Junos® OS

OVSDB and VXLAN Feature Guide (VMware NSX)

Release
14.1

Modified: 2016-11-02

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

Junos® OS OVSDB and VXLAN Feature Guide (VMware NSX)
14.1

END USER LICENSE AGREEMENT

Table of Contents

List of Figures

List of Tables

About the Documentation

Documentation and Release Notes

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

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

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

Supported Platforms

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

- EX Series
- MX Series
- QFX Series standalone switches

Using the Examples in This Manual

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

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

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

Merging a Full Example

To merge a full example, follow these steps:

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

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

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

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

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

Merging a Snippet

To merge a snippet, follow these steps:

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

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

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

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

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

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

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

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

Documentation Conventions

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

Table 1: Notice Icons

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

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

Table 2: Text and Syntax Conventions

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

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

; (semicolon)	Identifies a leaf statement at a configuration hierarchy level.	

GUI Conventions

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

Convention	Description	Examples
Bold text like this	Represents graphical user interface (GUI) items you click or select.	• In the Logical Interfaces box, select **All Interfaces**. • To cancel the configuration, click **Cancel**.
> (bold right angle bracket)	Separates levels in a hierarchy of menu selections.	In the configuration editor hierarchy, select **Protocols>Ospf**.

Documentation Feedback

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

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

Requesting Technical Support

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

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

Self-Help Online Tools and Resources

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

- Find CSC offerings: http://www.juniper.net/customers/support/
- Search for known bugs: http://www2.juniper.net/kb/
- Find product documentation: http://www.juniper.net/techpubs/
- Find solutions and answer questions using our Knowledge Base: http://kb.juniper.net/
- Download the latest versions of software and review release notes: http://www.juniper.net/customers/csc/software/
- Search technical bulletins for relevant hardware and software notifications: http://kb.juniper.net/InfoCenter/
- Join and participate in the Juniper Networks Community Forum: http://www.juniper.net/company/communities/
- Open a case online in the CSC Case Management tool: http://www.juniper.net/cm/

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

Opening a Case with JTAC

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

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

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

PART 1

Overview

CHAPTER 1

OVSDB and VXLAN Overview

Understanding VXLANs

Supported Platforms EX Series, MX Series, QFX Series standalone switches

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 32K VXLANs, 32K multicast groups, and 8K 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.
  - QFX 5100 switches support 4K VXLANs, 4K multicast groups, and 2K VTEPs.
- 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 don't need to use 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

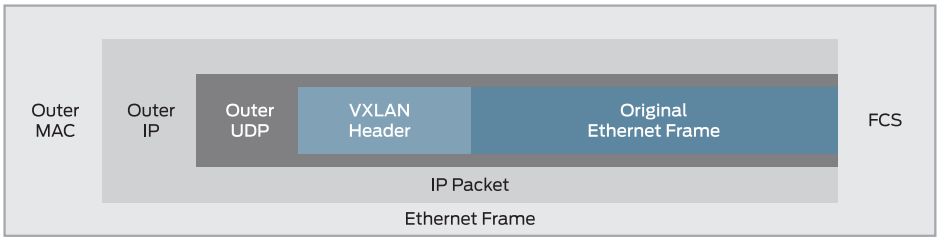
the VXLAN tunnel. To encapsulate an Ethernet frame, VTEPs add a number of fields, including the following:

- Outer 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 1 on page 5 shows the VXLAN packet format.

Figure 1: VXLAN Packet Format



VXLAN Implementation Methods

Junos OS supports implementing VXLANs in the following environments:

- Without SDN controllers. VXLANs implemented in this type of environment are known as manual VXLANs.
- With SDN controllers. 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 Contrail controller) and Juniper Networks devices that support OVSDB can communicate.

Using a QFX5100 Switch with VXLANs

You can configure a QFX5100 switch to perform all of the following roles:

- 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 7 for more information.)
- In an environment with or without an SDN controller, act as a Layer 2 gateway between virtualized and non-virtualized networks in the same data center or between data centers. For example, you can use a QFX5100 switch 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. 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 QFX5100 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.

NOTE: A QFX 5100 switch cannot route traffic between different VXLANs. To connect devices in different VXLANs you need a VXLAN-capable Layer 3 gateway, such as a Juniper Networks MX Series router.

