfaces and their association with an OVSDB-managed VXLAN is supported. In this example, an application running directly on a physical server needs to communicate with a VM in a VXLAN, while another application on the physical server needs to communicate with VMs in another VXLAN. Therefore, the packets exchanged between the applications running on the physical server and the respective VMs with which they must communicate are tagged. As a result, a trunk interface is used for the connection between the physical server and the Juniper Networks device.

Requirements

This example includes the following hardware and software components:

- A physical server on which software applications directly run.
- A Juniper Networks switch that supports VXLAN and OVSDB. This switch can be a QFX5100 switch running Junos OS Release 14.1X53-D15 or later.

- On the Juniper Networks switch, physical interface ge-1/0/0 provides a connection to physical server 1.
- A cluster of five NSX controllers. (In this example, you explicitly configure a connection with one NSX controller.)
- NSX Manager.
- A service node that handles the replication and forwarding of Layer 2 broadcast, unknown unicast, and multicast (BUM) traffic within the VXLANs.
- Two hosts that include VMs. Each host is managed by a hypervisor, and each hypervisor includes a software VTEP.

Before you begin the configuration, you must perform the following tasks:

- Create an SSL private key and certificate, and install them in the **/var/db/certs** directory of the Juniper Networks switch. See *Creating and Installing an SSL Key and Certificate on a Juniper Networks Device for Connection with SDN Controllers*.
- Using NSX Manager, specify the IP address of the service node.

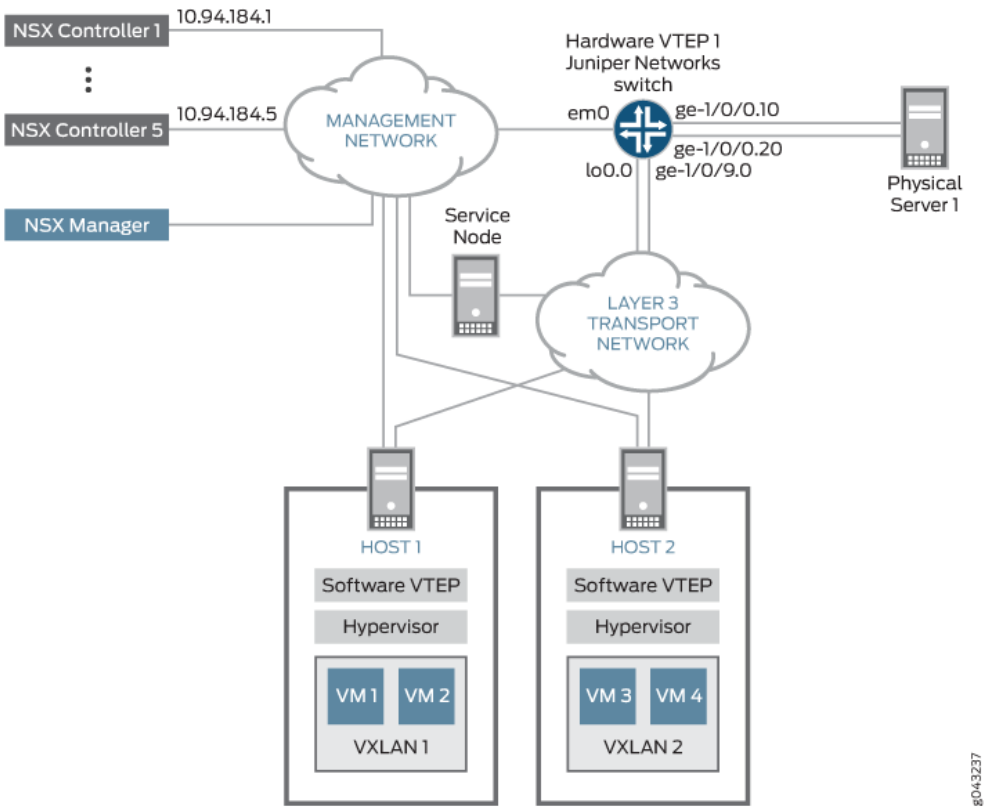
For information about using NSX Manager, see the documentation that accompanies NSX Manager.

Overview and Topology

IN THIS SECTION

Figure 7 on page 84 shows a topology in which a software application running directly on physical server 1 in the physical network needs to communicate with virtual machine VM 1 in VXLAN 1 and vice versa, and another software application on physical server 1 needs to communicate with virtual machines VM 3 and VM 4 in VXLAN 2 and vice versa.

Figure 7: VXLAN/OVSDB Layer 2 Gateway Topology



To establish communication between the software applications on physical server 1 and the VMs in VXLANs 1 and 2, some entities in the VXLAN-OVSDB topology must be configured in both NSX Manager and on the Juniper Networks switch. Table 12 on page 85 provides a summary of the entities that must be configured and where they must be configured.

NOTE: The term used for an entity configured in NSX Manager can differ from the term used for essentially the same entity configured on the Junos Network switch. To prevent confusion, Table 12 on page 85 shows the NSX Manager and Junos OS entities side-by-side.

Table 12: NSX Manager and Junos OS Entities That Must Be Configured

Entities	What Must Be Configured In NSX Manager	What Must Be Configured on Juniper Networks Switch
VXLAN 1 VXLAN 2	Logical switch for VXLAN 1 Logical switch for VXLAN 2	VXLAN 1 VXLAN 2 **NOTE**: The Juniper Networks switch dynamically configures these VXLANs.
Interface (ge-1/0/0) between physical server 1 and Juniper Networks switch	A gateway service. For gateway service type, select VTEP L2 gateway service.	OVSDB management. Specify that interface ge-1/0/0 is managed by OVSDB.
One logical interface associated with VXLAN 1 One logical interface associated with VXLAN 2	One logical switch port for VXLAN 1. For this port, specify VLAN number 10. One logical switch port for VXLAN 2. For this port, specify VLAN number 20. **NOTE**: A VLAN number from **1** through **4000** indicates that the port is a trunk port.	One logical interface (ge-1/0/0.10) for VXLAN 1 One logical interface (ge-1/0/0.20) for VXLAN 2 **NOTE**: The Juniper Networks switch dynamically configures these logical interfaces.
Juniper Networks switch (hardware VTEP 1)	Gateway	–

Based on the configuration of the entities in NSX Manager as described in Table 12 on page 85, the Juniper Networks switch dynamically creates VXLANs 1 and 2 and their associated logical interfaces. Table 13 on page 86 provides the relevant NSX Manager configuration and the resulting VXLANs and associated logical interfaces that the Juniper Networks switch dynamically configures.

Table 13: NSX Manager Configurations and Dynamic Configurations by Juniper Networks Switch

NSX Manager Configuration: Logical Switch and Logical Switch Port	VXLANs and Associated Logical Interfaces Dynamically Configured By Juniper Networks Switch

Logical switch configuration:
 	
UUID: 28805c1d-0122-495d-85df-19abd647d772
 	
VNI: 100
 	
Logical switch port configuration:
 	
VLAN ID: 10
	For VXLAN 1:
	
	`set vlans 28805c1d-0122-495d-85df-19abd647d772 vxlan vni 100`
	
	For associated logical interface ge-1/0/0.10:
	
	`set interfaces ge-1/0/0 flexible-vlan-tagging`
	`set interfaces ge-1/0/0 encapsulation extended-vlan-bridge`
	`set interfaces ge-1/0/0 unit 10 vlan-id 10`
	`set vlans 28805c1d-0122-495d-85df-19abd647d772 interfaces ge-1/0/0.10`

Logical switch configuration:
 	
UUID: 9acc24b3-7b0a-4c2e-b572-3370c3e1acff
 	
VNI: 200
 	
Logical switch port configuration:
 	
VLAN ID: 20
	For VXLAN 2:
	
	`set vlans 9acc24b3-7b0a-4c2e-b572-3370c3e1acff vxlan vni 200`
	
	For associated logical interface ge-1/0/0.20:
	
	`set interfaces ge-1/0/0 flexible-vlan-tagging`
	`set interfaces ge-1/0/0 encapsulation extended-vlan-bridge`
	`set interfaces ge-1/0/0 unit 20 vlan-id 20`
	`set vlans 9acc24b3-7b0a-4c2e-b572-3370c3e1acff interfaces ge-1/0/0.20`

For VXLANs 1 and 2, the Juniper Networks switch uses the UUIDs and VNI values that were provided for the corresponding logical switches.

In the logical switch port configurations in NSX Manager, VLAN ID values 10 and 20 and logical switch mappings are specified. As a result, the Juniper Networks switch creates logical interfaces ge-1/0/0.10 and ge-1/0/0.20, respectively. Both of these logical interfaces function as trunk interfaces. The Juniper Networks switch also maps the logical interfaces ge-1/0/0.10 and ge-1/0/0.20 to their respective VXLANs.

Based on the configurations generated by the Juniper Networks switch, the interface ge-1/0/0.10 accepts packets with a VLAN tag of 10 from VXLAN 1, and interface ge-1/0/0.20 accepts packets with a VLAN tag of 20 from VXLAN 2. On receiving packets from VXLAN 1, a VLAN tag of 100 is added to the packets, and a VLAN tag of 200 is added to packets from VXLAN 2. These tags are added to the respective packet streams to map the VLAN ID in a particular VXLAN to the corresponding VNI.

Table 14 on page 87 provides a summary of the components that are configured on the Juniper Networks switch. Unless noted, all configurations are performed manually in the Junos OS CLI.