Because the additional headers add 50 to 54 bytes, you might need to increase the MTU on a QFX5100 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 a QFX5100 Switch

Starting with Junos OS 14.1X53-D25, you can configure the UDP port used as the destination port for VXLAN traffic on a QFX5100 switch. To configure the VXLAN destination port to be something other than the default UDP port of 4789, enter

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

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

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

Controlling Transit Multicast Traffic on a QFX5100 Switch

When a QFX5100 switch acting as a VXLAN 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. With Junos OS 14.1X53-D30 and later, if you know that there are no multicast receivers attached to other VTEPs in the VXLAN who 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 or EX9200 Switch as a VTEP

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

- Act as a Layer 2 gateway between virtualized and non-virtualized 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 VPLS tunnels.

NOTE: If you want an MX Series router or EX9200 switch 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 VMware's NSX), 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 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, though 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 the appropriate Layer 2 interfaces. The remote VTEPs also do not copy and send copies of the packets according to the multicast tree.

Load Balancing VXLAN Traffic

On QFX5100 switches, 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.)

Using ping and traceroute With a VXLAN

On a QFX5100 switch, 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.

VXLAN Constraints on QFX5100 Switches

Supported Platforms QFX Series standalone switches

When configuring VXLANs on QFX5100 switches, be aware of the constraints in the following list. In this list, “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.

- You can use VXLANs on a Virtual Chassis or Virtual Chassis Fabric if all of the members are QFX5100 switches. You cannot use VXLANs if any of the members is not a QFX5100 switch.
- VXLAN configuration is supported only in the default routing instance.
- A QFX5100 switch cannot route traffic between different VXLANs.
- 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 a 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.
- Multichassis link aggregation groups (MC-LAGS) are not supported with VXLAN.
- IP fragmentation and defragmentation are not supported on the Layer 3 side.
- The following features are not supported on the Layer 2 side:
  - STP (any variant)
  - IGMP snooping
- 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
- See the following to determine whether ingress node replication is supported on QFX switches:
  - PIM used for control plane: ingress node replication is not supported.
  - Control plane provided by a controller: ingress node replication is not supported.
  - EVPN used with VXLANs: ingress node replication is supported.
- PIM-BIDIR and PIM-SSM are not supported with VXLANs.

- Class of service (CoS) features are not supported with VXLANs.
- If you configure a port-mirroring instance to mirror traffic egressing 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.

Related Documentation

- Examples: Manually Configuring VXLANs on QFX Series Switches on page 81
- Example: Manually Configuring VXLANs on MX Series Routers on page 72
- OVSDB Support on Juniper Networks Devices on page 10
- *mtu*

OVSDB Support on Juniper Networks Devices

Supported Platforms MX Series, QFX Series standalone switches

Table 3 on page 10 lists the Juniper Networks devices that support the Open vSwitch Database (OVSDB) management protocol and the Junos OS releases in which OVSDB is supported. For each device and Junos OS release, the table outlines whether or not the OVSDB software is included in the Junos OS software (**jinstall**) package. If the OVSDB software is not included, the table also includes the name of the separate OVSDB software (**jsdn**) package that must be installed on the device in addition to the Junos OS release.

NOTE: **The separate OVSDB software package release must be the same as the Junos OS release running on the device.**

Table 3: OVSDB Support on Juniper Networks Devices

Juniper Networks Device	Junos OS Release	OVSDB Software Included in Junos OS Software (jinstall) Package?	Separate OVSDB Software (jsdn) Package Name
MX80 3D Universal Edge Routers	14.1R2 and later	No	jsdn-powerpc-*release*
MX240, MX480, MX960 3D Universal Edge Routers	14.1R2 and later	No	jsdn-i386-*release*
QFX5100 Switches, QFX5100 Virtual Chassis, and Virtual Chassis Fabric (VCF)*	14.1X53-D10 through 14.1X53-D27	No	jsdn-i386-*release*
	14.1X53-D30 and later	Yes	–

* OVSDB support with NSX on QFX5100 Virtual Chassis and VCF starts in Junos OS Release 14.1X53-D15. OVSDB support with Contrail on QFX5100 switches, QFX5100 Virtual Chassis, and VCF starts in Junos OS Release 14.1X53-D30.

Related Documentation

- Installing Open vSwitch Database Components on Juniper Networks Devices on page 35

Understanding the Junos OS Implementation of VXLAN and OVSDB in a VMware NSX for Multi-Hypervisor Environment for the Data Center

Supported Platforms MX Series, QFX Series standalone switches

Some Juniper Networks devices support Virtual Extensible LAN (VXLAN) and the Open vSwitch Database (OVSDB) management protocol. (For information about the Juniper Networks devices on which OVSDB is supported and the Junos OS release in which support is introduced, see "OVSDB Support on Juniper Networks Devices" on page 10.) 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 Multi-Hypervisor environment for the data center. Table 4 on page 11 outlines the components that compose this environment and products that are typically deployed for each component.