Topology

Table 14: Components for Two VXLAN Topologies Configured on a Juniper Networks Switch that Functions as a Hardware VTEP

Components	Settings
NSX controller	IP address: 10.94.184.1
OVSDB-managed interface	Interface name: ge-1/0/0
VXLAN 1 and associated logical interface	**NOTE**: The Juniper Networks switch dynamically configures the VXLAN and associated logical interface, which are based on the logical switch and logical switch port configurations in NSX Manager. Therefore, no manual configuration is required. VXLAN name: 28805c1d-0122-495d-85df-19abd647d772 VNI: 100 Logical interface name: ge-1/0/0.10 VLAN ID: 10 Interface type: trunk

Table 14: Components for Two VXLAN Topologies Configured on a Juniper Networks Switch that Functions as a Hardware VTEP *(Continued)*

Components	Settings
VXLAN 2 and associated logical interface	**NOTE**: The Juniper Networks switch dynamically configures the VXLAN and associated interface, which are based on the logical switch and logical switch port configurations in NSX Manager. Therefore, no manual configuration is required. VXLAN name: VXLAN 9acc24b3-7b0a-4c2e-b572-3370c3e1acff VNI: 200 Logical interface name: ge-1/0/0.20 VLAN ID: 20 Interface type: trunk
OVSDB tracing operations	Filename: /var/log/ovsdb File size: 10 MB Flag: All
Hardware VTEP source identifier	Source interface: loopback (lo0.0) Source IP address: 10.17.17.17/32
Handling of Layer 2 BUM traffic within VXLAN 28805c1d-0122-495d-85df-19abd647d772 and within VXLAN 9acc24b3-7b0a-4c2e-b572-3370c3e1acff	Service node **NOTE**: By default, one or more service nodes handle Layer 2 BUM traffic in a VXLAN; therefore, no configuration is required.

Non-OVSDB and Non-VXLAN Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [edit] hierarchy level, and then enter commit from configuration mode.

```
set interfaces ge-1/0/9 unit 0 family inet address 10.40.40.1/24
set routing-options static route 10.19.19.19/32 next-hop 10.40.40.2
set routing-options router-id 10.17.17.17
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols ospf area 0.0.0.0 interface ge-1/0/9.0
```

Procedure

Step-by-Step Procedure

To configure the Layer 3 network over which the packets exchanged between physical server 1 and VM1 are tunneled:

1. Configure the Layer 3 interface.

```
[edit interfaces]
user@switch# set ge-1/0/9 unit 0 family inet address 10.40.40.1/24
```

2. Set the routing options.

```
[edit routing-options]
user@switch# set static route 10.19.19.19/32 next-hop 10.40.40.2
user@switch# set router-id 10.17.17.17
```

3. Configure the routing protocol.

```
[edit protocols]
user@switch# set ospf area 0.0.0.0 interface lo0.0
user@switch# set ospf area 0.0.0.0 interface ge-1/0/9.0
```

OVSDB and VXLAN Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [edit] hierarchy level, and then enter commit from configuration mode.

```
set switch-options ovsdb-managed
set protocols ovsdb controller 10.94.184.1
set protocols ovsdb interfaces ge-1/0/0
set protocols ovsdb traceoptions file ovsdb
set protocols ovsdb traceoptions file size 10m
set protocols ovsdb traceoptions flag all
set interfaces lo0 unit 0 family inet address 10.17.17.17/32 primary
set interfaces lo0 unit 0 family inet address 10.17.17.17/32 preferred
set switch-options vtep-source-interface lo0.0
```

Procedure

Step-by-Step Procedure

To configure the Juniper Networks switch as hardware VTEP 1 and with an OVSDB connection to an NSX controller:

1. Enable the Juniper Networks switch to dynamically configure OVSDB-managed VXLANs and associated interfaces.

```
[edit switch-options]
user@switch# set ovsdb-managed
```

2. Explicitly configure a connection with an NSX controller.

```
[edit protocols]
user@switch# set ovsdb controller  10.94.184.1
```

3. Specify that interface ge-1/0/0 is managed by OVSDB.

```
[edit protocols]
user@switch# set ovsdb interfaces ge-1/0/0
```

4. Set up OVSDB tracing operations.

```
[edit protocols]
user@switch# set ovsdb traceoptions file ovsdb
user@switch# set ovsdb traceoptions file size 10m
user@switch# set ovsdb traceoptions flag all
```

5. Specify an IP address for the loopback interface. This IP address serves as the source IP address in the outer header of any VXLAN-encapsulated packets.

```
[edit interfaces]
user@switch# set lo0 unit 0 family inet address 10.17.17.17/32 primary
user@switch# set lo0 unit 0 family inet address 10.17.17.17/32 preferred
```

6. Set the loopback interface as the interface that identifies hardware VTEP 1.

```
[edit switch-options]
user@switch# set vtep-source-interface lo0.0
```

7. In NSX Manager, configure a logical switch for VXLAN 1 and a logical switch for VXLAN 2. See the documentation that accompanies NSX Manager.

8. In NSX Manager, configure a gateway for the Juniper Networks switch, a gateway service for OVSDB-managed interface ge-1/0/0, and a logical switch port for logical interface ge-1/0/0.10, which is associated with VXLAN 1, and a logical switch port for logical interface ge-1/0/0.20, which is associated with VXLAN 2.

   See *VMware NSX Configuration for Juniper Networks Devices Functioning as Virtual Tunnel Endpoints*.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying the Logical Switch Configuration

Purpose

Verify that the configuration of logical switches with the UUIDs of 28805c1d-0122-495d-85df-19abd647d772 and 9acc24b3-7b0a-4c2e-b572-3370c3e1acff are present in the OVSDB schema for physical devices and that the Flags field of the `show ovsdb logical-switch` output is Created by both.

Action

From operational mode, enter the `show ovsdb logical-switch` command.

```
user@switch> show ovsdb logical-switch
Logical switch information:
Logical Switch Name: 28805c1d-0122-495d-85df-19abd647d772
Flags: Created by both
VNI: 100
Num of Remote MAC: 1
```

```
Num of Local MAC: 0
Logical Switch Name: 9acc24b3-7b0a-4c2e-b572-3370c3e1acff
Flags: Created by both
VNI: 200
Num of Remote MAC: 2
Num of Local MAC: 0
```

Meaning

The output verifies that the configuration for the logical switches is present. The `Created by both` state indicates that the logical switches were configured in NSX Manager, and that the Juniper Networks switch dynamically configured the corresponding VXLANs. In this state, the logical switches and VXLANs are operational.

If the state of the logical switches is something other than `Created by both`, see *Troubleshooting a Nonoperational Logical Switch and Corresponding Junos OS OVSDB-Managed VXLAN*.

Verifying the MAC Addresses of VM 1, VM 3, and VM 4

Purpose

Verify that the MAC addresses of VM 1, VM 3, and VM 4 are present in the OVSDB schema.

Action

From operational mode, enter the `show ovsdb mac remote` command.

```
user@switch> show ovsdb mac remote
Logical Switch Name: 28805c1d-0122-495d-85df-19abd647d772
  Mac                   IP                Encapsulation      Vtep
  Address               Address                              Address
  a8:59:5e:f6:38:90     0.0.0.0           Vxlan over Ipv4    10.17.17.17
Logical Switch Name: 9acc24b3-7b0a-4c2e-b572-3370c3e1acff
 Mac                   IP                Encapsulation      Vtep
  Address               Address                              Address
  00:23:9c:5e:a7:f0     0.0.0.0           Vxlan over Ipv4    10.17.17.17
  00:23:9c:5e:a7:f0     0.0.0.0           Vxlan over Ipv4    10.17.17.17
```

Meaning

The output shows that the MAC addresses for VM 1, VM 3, and VM 4 are present and are associated with their respective logical switches. Given that the MAC addresses are present, VM 1, VM 3, and VM 4 are reachable through the Juniper Networks switch, which functions as a hardware VTEP.

Verifying the NSX Controller Connection

Purpose

Verify that the connection with the NSX controller is up.

Action

From operational mode, enter the `show ovsdb controller` command to verify that the controller connection state is `up`.

```
user@switch> show ovsdb controller
VTEP controller information:
Controller IP address: 10.94.184.1
Controller protocol: ssl
Controller port: 6632
Controller connection: up
Controller seconds-since-connect: 542325
Controller seconds-since-disconnect: 542346
Controller connection status: active
```

Meaning

The output shows that the connection state of the NSX controller is `up`, in addition to other information about the controller. By virtue of this connection being up, OVSDB is enabled on the Juniper Networks switch.

Verifying the OVSDB-Managed Interface

Purpose

Verify that interface ge-1/0/0 is managed by OVSDB.

Action

From operational mode, enter the `show ovsdb interface` command, and verify that logical interfaces ge-1/0/0.10 and ge-1/0/0.20 are managed by OVSDB.

```
user@switch> show ovsdb interface
Interface  VLAN ID Bridge-domain
ge-1/0/0   10      28805c1d-0122-495d-85df-19abd647d772
ge-1/0/0   20      9acc24b3-7b0a-4c2e-b572-3370c3e1acff
```

Meaning

The output shows that logical interfaces `ge-1/0/0.10` and `ge-1/0/0.20` are managed by OVSDB. It also indicates that interface `ge-1/0/0.10` is associated with VXLAN `28805c1d-0122-495d-85df-19abd647d772` and interface `ge-1/0/0.20` is associated with VXLAN `9acc24b3-7b0a-4c2e-b572-3370c3e1acff`.

Change History Table

Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
14.1X53-D15	Starting with Junos OS Release 14.1X53-D15 for QFX5100 switches, 15.1X53-D10 for QFX10002 switches, 15.1X53-D30 for QFX10008 switches, 15.1X53-D60 for QFX10016 switches, 15.1X53-D210 for QFX5110 and QFX5200 switches, and 18.1R1 for QFX5210 switches, the dynamic configuration of trunk interfaces and their association with an OVSDB-managed VXLAN is supported.

Verifying That a Logical Switch and Corresponding Junos OS OVSDB-Managed VXLAN Are Working Properly

IN THIS SECTION

Purpose

Verify the following:

- A logical switch, which is configured in an NSX environment, or a virtual network, which is configured in a Contrail environment, is learning MAC addresses in their respective environments.
- The corresponding OVSDB-managed Virtual Extensible LAN (VXLAN), which is configured on a Juniper Networks device, is learning MAC addresses in the Junos OS environment.
- The logical switch or virtual network and OVSDB-managed VXLAN are exchanging the MAC addresses learned in their respective environments so that virtual and physical servers can communicate.

Action

To verify that a logical switch or virtual network and its corresponding OVSDB-managed VXLAN are learning and exchanging MAC addresses in their respective environments, enter the `show ovsdb logical-switch` operational mode command.

```
user@device> show ovsdb logical-switch
Logical switch information:
Logical Switch Name: 28805c1d-0122-495d-85df-19abd647d772
Flags: Created by both
VNI: 100
Num of Remote MAC: 1
Num of Local MAC: 0
```

NOTE: In the Open vSwitch Database (OVSDB) schema for physical devices, the logical switch table stores information about the Layer 2 broadcast domain that you configured in a VMware NSX or Contrail environment. In the NSX environment, the Layer 2 broadcast domain is known as a *logical switch*, while in the Contrail environment, the domain is known as a *virtual network*.