Table 4: NSX Multi-Hypervisor Components and Products That Can Be Implemented

Component	Products
Cloud management platform (CMP)	CloudStack OpenStack Custom CMP
Network virtualization platform	NSX for Multi-Hypervisor
Hypervisor	Kernel-based Virtual Machine (KVM) Red Hat VMware ESXi Xen NOTE: Juniper Networks supports KVM and ESXi only.
Virtual switch	Open vSwitch (OVS) NSX vSwitch
SDN controller	NSX multi-hypervisor controller NOTE: Juniper Networks supports NSX multi-hypervisor controller version 4.0.3.
Overlay protocol	VXLAN
MAC learning protocol	OVSDB

Figure 2 on page 12 shows a high-level view of the architecture into which the NSX for Multi-Hypervisor platform fits, while Figure 3 on page 12 provides a more detailed representation of the components in the virtual and physical networks.

Figure 2: High-Level NSX for Multi-Hypervisor Architecture

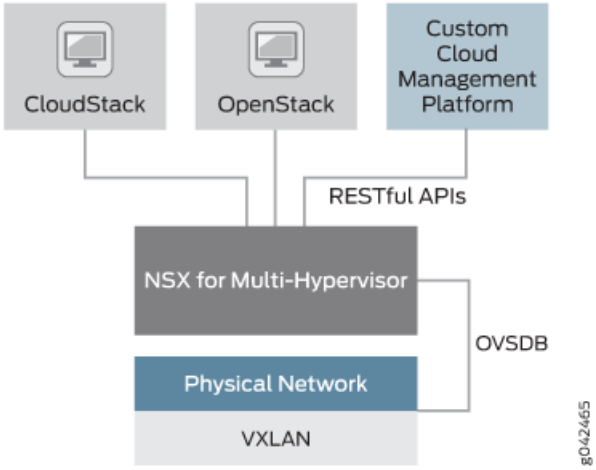
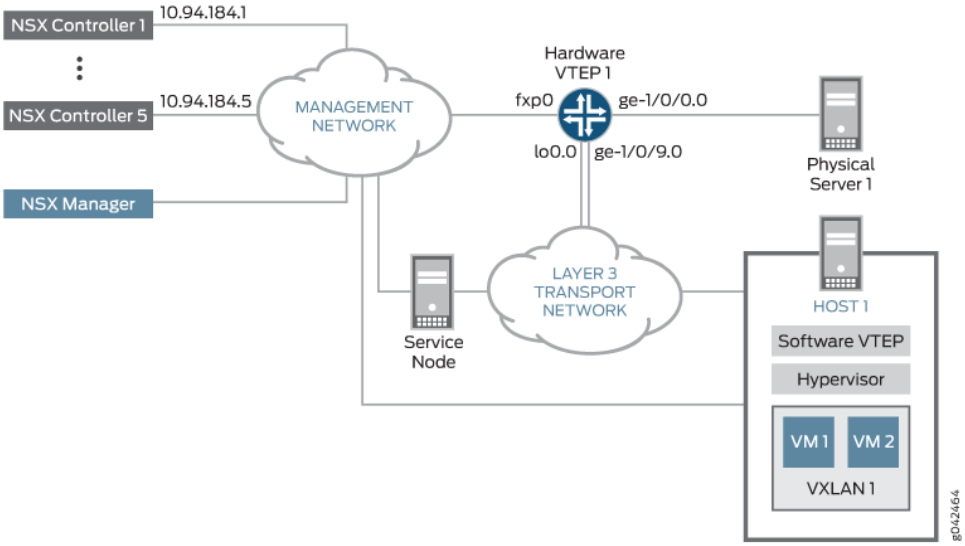


Figure 3: Integration of Juniper Networks Device That Implements VXLAN and OVSDB into NSX for Multi-Hypervisor Environment



In the data center topology shown in Figure 3 on page 12, 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 enables 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 hardware VTEP in Figure 3 on page 12 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 on page 13
- *Open vSwitch Database Schema for Physical Devices*
- *Open vSwitch Database Schema for Physical Devices*

Understanding the OVSDB Protocol Running on Juniper Networks Devices

Supported Platforms MX Series, QFX Series standalone switches

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 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.

Related Documentation

- Understanding How to Set Up Open vSwitch Database Connections Between Juniper Networks Devices and SDN Controllers on page 14

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

Supported Platforms MX Series, QFX Series standalone switches

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" on page 36.

- 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

- Understanding the OVSDB Protocol Running on Juniper Networks Devices on page 13

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

Supported Platforms MX Series, QFX Series standalone switches

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:

- 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.**

- 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

  NOTE: **Only MX Series routers support the Layer 3 multicast traffic scenario.**

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 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 QFX5100 switch.**

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 MX Series router 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 on page 42
- Understanding BFD in a VMware NSX Environment with OVSDB and VXLAN on page 17

Understanding BFD in a VMware NSX Environment with OVSDB and VXLAN

Supported Platforms QFX Series standalone switches

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, the hardware VTEP uses 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. (Juniper Networks switches that support OVSDB and VXLAN are 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.

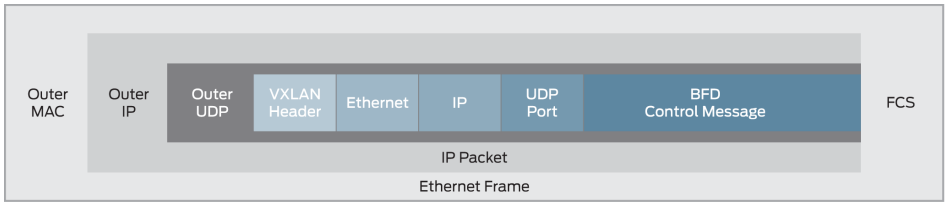
To view the tunnel information collected by the hardware VTEP, you can use the **show ovsdb tunnels** command. This command can help you to determine when there is an issue with the BFD protocol or a replicator.

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 18 shows the VXLAN packet format for BFD control messages.

Figure 4: VXLAN Packet Format for BFD Control Messages



Related Documentation

- Understanding How Layer 2 BUM and Layer 3 Routed Multicast Traffic Are Handled with OVSDB on page 15
- OVSDB Schema for Physical Devices on page 27
- **show ovsdb tunnels on page 163**

Understanding How to Set Up Virtual Extensible LANs in an Open vSwitch Database Environment

Supported Platforms MX Series

The Juniper Networks Junos operating system (Junos OS) implementation of the Open vSwitch Database (OVSDB) management protocol provides a means through which VMware NSX controllers and Juniper Networks devices that support OVSDB can communicate.

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*. In an NSX environment, the same concept is known as a *logical switch*. Understanding the different terminology in turn enables you to better understand the configuration tasks required for setting up OVSDB-managed VXLANs.

The following sections explain what you need to do to set up OVSDB-managed VXLANs properly for Juniper Networks devices that supports OVSDB and VXLAN:

Understanding How to Set Up OVSDB-Managed VXLANs

For each VXLAN that you plan to implement, you must first configure a logical switch, using NSX Manager or the NSX API. Based on the name and the VXLAN network identifier (VNI) that you specify, NSX automatically generates a universally unique identifier (UUID) for the logical switch. You must retain the UUID of the logical switch for later use.

Next, on the Juniper Networks device, you must configure the corresponding VXLAN, including the same VNI specified for the logical switch, using the Junos OS CLI. For the name of the VXLAN, you must specify the UUID for the logical switch.

When configuring a logical switch and a corresponding VXLAN, it is important that the UUID and VNI in both configurations are the same. If these elements are not the same, the logical switch and VXLAN cannot become operational, which means they cannot exchange MAC addresses learned in the NSX and Junos OS environments, respectively.

Table 5 on page 19 provides a summary of the procedure that you must perform for each OVSDB-managed VXLAN on each Juniper Networks device, where to get more information about the configuration task, and the configuration statements that you must use to configure the VXLAN.

Table 5: Summary of Configuration Tasks for Setting Up An OVSDB-Managed VXLAN

Juniper Networks Device That Supports OVSDB and VXLAN	Configure Logical Switch, Using NSX Manager or the NSX API?	Where to Find More Configuration Information	Configure Corresponding VXLAN on Juniper Networks Device?	Junos OS Statement to Configure the OVSDB-Managed VXLAN	Where to Find More Configuration Information
MX Series routers	Yes	See the documentation that accompanies NSX Manager or the NSX API.	Yes	**ovsdb-managed** statement in the **[edit bridge-domains *domain-name* vxlan]** hierarchy. For the name of the VXLAN, specify the UUID for the logical switch configured in NSX Manager or in the NSX API.	"Configuring OVSDB-Managed VXLANs" on page 42

Understanding How to Determine the State of an OVSDB-Managed VXLAN

Regardless of the Juniper Networks device and the procedure that you follow to set up OVSDB-managed VXLANs, after configuring one or more logical switches in NSX Manager or in the NSX API, the NSX controller pushes relevant information to the logical switch table in the OVSDB schema, which resides on the respective devices.