In the context of the `show ovsdb logical-switch` command, the term *logical switch* refers to the logical switch or virtual network that was configured in the NSX or Contrail environments, respectively, and the corresponding configuration that was pushed to the OVSDB schema.

Meaning

The output in the Flags field (`Created by both`) indicates that the logical switch or virtual network and its corresponding OVSDB-managed VXLAN are both properly configured. In this state, the logical switch or virtual network and the VXLAN are learning and exchanging MAC addresses in their respective environments.

If the output in the Flags field displays a state other than `Created by both`, see *Troubleshooting a Nonoperational Logical Switch and Corresponding Junos OS OVSDB-Managed VXLAN*.

RELATED DOCUMENTATION

- *show ovsdb logical-switch*

CHAPTER 3

Configuring VXLANs Without an SDN Controller

IN THIS CHAPTER

Manually Configuring VXLANs on QFX Series and EX4600 Switches

IN THIS SECTION

You can configure QFX5100, QFX5110, QFX5200, QFX5210, and EX4600 switches to act as a VTEP. (If the switch is acting as a transit Layer 3 switch for downstream VTEPs, you do not need to perform the steps in this topic as no special configuration is needed.)

NOTE: To ensure that QFX Series and EX4600 switches that are configured to act as VTEPs function properly, you must enable a routing protocol, for example, OSPF, on the VTEPs' loopback interface and Layer 3 interfaces. For more information about configuring OSPF on a VTEP, see "Examples: Manually Configuring VXLANs on QFX Series and EX4600 Switches" on page 101.

Configuring a Source IP Address

On a switch that will act as a VTEP, you must configure an IP address that will be used as the source address in the outer IP header of the VXLAN packet. This is the VXLAN tunnel source address.

1. Create a reachable IPv4 address on the loopback interface.

```
[edit]
user@switch# set interfaces lo0.0 unit 0 family inet address ip-address
```

2. Configure the address to be used as the tunnel source address.

```
[edit]
user@switch# set switch-options vtep-interface-source lo0.0
```

Configuring PIM for VXLANs

If you are not using an SDN controller to create a VXLAN control plane, you must enable PIM on the switch so that the VTEP can use multicast groups to establish reachability with other VTEPs and to forward BUM traffic.

1. Enable PIM on the interface that connects to the Layer 3 network. This is the interface that performs the VXLAN encapsulation and de-encapsulation.

```
[edit]
user@switch# set protocols pim interface interface-name
```

2. Configure the address of a PIM rendezvous point.

```
[edit]
user@switch#  set protocols pim rp static address ip-address
```

Configuring VXLANs

You configure VXLANs under the `vlan` stanza (which is why QFX5100, QFX5110, QFX5200, QFX5210, and EX4600 switches support 4000 VXLANs). You must also configure the server-facing interfaces to be VLAN members.

1. Create a VLAN to VXLAN mapping and assign a multicast group address to the VXLAN. All members of a VXLAN must use the same multicast group address.

```
[edit]
user@switch# set vlans name  vlan-id ID vxlan vni ID multicast-group multicast-group-address
```

2. (Optional) Configure the switch to retain the original VLAN tag (in the inner Ethernet packet) after VXLAN encapsulation. By default, the original tag is dropped when the packet is encapsulated.

```
[edit]
user@switch# set vlans name  vxlan encapsulate-inner-vlan
```

3. (Optional) Configure the switch to de-encapsulate and accept original VLAN tags in VXLAN packets. By default, the original tag is dropped when the packet is encapsulated.

```
[edit]
user@switch# set protocols l2-learning decapsulate-accept-inner-vlan
```

4. Configure server-facing interfaces to support multiple VLANs.

```
[edit]
user@switch# set interfaces interface unit unit family ethernet-switching interface-mode trunk
[edit]
user@switch# set interfaces interface unit unit family ethernet-switching vlan members all
```

You must create a VLAN to VXLAN mapping for each VLAN that will need Layer 2 connectivity over the Layer 3 network.

RELATED DOCUMENTATION

Understanding VXLANs

Examples: Manually Configuring VXLANs on QFX Series and EX4600 Switches

IN THIS SECTION

The following examples show use cases for manually configuring VXLANs on QFX5100, QFX5110, QFX5200, QFX5210, and EX4600 switches.

Example: Configuring a VXLAN Transit Switch

IN THIS SECTION

If a QFX Series or EX4600 switch acts as a transit switch for downstream devices acting as VTEPs, you do not need to configure any VXLAN information on the QFX Series or EX4600 switch. You do need to configure PIM on the switch so that it can form the multicast tree required so that the VTEPs can establish reachablity with each other.

Requirements

This example uses the following hardware and software components:

- Two QFX5100 switches
- Junos OS Release 14.1X53-D10

Overview

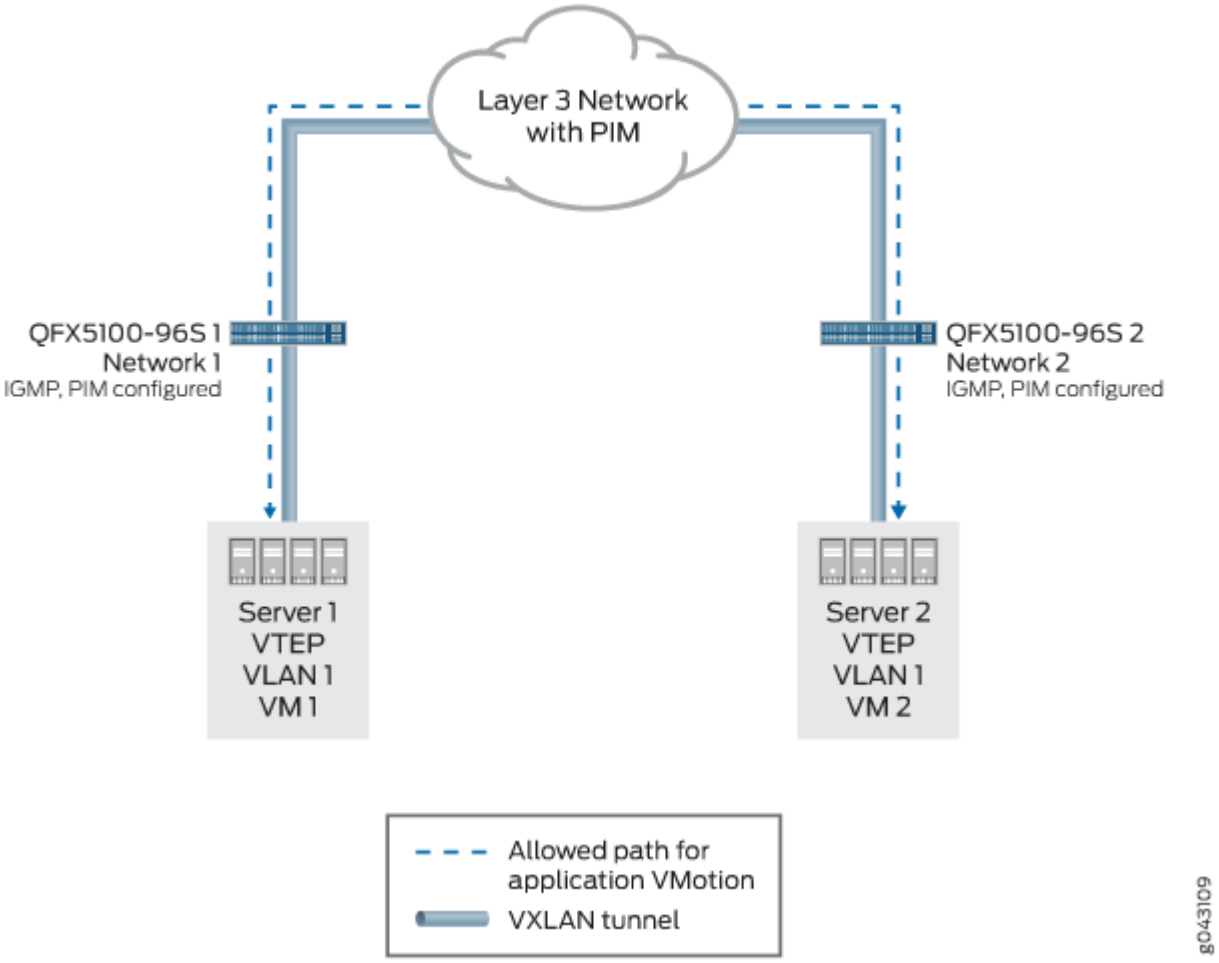
IN THIS SECTION

This example shows a simple use case in which QFX5100 switches are connected to downstream servers acting as VTEPs. The QFX5100 switches need to forward VXLAN packets between VM 1 on Server 1 and VM 2 on Server 2. Because this configuration allows Layer 2 connectivity between the VMs through the VXLAN tunnels, applications can VMotion between the VMs.

Topology

Figure 8 on page 103 shows QFX 5100 switches configured to forward VXLAN packets for downstream VTEPs.

Figure 8: QFX5100 Acting as a VXLAN Transit Switch



Configuring PIM on the Transit Switches

IN THIS SECTION

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.

```
set protocols pim interface all
set protocols pim rp static address ip-address
```

Procedure

Step-by-Step Procedure

If you are not using an SDN controller to create a VXLAN control plane, you must enable PIM on each switch so that the VTEP can use multicast groups to advertise its existence and to learn about other VTEPs. (Configuring PIM automatically enables IGMP.) You do not need to perform any VXLAN-specific configuration. Note that you also do not need to configure VLAN 1 on either switch.

1. Enable PIM.

```
[edit]
user@switch# set protocols pim interface all
```

2. Configure the address of a PIM rendezvous point.

```
[edit]
user@switch#  set protocols pim rp static address ip-address
```

SEE ALSO

Understanding VXLANs | 5

Example: Configuring a VXLAN Layer 2 Gateway

IN THIS SECTION

If a QFX Series or EX4600 switch is connected to a downstream server that hosts a VM that needs Layer 2 connectivity with another VM that is reachable only through a Layer 3 network, you must do the following:

- Configure the switch to act as a VTEP—that is, a Layer 2 gateway for downstream Layer 2 devices.
- Configure PIM on the switch so that it can form the multicast tree required for reachability with other VTEPs and to allow BUM traffic to be forwarded between the VTEPs.
- Enable a routing protocol, for example, OSPF, on the VTEP's loopback interface and Layer 3 interfaces.