To determine the state of a VXLAN and corresponding logical switch, you can use the **show ovsdb logical-switch** command. The following are possible states:

Created by Controller—A logical switch was configured in NSX Manager or in the NSX API, but the corresponding VXLAN is not yet created on the Juniper Networks device. In this state, the VXLAN and corresponding logical switch are not yet operational.

Created by L2ALD—A VXLAN was created, but the corresponding logical switch is not yet configured in NSX Manager or in the NSX API. In this state, the VXLAN and corresponding logical switch are not yet operational.

Created by both—A logical switch was configured in NSX Manager or in the NSX API, and a corresponding VXLAN was created. In this state, the VXLAN and corresponding logical switch are operational.

Tunnel key mismatch—The VNIs specified in the logical switch and corresponding VXLAN configurations do not match. In this state, the VXLAN and corresponding logical switch are not yet operational.

Related Documentation

- show ovsdb logical-switch on page 154
- Troubleshooting a Nonoperational Logical Switch and Corresponding Junos OS OVSDB-Managed VXLAN on page 113

Understanding Automatically Configured VXLANs in an OVSDB Environment

Supported Platforms QFX Series

NOTE: This topic applies only to QFX5100 switches, which support the automatic 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 automatic configuration.

On all other Juniper Networks devices that support OVSDB and VXLANs, 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" on page 42.

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 automatic configuration of OVSDB-managed VXLANs, including tasks that you must perform before and after the automatic configuration.

The following topics describe the automatic configuration of OVSDB-managed VXLANs:

- Performing Tasks Before and After the Automatic Configuration of OVSDB-Managed VXLANs on page 21
- What the Juniper Networks Switch Actually Creates on page 25

Performing Tasks Before and After the Automatic 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 automatic configuration. Table 6 on page 21 includes a sequentially ordered workflow of tasks and events for the automatic configuration of OVSDB-managed VXLANs in an NSX environment, while Table 7 on page 23 includes the equivalent information for a Contrail environment. Your familiarity with these workflows will ensure that the automatic configuration of OVSDB-managed VXLANs is properly implemented.

In Table 6 on page 21, 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 6 on page 21, the automatic configuration might not work or the automatically configured OVSDB-managed VXLAN might not become functional.

Table 6: Workflow of Tasks and Events for the Automatic 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 automatically 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 6: Workflow of Tasks and Events for the Automatic 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 automatically 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.	–
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" on page 39.	–
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 automatically 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 automatically creates the VXLAN and interface configuration.	For the name of the VXLAN, the Juniper Networks switch uses the UUID of the logical switch.

Table 6: Workflow of Tasks and Events for the Automatic 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 2 to ensure that it displays **Created by both**.	If the output of the **show ovsdb logical-switch** operational mode command displays a state other than **Created by both**, see "Troubleshooting a Nonoperational Logical Switch and Corresponding Junos OS OVSDB-Managed VXLAN" on page 113.

In Table 7 on page 23, 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 7 on page 23, the automatic configuration might not work or the automatically configured OVSDB-managed VXLAN might not become functional.

NOTE: Although you can perform the Contrail configurations outlined in Table 7 on page 23 in the Contrail Web user interface or in the Contrail REST API, Table 7 on page 23 only describes how to perform tasks in the Contrail Web user interface.

Table 7: Workflow of Tasks and Events for the Automatic 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.
2	Enable the Juniper Networks switch to automatically 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.

Table 7: Workflow of Tasks and Events for the Automatic 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
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.	–
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 7: Workflow of Tasks and Events for the Automatic 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 automatically 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 automatically 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 displays a state other than **Created by both**, see "Troubleshooting a Nonoperational Logical Switch and Corresponding Junos OS OVSDB-Managed VXLAN" on page 113.

What the Juniper Networks Switch Actually Creates

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 automatically 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 automatically 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 automatically 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 automatically generates for each interface:

- Automatic Association of a Trunk Interface Supporting Untagged Packets to an Automatically Created VXLAN on page 26
- Automatic Association of a Trunk Interface Supporting Tagged Packets to an Automatically Created VXLAN on page 27

Automatic Association of a Trunk Interface Supporting Untagged Packets to an Automatically 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 automatically configures a trunk interface that can handle untagged packets. (If you specified a valid VLAN ID other than 0, 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 automatically 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 automatically configure a corresponding interface. Also, a system log error message is generated.

Instead of automatically 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 have untagged packets (for example, logical interface ge-1/0/0.0) and

some of which are associated with VXLANs that handle tagged packets (for example, logical interfaces ge-1/0/0.10 and ge-1/0/0.20).

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

Automatic Association of a Trunk Interface Supporting Tagged Packets to an Automatically Created VXLAN

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. This interface accepts only tagged packets from the physical switch.

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 automatically 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.

Related Documentation

- Understanding the OVSDB Protocol Running on Juniper Networks Devices on page 13
- show ovsdb logical-switch on page 154

OVSDB Schema for Physical Devices

Supported Platforms MX Series, QFX Series standalone switches

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 8 on page 28 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 8 on page 28 also indicates when a particular table is not used in the Contrail environment.

Table 8: 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

Table 8: OVSDB Schema Tables *(continued)*

Table Name	Description	Source of Information	Command
Logical switch table	Includes the following information: • Logical switches, which you configured in a VMware NSX environment, or a 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
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 8: OVSDB Schema Tables *(continued)*

Table Name	Description	Source of Information	Command
Multicast MACs local table	NOTE: Only QFX5100 switches support this 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.	Juniper Networks device	show ovsdb mac
Tunnel table	NOTE: Only QFX5100 switches 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

Related Documentation

- Understanding the OVSDB Protocol Running on Juniper Networks Devices on page 13
- Understanding How to Set Up Open vSwitch Database Connections Between Juniper Networks Devices and SDN Controllers on page 14

VXLAN Constraints on QFX5100 Switches

Supported Platforms QFX Series standalone switches

When configuring VXLANs on QFX5100 switches, be aware of the constraints in the following list. In this list, “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.

- You can use VXLANs on a Virtual Chassis or Virtual Chassis Fabric if all of the members are QFX5100 switches. You cannot use VXLANs if any of the members is not a QFX5100 switch.
- VXLAN configuration is supported only in the default routing instance.

- A QFX5100 switch cannot route traffic between different VXLANs.
- 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 a 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.
- Multichassis link aggregation groups (MC-LAGS) are not supported with VXLAN.
- IP fragmentation and defragmentation are not supported on the Layer 3 side.
- The following features are not supported on the Layer 2 side:
  - STP (any variant)
  - IGMP snooping
- 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
- See the following to determine whether ingress node replication is supported on QFX switches:
  - PIM used for control plane: ingress node replication is not supported.
  - Control plane provided by a controller: ingress node replication is not supported.
  - EVPN used with VXLANs: ingress node replication is supported.
- PIM-BIDIR and PIM-SSM are not supported with VXLANs.
- Class of service (CoS) features are not supported with VXLANs.
- If you configure a port-mirroring instance to mirror traffic egressing 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.

Related Documentation

- Understanding VXLANs on page 3
- Examples: Manually Configuring VXLANs on QFX Series Switches on page 81
- Configuring VXLANs on a QFX5100 Switch on page 45

PART 2

Configuration

CHAPTER 2

Configuring OVSDB and VXLANs

Installing OVSDB Software on Juniper Networks Devices

Supported Platforms MX Series, QFX Series standalone switches

NOTE: In Junos OS Release 14.1X53-D30 and later, Open vSwitch Database (OVSDB) software for the QFX5100 switch is included in the Junos OS software (jinstall) package. As a result, you do not need to install a separate OVSDB software (jsdn) package on this switch as was required in previous releases.

You must download the OVSDB software (jsdn) package to the Juniper Networks device and then install the package. The OVSDB software package name uses the following format:

jsdn-*packageID-release*

where:

- *packageID* identifies the package that should run on each Juniper Networks device.
- *release* identifies the OVSDB release; for example, 14.1R2. The OVSDB software release and the Junos OS release running on the device must be the same.

For information about OVSDB support on Juniper Networks devices and the software package for each device, see "OVSDB Support on Juniper Networks Devices" on page 10.

To install the OVSDB software package on a Juniper Networks device:

1. Download the software package to the Juniper Networks device.
2. If an OVSDB software 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 software 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 on page 13

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

Supported Platforms MX Series, QFX Series standalone switches

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 Open vSwitch Database Connections Between Juniper Networks Devices and SDN Controllers on page 14

Setting Up the OVSDB Management Protocol on Juniper Networks Devices

Supported Platforms MX Series, QFX Series standalone switches

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 at least one 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 create an SSL private key and certificate, if they don't 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" on page 36.

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.

NOTE: After completing this procedure, if you have any Juniper Networks device except a QFX5100 switch, you must manually configure OVSDB-managed VXLANs. See "Configuring OVSDB-Managed VXLANs" on page 42.