Requirements

This example uses the following hardware and software components:

- Two QFX5100 switches
- Junos OS Release 14.1X53-D10

Overview

IN THIS SECTION

This example shows a use case in which QFX5100 switches act as VTEPs that allow Layer 2 connectivity between VM 1 on Server 1 and VM 2 on Server 2 so that VMotion can occur between the

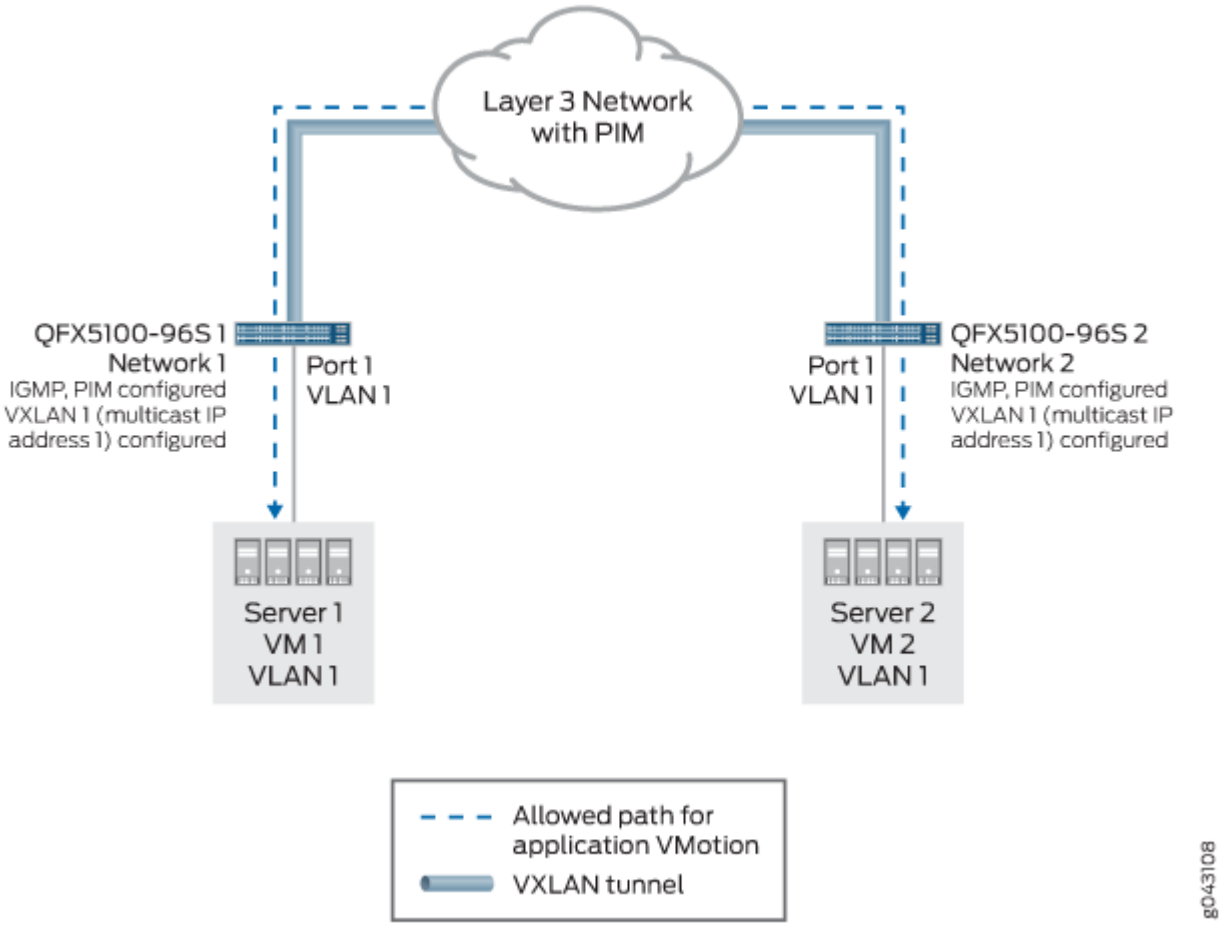
VMs. The servers in this example can be in the same or different data centers—the only constraint is that there must be Layer 3 connectivity between the QFX5100 switches. This allows your network to be very agile in response to demand for server usage or changes in bandwidth requirements.

Note that because the same VLAN exists in both Layer 2 domains and both switches encapsulate the VLAN traffic into the same VXLAN, you do not need a gateway for the VXLAN traffic in the Layer 3 network. The Layer 3 VXLAN packets are routed normally and no de-encapsulation or re-encapsulation is required.

Topology

Figure 9 on page 106 shows QFX5100 switches configured to act as VTEPs.

Figure 9: QFX5100 Acting as a VTEP



Configuring the Switches

IN THIS SECTION

CLI Quick Configuration

To quickly configure the QFX5100-96S 1 in 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.

```
set interfaces lo0 unit 0 family inet address 10.1.1.1
set switch-options vtep-source-interface lo0.0
set protocols pim interface lo0.0
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols pim interface xe-0/0/0.0
set protocols ospf area 0.0.0.0 interface xe-0/0/0.0
set protocols pim rp static address 10.2.2.2
set vlans VLAN1 vlan-id 100 vxlan vni 100 multicast-group 233.252.0.2
set vlans VLAN1 vxlan encapsulate-inner-vlan
set vlans VLAN1 vxlan unreachable-vtep-aging-timer 600
set protocols l2-learning decapsulate-accept-inner-vlan
set interfaces xe-0/0/0 unit 0 family inet address 10.2.2.100/24
set interfaces xe-0/0/1 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/1 unit 0 family ethernet-switching vlan members all
set protocols ospf enable
set protocols ospf area 0.0.0.0 interface em0.0 disable
set protocols ospf area 0.0.0.0 interface all
```

The configuration for QFX5100-96S 2 is identical except for changes to a few of the addresses:

```
set interfaces lo0 unit 0 family inet address 10.1.1.2
set switch-options vtep-source-interface lo0.0
set protocols pim interface lo0.0
```

```
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols pim interface xe-0/0/0.0
set protocols ospf area 0.0.0.0 interface xe-0/0/0.0
set protocols pim rp static address 10.2.2.2
set vlans VLAN1 vlan-id 100 vxlan vni 100 multicast-group 233.252.0.2
set vlans VLAN1 vxlan encapsulate-inner-vlan
set vlans VLAN1 vxlan unreachable-vtep-aging-timer 600
set protocols l2-learning decapsulate-accept-inner-vlan
set interfaces xe-0/0/0 unit 0 family inet address 10.2.2.200/24
set interfaces xe-0/0/1 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/1 unit 0 family ethernet-switching vlan members all
set protocols ospf enable
set protocols ospf area 0.0.0.0 interface em0.0 disable
set protocols ospf area 0.0.0.0 interface all
```

NOTE: You must configure the same multicast group address for `VLAN1` on both switches.

Procedure

Step-by-Step Procedure

Perform the following procedure on both switches to set up the example configuration.

1. Create a reachable IPv4 address on the loopback interface.

   ```
   [edit]
   user@switch# set interfaces lo0 unit 0 family inet address 10.1.1.1
   ```

   For switch QFX5100-96S 2, use address `10.1.1.2`.

2. Configure the loopback interface—and therefore, its associated address—to be used as the tunnel source address.

   ```
   [edit]
   user@switch# set switch-options vtep-source-interface lo0.0
   ```

3. Enable PIM on the loopback interface.

```
[edit]
user@switch# set protocols pim interface lo0.0
```

4. Add the loopback interface to OSPF area 0.0.0.0.

```
[edit]
user@switch# set protocols ospf area 0.0.0.0 interface lo0.0
```

5. Enable PIM on the interface that connects to the Layer 3 network.

```
[edit]
user@switch# set protocols pim interface xe-0/0/0.0
```

6. Add the interface that connects to the Layer 3 network to OSPF area 0.0.0.0.

```
[edit]
user@switch# set protocols ospf area 0.0.0.0 interface xe-0/0/0.0
```

7. Configure the address of a PIM rendezvous point.

```
[edit]
user@switch#  set protocols pim rp static address 10.2.2.2
```

8. Create a VLAN, map it to a VXLAN, and assign a multicast group address to the VXLAN. All members of a VXLAN must use the same multicast group address.

```
[edit]
user@switch# set vlans VLAN1 vlan-id 100 vxlan vni 100 multicast-group 233.252.0.2
```

In this example, the `vlan-id` and `vni` are both set to `100`. This is done only for simplicity and clarity. You do not need to set the `vlan-id` and `vni` to the same value.

9. (Optional) Configure the switch to retain the original VLAN tag (in the inner Ethernet packet) after VXLAN encapsulation. By default, the original tag is dropped when the packet is encapsulated.

```
[edit]
user@switch# set vlans VLAN1 vxlan encapsulate-inner-vlan
```

10. (Optional) Configure the system to age out the address for the remote VTEP (the other QFX5100 switch) if all the MAC addresses learned from that VTEP age out. The address for the remote VTEP expires the configured number of seconds after the last learned MAC address expires.

```
[edit]
user@switch# set vlans VLAN1 vxlan unreachable-vtep-aging-timer 600
```

(Optional) Configure the switch to de-encapsulate and accept original VLAN tags in VXLAN packets. By default, a preserved VLAN tag is dropped when the packet is de-encapsulated.

```
[edit]
user@switch# set protocols l2-learning decapsulate-accept-inner-vlan
```

11. Configure the interface that connects to the Layer 3 network.

```
[edit]
user@switch# set interfaces xe-0/0/0 unit 0 family inet address 10.2.2.100/24
```

For switch QFX5100-96S 2, use address `10.2.2.200`.