QFX5100 switches support the automatic configuration of OVSDB-managed VXLANs. On these switches, although the OVSDB-managed VXLAN configuration is automated, there are tasks that you must perform before and after the automatic configuration. See one of the following topics:

- For Junos OS Releases 14.1X53-D15 through 14.1X53-D25, see *Understanding Automatically Configured Virtual Extensible LANs in an Open vSwitch Database Environment*.
- For Junos OS Releases 14.1X53-D26 and later, see "Understanding Automatically Configured VXLANs in an OVSDB Environment" on page 20.

Related Documentation

- Understanding How to Set Up Open vSwitch Database Connections Between Juniper Networks Devices and SDN Controllers on page 14

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

Supported Platforms MX Series, QFX Series standalone switches

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 9 on page 40 provides a summary of key configuration fields in NSX Manager and how to configure them when creating a gateway.

Table 9: 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 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 40.

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 40.
- The OVSDB-managed physical interfaces on the Juniper Networks device. For information about configuring OVSDB-managed interfaces on Juniper Networks devices that support the automatic configuration of VXLANs, see "Setting Up the OVSDB Management Protocol on Juniper Networks Devices" on page 37. For information about configuring OVSDB-managed interfaces on Juniper Networks devices that support the manual configuration of VXLANs, see *Setting Up the Open vSwitch Database Management Protocol on Juniper Networks Devices*.

Table 10 on page 41 provides a summary of key configuration fields in NSX Manager and how to configure them when creating a gateway service.

Table 10: 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 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 40.

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

- A logical switch with which this logical switch 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 40.

Table 11 on page 42 provides a summary of key configuration fields in NSX Manager and how to configure them when creating a logical switch port.

Table 11: 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: In Junos OS Release 14.1X53-D26 and later, 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

- Configuring OVSDB-Managed VXLANs on page 42

Configuring OVSDB-Managed VXLANs

Supported Platforms MX Series

NOTE: This topic does not apply to QFX5100 switches, which support the automatic configuration of OVSDB-managed VXLANs. .

To implement the OVSDB management protocol on a Juniper Networks device, you must configure OVSDB-managed VXLANs.

For 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, you can optionally enable ingress node replication. With this feature enabled, the Juniper Networks device handles the replication of these packets and the forwarding of the replicas to interfaces within the same OVSDB-managed VXLAN. For more information about using ingress node replication or a service node, which is the default way to handle Layer 2 BUM traffic, see "Understanding How Layer 2 BUM and Layer 3 Routed Multicast Traffic Are Handled with OVSDB" on page 15.

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

Before you configure VXLANs on a Juniper Networks device, using the Junos OS CLI:

- For each OVSDB-managed VXLAN that you plan to configure on a Juniper Networks device, you must configure a logical switch in VMware NSX Manager or the NSX API. (For information about configuring a logical switch, see the documentation that accompanies NSX Manager or the NSX API.) Based on the name and VXLAN network identifier (VNI) that you configure for the logical switch, NSX automatically generates a universally unique identifier (UUID) for the logical switch. You must retain the UUID of the logical switch for use when configuring a corresponding VXLAN on the Juniper Networks device as described in the following procedure.
- You must perform the configuration described in "Setting Up the OVSDB Management Protocol on Juniper Networks Devices" on page 37.

To configure an OVSDB-managed VXLAN on a Juniper Networks device:

1. Configure the VXLANs that you want OVSDB to manage. You can configure the VXLANs in the context of a bridge domain, routing instance, or switching instance.

   NOTE: For the name of the bridge domain, you must specify the UUID for the logical switch configured in NSX Manager or the NSX API. If the name of the bridge domain and the UUID are not the same, the logical switch and VXLAN cannot become operational, which means they cannot exchange MAC addresses learned in the NSX and Junos OS environments, respectively.

   Bridge domains:


```
[edit bridge-domains bridge-domain-name vxlan]
user@host# set ovsdb-managed
```

Bridge domains within the specified routing instance:

```
[edit routing-instances routing-instance-name bridge-domains bridge-domain-name
vxlan]
user@host# set ovsdb-managed
```

VLANs within the specified routing instance:

```
[edit routing-instances routing-instance-name vlans vlan-name vxlan]
user@device# set ovsdb-managed
```

Default switching instance within the specified routing instance:

```
[edit routing-instances routing-instance-name switch-options]
user@host# set ovsdb-managed
```

All VXLAN entities within the specified routing instance:

```
[edit routing-instances routing-instance-name vxlan]
user@host# set ovsdb-managed
```

2. (Optional) Enable ingress node replication to handle Layer 2 BUM traffic on interfaces in the same VXLAN in which the traffic originated. You can configure ingress node replication in the context of a bridge domain or routing instance.