12. Configure the server-facing interface to support multiple VLANs.

```
[edit]
user@switch# set interfaces xe-0/0/1 unit 0 family ethernet-switching interface-mode trunk
[edit]
user@switch# set interfaces xe-0/0/1 unit 0 family ethernet-switching vlan members all
```

NOTE: Because this example shows only one VLAN, this step is not required for the example. In a real-world configuration, however, it would be required in order to support multiple VMs connected to multiple VLANs. In this case you would also need to configure additional VLAN to VXLAN mappings.

13. Enable OSPF on all interfaces that are mapped to area 0.0.0.0..

```
[edit]
user@switch# set protocols ospf enable
[edit]
user@switch# set protocols ospf area 0.0.0.0 interface em0.0 disable
[edit]
user@switch# set protocols ospf area 0.0.0.0 interface all
```

Results

From configuration mode, confirm your configuration by entering the following commands on QFX5100-96S 1. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@switch# show switch-options
```

```
vtep-source-interface lo0.0;
```

```
user@switch# show vlans
```

```
VLAN1 {
    vlan-id 100;
    vxlan {
        vni 100;
        multicast-group 233.252.0.2;
        encapsulate-inner-vlan;
    }
}
```

```
user@switch# show interfaces
```

```
xe-0/0/0 {
    unit 0 {
```

```
        family inet {
            address 10.2.2.100/24;
        }
    }
}
xe-0/0/1 {
    unit 0 {
        family ethernet-switching {
            interface-mode trunk;
            vlan {
                members all;
            }
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.1.1.1/32;
        }
    }
}
```

```
user@switch# show protocols pim
```

```
rp {
    static {
        address 10.2.2.2;
    }
}
interface xe-0/0/0.0
```

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying VXLAN Reachability

Purpose

On QFX5100-96S 1, verify that there is connectivity with the remote VTEP (QFX5100-96S 2).

Action

```
user@switch> show ethernet-switching vxlan-tunnel-end-point remote
Logical System Name       Id  SVTEP-IP         IFL   L3-Idx
<default>                 0   10.1.1.2     lo0.0    0
RVTEP-IP         IFL-Idx   NH-Id
10.1.1.2         559       1728
VNID             MC-Group-IP
100              233.252.0.2
```

Meaning

The VTEP on QFX5100-96S 2 is reachable because its IP address (the address assigned to the loopback interface) appears in the output. The output also shows that the VXLAN (VNI 100) and corresponding multicast group are configured correctly on the remote VTEP.

Verifying That the Local VTEP Is Configured Correctly

Purpose

On QFX5100-96S 1, verify that the tunnel endpoint is correct.

Action

```
user@switch> show ethernet-switching vxlan-tunnel-end-point source

Logical System Name       Id  SVTEP-IP         IFL   L3-Idx
<default>                 0   10.1.1.1     lo0.0    0
L2-RTT                   Bridge Domain             VNID     MC-Group-IP
default-switch           VLAN1+100                 100      233.252.0.2
```

Meaning

The VTEP on QFX5100-96S 1 shows the correct tunnel source IP address (assigned to the loopback interface), VLAN, and multicast group for the VXLAN.

Verifying MAC Learning from the Remote VTEP

Purpose

On QFX5100-96S 1, verify that it is learning MAC addresses from the remote VTEP.

Action

```
user@switch> show ethernet-switching table

MAC flags (S - static MAC, D - dynamic MAC, L - locally learned, P - Persistent static
           SE - statistics enabled, NM - non configured MAC, R - remote PE MAC)
Ethernet switching table : 2 entries, 2 learned
Routing instance : default-switch
    Vlan                MAC                 MAC         Age    Logical
    name                address             flags              interface
    VLAN1                00:00:00:ff:ff:ff   D             -    vtep.12345
    VLAN1                00:10:94:00:00:02   D         -        xe-0/0/0.0
```

Meaning

The output shows the MAC addresses learned from the remote VTEP (in addition to those learned on the normal Layer 2 interfaces). It also shows the logical name of the remote VTEP interface (`vtep.12345` in the above output).

Monitor the Remote Interface

Purpose

On QFX5100-96S 1, monitor traffic details for the remote VTEP interface.

Action

```
user@switch> show interface vtep.12345 detail

M    Flags: Up SNMP-Traps Encapsulation: ENET2
    VXLAN Endpoint Type: Remote, VXLAN Endpoint Address: 10.1.1.2, L2 Routing Instance: default-
switch, L3 Routing Instance: default
    Traffic statistics:
     Input  bytes  :         228851738624
     Output bytes  :                    0
     Input  packets:            714162415
     Output packets:                    0
    Local statistics:
     Input  bytes  :                    0
     Output bytes  :                    0
     Input  packets:                    0
     Output packets:                    0
    Transit statistics:
     Input  bytes  :         228851738624                    0 bps
     Output bytes  :                    0                    0 bps
     Input  packets:            714162415                    0 pps
     Output packets:                    0                    0 pps
    Protocol eth-switch, MTU: 1600, Generation: 277, Route table: 5
```

Meaning

The output shows traffic details for the remote VTEP interface. To get this information, you must supply the logical name of the remote VTEP interface (vtep.12345 in the above output), which you can learn by using the `show ethernet-switching table` command.

Verifying OSPF Neighbor

Purpose

On QFX5100-96S 1, verify that the remote VTEP (QFX5100-96S 2) has established itself as an OSPF neighbor.

Action

```
user@switch> show ospf neighbor

Address          Interface              State     ID               Pri  Dead
10.2.2.200       xe-0/0/0.0             Full      10.2.3.2          128    38
```

Meaning

The output shows that interface xe-0/0/0.0 on QFX5100-96S 2 is in the `Full` state, which means that QFX5100-96S 2 is a fully adjacent neighbor. The `Full` state also means that Layer 3 connectivity exists between QFX5100-96S 1 and QFX5100-96S 2.

SEE ALSO

Understanding VXLANs | 5

RELATED DOCUMENTATION

Understanding VXLANs

VXLAN Constraints on QFX Series and EX Series Switches

Verifying That a Local VXLAN VTEP Is Configured Correctly

IN THIS SECTION

Purpose

Verify that a local VTEP is correct.

Action

```
user@switch> show ethernet-switching vxlan-tunnel-end-point source

Logical System Name       Id  SVTEP-IP         IFL   L3-Idx
<default>                 0   10.1.1.1     lo0.0    0
L2-RTT                  Bridge Domain              VNID     MC-Group-IP
default-switch          VLAN1+100                  100      233.252.0.1
```

Meaning

The output shows the correct tunnel source IP address (loopback address), VLAN, and multicast group for the VXLAN.

RELATED DOCUMENTATION

Understanding VXLANs

Verifying MAC Learning from a Remote VTEP

IN THIS SECTION

Purpose

Verify that a local VTEP is learning MAC addresses from a remote VTEP.

Action

```
user@switch> show ethernet-switching table

MAC flags (S - static MAC, D - dynamic MAC, L - locally learned, P - Persistent static
           SE - statistics enabled, NM - non configured MAC, R - remote PE MAC)
Ethernet switching table : 2 entries, 2 learned
Routing instance : default-switch
    Vlan                MAC                 MAC         Age    Logical
    name                address             flags              interface
    VLAN1                00:00:00:ff:ff:ff   D             -    vtep.12345
    VLAN1                00:10:94:00:00:02   D         -        xe-0/0/0.0
```

Meaning

The output shows the MAC addresses learned from the remote VTEP (in addition to those learned on the normal Layer 2 interfaces). It also shows the logical name of the remote VTEP interface (`vtep.12345` in the above output).

RELATED DOCUMENTATION

Understanding VXLANs

Manually Configuring VXLANs on QFX Series and EX4600 Switches | 98

Examples: Manually Configuring VXLANs on QFX Series and EX4600 Switches | 101

3

PART

Troubleshooting

CHAPTER 4

Troubleshooting Tasks

IN THIS CHAPTER

Troubleshooting a Nonoperational Logical Switch and Corresponding Junos OS OVSDB-Managed VXLAN

IN THIS SECTION

Problem

Description

The Flags field in the `show ovsdb logical-switch` operational mode command output is one of the following:

- `Created by Controller`
- `Created by L2ALD`

- `Tunnel key mismatch`

Cause

- If the Flags field displays `Created by Controller`, a logical switch is configured in the NSX environment or a virtual network is configured in the Contrail environment. However, an equivalent VXLAN is not configured or is improperly configured on the Juniper Networks device.
- If the Flags field displays `Created by L2ALD`, a VXLAN is configured on the Juniper Networks device. However, an equivalent logical switch is not configured in the NSX environment or an equivalent virtual network is not configured in the Contrail environment.
- If the Flags field displays `Tunnel key mismatch`, the VXLAN network identifier (VNI) specified in the logical switch configuration or the VXLAN identifier specified in the virtual network configuration do not match the VNI in the equivalent VXLAN configuration.

Solution

If the Flags field displays `Created by Controller`, take the following action:

- On a QFX Series switch, verify that the `set switch-options ovsdb-managed` configuration command was issued in the Junos OS CLI. Issuing this command and committing the configuration enable the Juniper Networks device to dynamically create OVSDB-managed VXLANs.

  Another possible cause is that the L2ALD daemon has become nonfunctional. If this is the case, wait for a few seconds, reissue the `show ovsdb logical-switch` operational mode command, and recheck the setting of the Flags field.

  Another possible cause is that the Juniper Networks device dynamically configured the VXLAN and its associated logical interface, but there is an error in the configuration of these entities themselves or in an entity that was committed in the same transaction. If there is an issue with one or more of the configurations in a transaction, all configurations in the transaction, even the ones that are correctly configured, remain uncommitted and in a queue until you troubleshoot and resolve the configuration issues. As a result, the Juniper Networks device was unable to commit all configurations in the transaction. Starting with Junos OS Release 14.1X53-D26 for QFX5100 switches, Junos OS Release 15.1X53-D210 for QFX5110 and QFX5200 switches, and 18.1R1 for QFX5210 switches, you can enter the `show ovsdb commit failures` operational mode command to determine which configurations in a transaction are erroneous. After resolving the errors, enter the `clear ovsdb commit failures` command to remove the transaction from the queue and then retry committing all configurations in the transaction. Issues that can cause commitment errors include but are not limited to the detection of the same VXLAN name or VXLAN network identifier (VNI) in a dynamically configured VXLAN and in a VXLAN that was previously configured using the Junos OS CLI.

- On all other Juniper Networks devices that support VXLAN and OVSDB, determine whether a VXLAN equivalent to the logical switch configuration or virtual network configuration exists on the device. If the VXLAN is not configured, configure it using the procedure in *Configuring OVSDB-Managed VXLANs*. If a VXLAN is configured, check the VXLAN name to make sure that it is the same as the universally unique identifier (UUID) of the logical switch (NSX) or virtual network (Contrail) configuration. Also, check the VNI to make sure that the value is the same as the value in the logical switch (NSX) or virtual network (Contrail) configuration.

If the Flags field displays `Created by L2ALD`, take the following action:

- On a QFX Series switch, two issues exist. First, despite the fact that the Juniper Networks device dynamically creates OVSDB-managed VXLANs, this VXLAN was configured by using the Junos OS CLI. Second, a corresponding logical switch (NSX) or virtual network (Contrail) was not configured. To resolve both issues, configure a logical switch in the NSX environment or a virtual network in the Contrail environment. After the software-defined networking (SDN) controller pushes relevant logical switch or virtual network information to the Juniper Networks device, the device dynamically creates a corresponding VXLAN and deletes the VXLAN configured using the Junos OS CLI.

- On all other Juniper Networks devices that support VXLAN and OVSDB, determine whether an equivalent logical switch is configured in the NSX environment or a virtual network is configured in the Contrail environment. If a logical switch or virtual network is not configured, configure one, keeping in mind that a UUID is automatically generated for the logical switch or virtual network and that this UUID must be used as the name of the VXLAN. That is, the VXLAN name must be reconfigured with the logical switch or virtual network UUID.

  Another possibility is that the logical switch or virtual network configuration might exist, but the UUID of the entity might not match the VXLAN name. In the NSX or Contrail environment, check for a logical switch or virtual network, respectively, that has the same configuration as the VXLAN but has a different UUID.

If the Flags field displays `Tunnel key mismatch`, take the following action:

- For a QFX Series switch, check the configuration of the VNI in the NSX environment or the VXLAN identifier in the Contrail environment to see whether it was changed after the Juniper Networks device dynamically created the equivalent VXLAN. If it was changed, update the VNI on the QFX Series switch using the Junos OS CLI.

- On all other Juniper Networks devices that support VXLAN and OVSDB, check the value of the VNI in the NSX environment or the VXLAN identifier in the Contrail environment and the Junos OS CLI. Change the incorrect value.

Change History Table

Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
14.1X53-D26	Starting with Junos OS Release 14.1X53-D26 for QFX5100 switches, Junos OS Release 15.1X53-D210 for QFX5110 and QFX5200 switches, and 18.1R1 for QFX5210 switches, you can enter the `show ovsdb commit failures` operational mode command to determine which configurations in a transaction are erroneous.

RELATED DOCUMENTATION

- Understanding Dynamically Configured VXLANs in an OVSDB Environment | 55
- *Understanding How to Manually Configure OVSDB-Managed VXLANs*
- *show ovsdb logical-switch*
- *show ovsdb commit failures*
- *clear ovsdb commit failures*

Verifying VXLAN Reachability

IN THIS SECTION

Purpose

On the local VTEP, verify that there is connectivity with the remote VTEP.

Action

```
user@switch> show ethernet-switching vxlan-tunnel-end-point remote
Logical System Name       Id  SVTEP-IP         IFL   L3-Idx
```

```
<default>                0   10.1.1.2     lo0.0    0
RVTEP-IP         IFL-Idx   NH-Id
10.1.1.2         559       1728
VNID             MC-Group-IP
100              233.252.0.1
```

Meaning

The remote VTEP is reachable because its IP address appears in the output. The output also shows that the VXLAN (VNI 100) and corresponding multicast group are configured correctly on the remote VTEP.

RELATED DOCUMENTATION

Understanding VXLANs

Manually Configuring VXLANs on QFX Series and EX4600 Switches | 98

Examples: Manually Configuring VXLANs on QFX Series and EX4600 Switches | 101

Monitoring a Remote VTEP Interface

IN THIS SECTION

Purpose

Monitor traffic details for a remote VTEP interface.

Action

```
user@switch> show interface logical-name detail
```

```
M    Flags: Up SNMP-Traps Encapsulation: ENET2
    VXLAN Endpoint Type: Remote, VXLAN Endpoint Address: 10.1.1.2, L2 Routing Instance: default-
switch, L3 Routing Instance: default
    Traffic statistics:
     Input  bytes  :         228851738624
     Output bytes  :                    0
     Input  packets:            714162415
     Output packets:                    0
    Local statistics:
     Input  bytes  :                    0
     Output bytes  :                    0
     Input  packets:                    0
     Output packets:                    0
    Transit statistics:
     Input  bytes  :         228851738624                  0 bps
     Output bytes  :                    0                  0 bps
     Input  packets:            714162415                  0 pps
     Output packets:                    0                  0 pps
    Protocol eth-switch, MTU: 1600, Generation: 277, Route table: 5
```

Meaning

The output shows traffic details for the remote VTEP interface. To get this information, you must supply the logical name of the remote VTEP interface (vtep.12345 in the above output), which you can learn by using the `show ethernet-switching table` command.

RELATED DOCUMENTATION

Understanding VXLANs

Manually Configuring VXLANs on QFX Series and EX4600 Switches | **98**

Examples: Manually Configuring VXLANs on QFX Series and EX4600 Switches | **101**

Example: Troubleshoot a VXLAN Overlay Network with Overlay Ping and Overlay Traceroute on QFX Series Switches

IN THIS SECTION

In a Virtual Extensible LAN (VXLAN) overlay network, the existing `ping` and `traceroute` commands can verify the basic connectivity between two Juniper Networks devices that function as virtual tunnel endpoints (VTEPs) in the underlying physical network. However, in between the two VTEPs, there could be multiple routes through intermediary devices to the same destinations, and the ping and traceroute packets might successfully reach their destinations, while a connectivity issue exists in another route along which the data packets are typically forwarded.

With the introduction of the `overlay` parameter and other options in Junos OS Release 14.1X53-D30 for QFX5100 switches, you can use the `ping` and `traceroute` commands to troubleshoot a VXLAN overlay network.

For ping and traceroute mechanisms to work in a VXLAN overlay network, the ping and traceroute packets, also referred to as Operations, Administration, and Management (OAM) packets, must be encapsulated with the same VXLAN UDP headers (outer headers) as the data packets forwarded over the VXLAN segment with possible connectivity issues. If any connectivity issues arise, the overlay OAM packet would experience the same issues as the data packet.

This example shows how to use overlay ping and traceroute on a VTEP to verify the following in a VXLAN overlay network:

- Scenario 1—Verify that a particular VXLAN is configured on another VTEP.
- Scenario 2—Verify that the MAC address of a particular endpoint is associated with a VXLAN on another VTEP.
- Scenario 3—Verify that no issues exist in a particular data flow between sending and receiving endpoints.

NOTE: When issuing the `ping overlay` and `traceroute overlay` commands, the source VTEP on which you issue the command and the destination VTEP that receives the ping or traceroute packet must be Juniper Networks devices that support overlay ping and traceroute.

Requirements

This example uses the following hardware and software components:

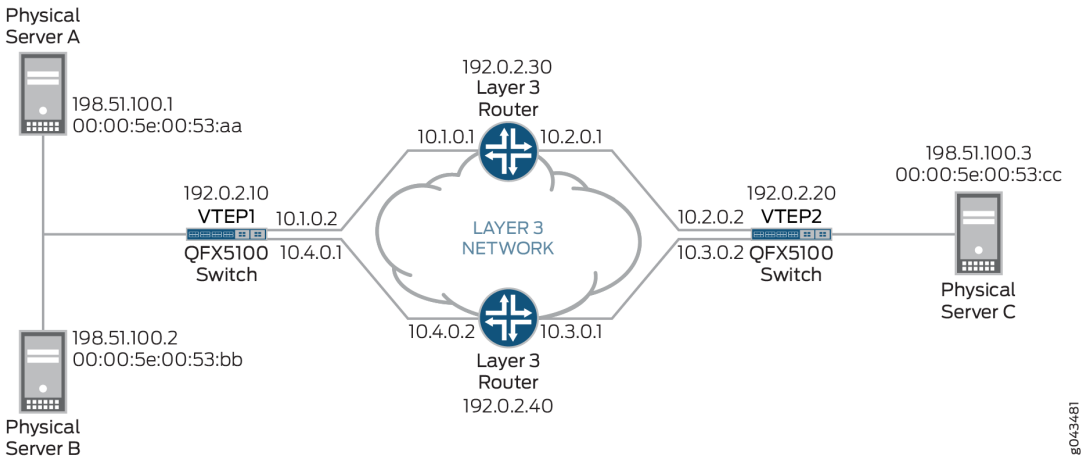
- Three physical (bare-metal) servers on which applications directly run.
- Two QFX5100 switches running Junos OS Release 14.1X53-D30 or later software. These switches function as VTEPs.
- Two Layer 3 routers, which can be Juniper Networks routers or routers provided by another vendor.

Before issuing the `ping overlay` and `traceroute overlay` commands, gather the information needed for each parameter—for example, IP addresses or MAC addresses—used for a particular scenario. See Table 15 on page 128 to determine which parameters are used for each scenario.

Overview and Topology

The VXLAN overlay network topology shown in Figure 10 on page 128 includes physical servers A, B, and C on which applications directly run. The applications on physical servers A and B need to communicate with the applications on physical server C. These servers are on the same subnet, so the communication between the applications occurs at the Layer 2 level, and VXLAN encapsulation or tunnels are used to transport their data packets over a Layer 3 network.

Figure 10: Using Overlay Ping and Traceroute to Troubleshoot a VXLAN Overlay Network



In this topology, there are two QFX5100 switches that function as VTEPs. VTEP1 initiates and terminates VXLAN tunnels for physical servers A and B, and VTEP2 does the same for physical server C. VTEP1 and VTEP2 are in VXLAN 100.

A data packet sent from physical server A is typically routed to the Layer 3 router with the IP address of 192.0.2.30 to reach physical server C.

In this VXLAN overlay network topology, a communication issue arises between physical servers A and C. To troubleshoot the issue with this data flow, you can initiate the `ping overlay` and `traceroute overlay` commands on VTEP1 (the source VTEP or `tunnel-src`) and specify that VTEP2 is the destination VTEP or `tunnel-dst`.

The `ping overlay` and `traceroute overlay` commands include several parameters. Table 15 on page 128 explains the purpose and provides a value for each of the parameters used in scenarios 1, 2, and 3.

Table 15 on page 128 does not include all available `ping overlay` and `traceroute overlay` parameters. This example uses the default values of these omitted parameters.

Table 15: Ping and Traceroute Overlay Parameter Values For Scenarios 1, 2, and 3

`ping overlay` and `traceroute overlay` Parameters	Description	Scenario to Which Parameter Applies	Value
`tunnel-type`	Identifies type of tunnel that you are troubleshooting.	All	vxlan

Table 15: Ping and Traceroute Overlay Parameter Values For Scenarios 1, 2, and 3 *(Continued)*

`ping overlay` and `traceroute overlay` Parameters	Description	Scenario to Which Parameter Applies	Value
`vni`	VXLAN network identifier (VNI) of VXLAN used in this example.	All	100
`tunnel-src`	IP address of VTEP1, on which you initiate overlay ping or traceroute.	All	192.0.2.10
`tunnel-dst`	IP address of VTEP2, which receives the overlay ping or traceroute packets.	All	192.0.2.20
`mac`	MAC address of physical server C, which is the destination endpoint.	Scenarios 2 and 3 only	00:00:5E:00:53:cc
`count`	Number of overlay ping requests that VTEP1 sends. **NOTE**: The count parameter does not apply to overlay traceroute.	All	5
`hash-source-mac`	MAC address of physical server A, which is the source endpoint.	Scenario 3 only	00:00:5E:00:53:aa

Table 15: Ping and Traceroute Overlay Parameter Values For Scenarios 1, 2, and 3 *(Continued)*

`ping overlay` and `traceroute overlay` Parameters	Description	Scenario to Which Parameter Applies	Value
`hash-destination-mac`	MAC address of physical server C, which is the destination endpoint. **NOTE**: When specifying this parameter for scenario 3, the MAC address must be the same MAC address as specified for the `mac` parameter.	Scenario 3 only	00:00:5E:00:53:cc
`hash-source-address`	IP address of physical server A.	Scenario 3 only	198.51.100.1
`hash-destination-address`	IP address of physical server C.	Scenario 3 only	198.51.100.3
`hash-vlan`	VLAN ID of source endpoint. **NOTE**: If the source endpoint is not a member of a VLAN, you do not need to use this parameter.	Scenario 3 only	150
`hash-input-interface`	VTEP1 interface on which data flow originates.	Scenario 3 only	xe-0/0/2
`hash-protocol`	A value for the protocol used in the data flow.	Scenario 3 only	17
`hash-source-port`	A value for the outer TCP/UDP source port.	Scenario 3 only	4456

Table 15: Ping and Traceroute Overlay Parameter Values For Scenarios 1, 2, and 3 *(Continued)*

`ping overlay` and `traceroute overlay` Parameters	Description	Scenario to Which Parameter Applies	Value
`hash-destination-port`	A value for the outer UDP destination port.	Scenario 3 only	4540

Table 15 on page 128 includes several hash parameters, which are used for scenario 3. For each of these parameters, you must specify a value associated with the data flow that you are troubleshooting. Based on the values that you specify, the system calculates a VXLAN UDP header source port hash, which is included in the VXLAN UDP header of the overlay ping and traceroute packets. Including the calculated hash in the VXLAN UDP header enables the overlay ping and traceroute packets to emulate data packets in the flow that you are troubleshooting.

BEST PRACTICE: When using the hash parameters, we recommend that you specify a value for each parameter. The exception to this guideline is the hash-vlan parameter, which you do not have to use if the source endpoint is not a member of a VLAN. This practice ensures that the overlay ping and traceroute processes are successful and that the output for each command is accurate. If you do not specify a value for one or more of the hash parameters, the system sends an OAM request that might include incorrect hash values and generates a warning message.

Verification

IN THIS SECTION

This section includes the following verification tasks:

Scenario 1: Verifying That VXLAN 100 Is Configured on VTEP2

Purpose

Verify that a VXLAN with the VNI of 100 is configured on VTEP2. You can use either overlay ping or traceroute to perform this verification.

Action

Overlay Ping

On VTEP1, initiate an overlay ping:

```
user@switch> ping overlay tunnel-type vxlan vni 100 tunnel-src 192.0.2.10 tunnel-dst 192.0.2.20
count 5
ping-overlay protocol vxlan

        vni 100
        tunnel src ip 192.0.2.10
        tunnel dst ip 192.0.2.20
        mac address 00:00:00:00:00:00
        count 5
        ttl 255

            WARNING: following hash-parameters are missing -
                hash computation may not succeed

                end-host smac
                end-host dmac
                end-host src ip
                end-host dst ip
                end-host vlan
                end-host input interface
                end-host protocol
                end-host l4-src-port
                end-host l4-dst-port

Request for seq 1, to 192.0.2.20, at 09-24 22:03:16 PDT.033 msecs

Response for seq 1, from 192.0.2.20, at 09-24 22:03:16 PDT.036 msecs, rtt 10 msecs

  Overlay-segment  not present at RVTEP 192.0.2.20
```

```
Request for seq 2, to 192.0.2.20, at 09-24 22:03:16 PDT.044 msecs

Response for seq 2, from 192.0.2.20, at 09-24 22:03:16 PDT.046 msecs, rtt 10 msecs

  Overlay-segment  not present at RVTEP 192.0.2.20


Request for seq 3, to 192.0.2.20, at 09-24 22:03:16 PDT.054 msecs

Response for seq 3, from 192.0.2.20, at 09-24 22:03:16 PDT.057 msecs, rtt 10 msecs

  Overlay-segment  not present at RVTEP 192.0.2.20


Request for seq 4, to 192.0.2.20, at 09-24 22:03:16 PDT.065 msecs

Response for seq 4, from 192.0.2.20, at 09-24 22:03:16 PDT.069 msecs, rtt 10 msecs

  Overlay-segment  not present at RVTEP 192.0.2.20

Request for seq 5, to 192.0.2.20, at 09-24 22:03:16 PDT.076 msecs

Response for seq 5, from 192.0.2.20, at 09-24 22:03:16 PDT.079 msecs, rtt 10 msecs

  Overlay-segment  not present at RVTEP 192.0.2.20
```

Overlay Traceroute

On VTEP1, initiate an overlay traceroute:

```
user@switch> traceroute overlay tunnel-type vxlan vni 100 tunnel-src 192.0.2.10 tunnel-dst
192.0.2.20
traceroute-overlay protocol vxlan

        vni 100
        tunnel src ip 192.0.2.10
        tunnel dst ip 192.0.2.20
        mac address 00:00:00:00:00:00
        ttl 255
```

```
            WARNING: following hash-parameters are missing -
                hash computation may not succeed

                end-host smac
                end-host dmac
                end-host src ip
                end-host dst ip
                end-host vlan
                end-host input interface
                end-host protocol
                end-host l4-src-port
                end-host l4-dst-port

ttl  Address    Sender Timestamp               Receiver Timestamp             Response Time
  1   10.1.0.1   09-25 00:51:10 PDT.599 msecs            *                     10 msecs
  2   192.0.2.20   09-25 00:51:10 PDT.621 msecs   09-25 00:51:10 PDT.635 msecs   21 msecs

  Overlay-segment not present at RVTEP 192.0.2.20
```

Meaning

The sample overlay ping output indicates the following:

- VTEP1 sent five ping requests to VTEP2, and VTEP2 responded to each request.
- VTEP2 indicated that the VNI of 100 is not configured (`Overlay-segment not present at RVTEP 192.0.2.20`) and included this information in its response to VTEP1.

The sample overlay traceroute output indicates the following:

- Upon receiving an overlay traceroute packet with a time-to-live (TTL) value of 1 hop, the Layer 3 router responds to VTEP1.
- Upon receiving an overlay traceroute packet with a TTL value of 2 hops, VTEP2 responds to VTEP1.
- VTEP2 indicated that the VNI of 100 is not configured (`Overlay-segment not present at RVTEP 192.0.2.20`) and included this information in its response to VTEP1.

NOTE: The asterisk (*) in the Receiver Timestamp column of the overlay traceroute output indicates that the Layer 3 router that received the overlay traceroute packet is not a Juniper Networks device or is a Juniper Networks device that does not support overlay traceroute.

Given that the output of both overlay ping and traceroute indicates that VXLAN 100 is not present, check for this configuration on VTEP2. If you must configure a VNI of 100 on VTEP2, use the `vni` configuration statement at the `[edit vlans vlan-id vxlan]` hierarchy level, and reissue the `ping overlay` or `traceroute overlay` command to verify that VXLAN 100 is now recognized.

Scenario 2: Verifying That the MAC Address of the Destination Endpoint Is on VTEP2

Purpose

Verify that the MAC address (00:00:5E:00:53:cc) of physical server C, which is the destination endpoint, is in the forwarding table of VTEP2. You can use either overlay ping or traceroute to perform this verification.

Action

Overlay Ping

On VTEP1, initiate an overlay ping:

```
user@switch> ping overlay tunnel-type vxlan vni 100 tunnel-src 192.0.2.10 tunnel-dst 192.0.2.20
mac 00:00:5E:00:53:cc count 5
ping-overlay protocol vxlan

        vni 100
        tunnel src ip 192.0.2.10
        tunnel dst ip 192.0.2.20
        mac address 00:00:5E:00:53:cc
        count 5
        ttl 255

            WARNING: following hash-parameters are missing -
                hash computation may not succeed

                end-host smac
                end-host dmac
                end-host src ip
                end-host dst ip
                end-host vlan
                end-host input interface
                end-host protocol
                end-host l4-src-port
                end-host l4-dst-port
```

```
Request for seq 1, to 192.0.2.20, at 09-24 23:53:54 PDT.089 msecs

Response for seq 1, from 192.0.2.20, at 09-24 23:53:54 PDT.089 msecs, rtt 6 msecs

  Overlay-segment present at RVTEP 192.0.2.20

       End-System Not Present


Request for seq 2, to 192.0.2.20, at 09-24 23:53:54 PDT.096 msecs

Response for seq 2, from 192.0.2.20, at 09-24 23:53:54 PDT.100 msecs, rtt 10 msecs

  Overlay-segment present at RVTEP 192.0.2.20

       End-System Not Present


Request for seq 3, to 192.0.2.20, at 09-24 23:53:54 PDT.107 msecs

Response for seq 3, from 192.0.2.20, at 09-24 23:53:54 PDT.111 msecs, rtt 10 msecs

  Overlay-segment present at RVTEP 192.0.2.20

       End-System Not Present

Request for seq 4, to 192.0.2.20, at 09-24 23:53:54 PDT.118 msecs

Response for seq 4, from 192.0.2.20, at 09-24 23:53:54 PDT.122 msecs, rtt 11 msecs

  Overlay-segment present at RVTEP 192.0.2.20

       End-System Not Present


Request for seq 5, to 192.0.2.20, at 09-24 23:53:54 PDT.129 msecs

Response for seq 5, from 192.0.2.20, at 09-24 23:53:54 PDT.133 msecs, rtt 10 msecs

  Overlay-segment present at RVTEP 192.0.2.20
```

```
      End-System Not Present
```

Overlay Traceroute

On VTEP1, initiate an overlay traceroute:

```
user@switch> traceroute overlay tunnel-type vxlan vni 100 tunnel-src 192.0.2.10 tunnel-dst
192.0.2.20 mac 00:00:5E:00:53:cc
traceroute-overlay protocol vxlan

        vni 100
        tunnel src ip 192.0.2.10
        tunnel dst ip 192.0.2.20
        mac address 00:00:5E:00:53:cc
        ttl 255


            WARNING: following hash-parameters are missing -
                hash computation may not succeed

                end-host smac
                end-host dmac
                end-host src ip
                end-host dst ip
                end-host vlan
                end-host input interface
                end-host protocol
                end-host l4-src-port
                end-host l4-dst-port

ttl  Address    Sender Timestamp               Receiver Timestamp             Response Time
  1   10.1.0.1   09-25 00:56:17 PDT.663 msecs            *                     10 msecs
  2   192.0.2.20   09-25 00:56:17 PDT.684 msecs   09-25 00:56:17 PDT.689 msecs   11 msecs

  Overlay-segment present at RVTEP 192.0.2.20

      End-System Not Present
```

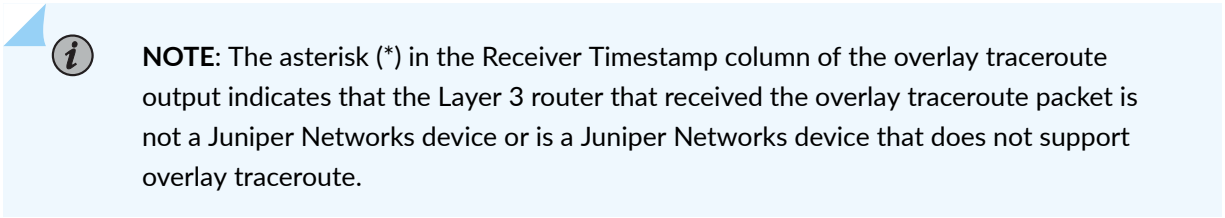
Meaning

The sample overlay ping output indicates the following:

- VTEP1 sent five ping requests to VTEP2, and VTEP2 responded to each request.
- VTEP2 verified that the VNI of 100 is configured (`Overlay-segment present at RVTEP 192.0.2.20`) but that the MAC address of physical server C is not in the forwarding table (`End-System Not Present`). VTEP2 included this information in its response to VTEP1.

The sample overlay traceroute output indicates the following:

- Upon receiving an overlay traceroute packet with a TTL value of 1 hop, the Layer 3 router responds to VTEP1.
- Upon receiving an overlay traceroute packet with a TTL value of 2 hops, VTEP2 responds to VTEP1.
- VTEP2 verified that the VNI of 100 is configured (`Overlay-segment present at RVTEP 192.0.2.20`) but that the MAC address of physical server C is not in the forwarding table (`End-System Not Present`). VTEP2 included this information in its response to VTEP1.



NOTE: The asterisk (*) in the Receiver Timestamp column of the overlay traceroute output indicates that the Layer 3 router that received the overlay traceroute packet is not a Juniper Networks device or is a Juniper Networks device that does not support overlay traceroute.

Given that the output of both overlay ping and traceroute indicates that the MAC address of physical server C is not known by VTEP2, you must further investigate to determine why this MAC address is not in the forwarding table of VTEP2.

Scenario 3: Verifying a Data Flow

Purpose

Verify that there are no issues that might impede the flow of data from physical server A to physical server C. The networking devices that support this flow include VTEP1, the Layer 3 router with the IP address of 192.0.2.30, and VTEP2 (see Figure 10 on page 128).

Initially, use overlay ping, and if the overlay ping results indicate an issue, then use overlay traceroute to determine in which segment of the path the issue exists.

With both overlay ping and traceroute, use the hash parameters to specify information about the devices in this data flow so that the system can calculate a VXLAN UDP header source port hash, which is included in the VXLAN UDP header of the overlay ping and traceroute packets. With the calculated hash included in the VXLAN UDP header, the overlay ping and traceroute packets can emulate data packets in this flow, which should produce more accurate ping and traceroute results.

BEST PRACTICE: When using the hash parameters, we recommend specifying a value for each parameter. The exception to this guideline is the hash-vlan parameter, which you do not have to use if the source endpoint is not a member of a VLAN. This practice ensures that the overlay ping and traceroute processes are successful and that the output for each command is accurate. If you do not specify a value for one or more of the hash parameters, the system sends an OAM request that might include incorrect hash values and generates a warning message.

Action

Overlay Ping

On VTEP1, initiate an overlay ping:

```
user@switch> ping overlay tunnel-type vxlan vni 100 tunnel-src 192.0.2.10 tunnel-dst 192.0.2.20
mac 00:00:5E:00:53:cc count 5 hash-source-mac 00:00:5E:00:53:aa hash-destination-mac
00:00:5E:00:53:cc hash-source-address 198.51.100.1 hash-destination-address 198.51.100.3 hash-
vlan 150 hash-input-interface xe-0/0/2 hash-protocol 17 hash-source-port 4456 hash-destination-
port 4540
ping-overlay protocol vxlan

        vni 100
        tunnel src ip 192.0.2.10
        tunnel dst ip 192.0.2.20
        mac address 00:00:5E:00:53:cc
        count 5
        ttl 255

        hash-parameters:
                input-ifd-idx 653
                end-host smac 00:00:5E:00:53:aa
                end-host dmac 00:00:5E:00:53:cc
                end-host src ip 198.51.100.1
                end-host dst ip 198.51.100.3
                end-host protocol 17
                end-host l4-src-port 4456
                end-host l4-dst-port 4540
                end-host vlan 150

Request for seq 1, to 192.0.2.20, at 09-24 19:15:33 PDT.352 msecs
```

```
Request for seq 2, to 192.0.2.20, at 09-24 19:15:33 PDT.363 msecs

Request for seq 3, to 192.0.2.20, at 09-24 19:15:33 PDT.374 msecs

Request for seq 4, to 192.0.2.20, at 09-24 19:15:33 PDT.385 msecs

Request for seq 5, to 192.0.2.20, at 09-24 19:15:33 PDT.396 msecs
```

Overlay Traceroute

If needed, on VTEP1, initiate an overlay traceroute:

```
user@switch> traceroute overlay tunnel-type vxlan vni 100 tunnel-src 192.0.2.10 tunnel-dst
192.0.2.20 mac 00:00:5E:00:53:cc hash-source-mac 00:00:5E:00:53:aa hash-destination-mac
00:00:5E:00:53:cc hash-source-address 198.51.100.1 hash-destination-address 198.51.100.3 hash-
vlan 150 hash-input-interface xe-0/0/2 hash-protocol 17 hash-source-port 4456 hash-destination-
port 4540
traceroute-overlay protocol vxlan

        vni 100
        tunnel src ip 192.0.2.10
        tunnel dst ip 192.0.2.20
        mac address 00:00:5E:00:53:cc
        ttl 255

        hash-parameters:
                input-ifd-idx 653
                end-host smac 00:00:5E:00:53:aa
                end-host dmac 00:00:5E:00:53:cc
                end-host src ip 198.51.100.1
                end-host dst ip 198.51.100.3
                end-host protocol 17
                end-host l4-src-port 4456
                end-host l4-dst-port 4540
                end-host vlan 150

ttl  Address    Sender Timestamp               Receiver Timestamp             Response Time
  1   10.1.0.1   09-25 00:56:17 PDT.663 msecs            *                     10 msecs
```

Meaning

The sample overlay ping output indicates that VTEP1 sent five ping requests to VTEP2, but VTEP2 did not respond to any of the requests. The lack of response from VTEP2 indicates that a connectivity issue exists along the path between VTEP1 and the Layer 3 router or the path between the Layer 3 router and VTEP2.

To further troubleshoot in which path the issue lies, overlay traceroute is used. The sample overlay traceroute output indicates the following:

- Upon receiving an overlay traceroute packet with a TTL value of 1 hop, the Layer 3 router responds to VTEP1, which indicates that the path between VTEP1 and the Layer 3 router is up.
- VTEP2 does not respond to the overlay traceroute packet, which indicates that the path between the Layer 3 router and VTEP2 might be down.

NOTE: The asterisk (*) in the Receiver Timestamp column of the overlay traceroute output indicates that the Layer 3 router that received the overlay traceroute packet is not a Juniper Networks device or is a Juniper Networks device that does not support overlay traceroute.

Given that the overlay traceroute output indicates that there is a connectivity issue between the Layer 3 router and VTEP2, you must further investigate this path segment to determine the source of the issue.

RELATED DOCUMENTATION

Understanding Overlay ping and traceroute Packet Support

ping overlay

traceroute overlay

4

PART

Configuration Statements and Operational Commands

Junos CLI Reference Overview

We've consolidated all Junos CLI commands and configuration statements in one place. Read this guide to learn about the syntax and options that make up the statements and commands. Also understand the contexts in which you'll use these CLI elements in your network configurations and operations.

- Junos CLI Reference

Click the links to access Junos OS and Junos OS Evolved configuration statement and command summary topics.

- Configuration Statements
- Operational Commands