   Bridge domains:

   ```
   [edit bridge-domains bridge-domain-name vxlan]
   user@host# set ingress-node-replication
   ```

   Bridge domains or all VXLAN entities, respectively, within the specified routing instance:

   ```
   [edit routing-instances routing-instance-name bridge-domains bridge-domain-name
   vxlan]
   [edit routing-instances routing-instance-name vlans vlan-name vxlan]
   [edit routing-instances routing-instance-name vxlan]
   user@host# set ingress-node-replication
   ```

3. For each Juniper Networks device that you plan to implement as a hardware VTEP, you must perform some configuration tasks in NSX Manager or in the NSX API.

   For more information about the tasks you must perform and key NSX Manager configuration details, see "VMware NSX Configuration for Juniper Networks Devices Functioning as Virtual Tunnel Endpoints" on page 39.

Related Documentation

- Example: Setting Up Inter-VXLAN Routing and OVSDB Connections in a Data Center on page 63

Configuring VXLANs on a QFX5100 Switch

Supported Platforms QFX Series standalone switches

Follow these steps to configure a QFX5100 switch 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 these steps. No special configuration is needed in this case.)

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 and configure it to be used as the tunnel source address:

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

Configuring PIM for VXLANs

If you are not using a 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 forward BUM traffic.

1. Enable PIM on the interface that connects to the Layer 3 network. This is the interface the 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 a QFX5100 switch supports 4K VLANs). 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 on page 3
- Examples: Manually Configuring VXLANs on QFX Series Switches on page 81
- VXLAN Constraints on QFX5100 Switches on page 9

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

Supported Platforms QFX Series standalone switches

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 Junipet Networks device in the physical network.

This example explains how to configure a QFX5100 switch as a hardware VTEP, which serves as a Layer 2 gateway, and set up 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 QFX5100 switch automatically 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 QFX5100 switch running Junos OS Release 14.1X53-D30.
- On the QFX5100 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 QFX5100 switch. See "Creating and Installing an SSL Key and Certificate on a Juniper Networks Device for Connection with SDN Controllers" on page 36.
- 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

Figure 5 on page 48 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 5: VXLAN-OVSDB Layer 2 Gateway Topology

ERROR: Unresolved graphic fileref="g042462.gif" not found in "//cmsxml/default/main/supplemental/STAGING/images/".

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 em0 or em1 of the QFX5100 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 QFX5100 switch. Table 12 on page 48 provides a summary of the entities that must be configured and where they must be configured.

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 a QFX5100 Switch
VXLAN 1	Logical switch for VXLAN 1	VXLAN 1 NOTE: The QFX5100 switch automatically configures this VXLAN.
Physical interface (ge-1/0/0) between physical server 1 and QFX5100 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 (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 QFX5100 switch automatically configures this logical interface.
QFX5100 switch (hardware VTEP 1)	Gateway	–

In NSX Manager, a logical switch 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 QFX5100 switch automatically 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 QFX5100 switch automatically 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 13 on page 49 provides a summary of the VXLAN-OVSDB topology components that are configured on the QFX5100 switch and the configuration settings for each component.

Table 13: 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 physcal interface	Interface name: ge-1/0/0 Native VLAN ID: 4094
	NOTE: The QFX5100 switch automatically 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 type: trunk Member of native VLAN 4094 Associated with VXLAN 28805c1d-0122-495d-85df-19abd647d772
OVSDB-managed VXLAN	NOTE: The QFX5100 switch automatically 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 13: 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

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
```

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

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
```

Step-by-Step Procedure

To configure the QFX5100 switch as a hardware VTEP with an OVSDB connection to an NSX controller:

1. Enable the QFX5100 switch to automatically 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 QFX5100 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 39.

Verification

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 state (**Flags**) of the logical switch 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
```

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 QFX5100 switch automatically 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 113.

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 QFX5100 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 QFX5100 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.

Related Documentation

- Example: Setting Up a VXLAN Layer 2 Gateway and OVSDB Connections in a VMware NSX Environment (Trunk Interfaces Supporting Tagged Packets) on page 54

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

Supported Platforms QFX Series standalone switches

In a physical network, a Juniper Networks device that supports Virtual Extensible LAN (VXLAN) can function as a hardware virtual tunnel endpoint (VTEP), which enables software applications running directly on physical servers to communicate with virtual machines (VMs) in a virtual network.

In this 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 that support OVSDB can communicate. These components serve the following purposes:

- Centralized configuration (QFX5100 switch only)—After you configure a logical switch, using NSX Manager or the NSX API, the NSX controller pushes relevant information about the configuration to the switch (the QFX5100 switch in this example) that functions as a hardware VTEP. Using this information, the switch automatically creates the configuration of a VXLAN, which is the Junos OS equivalent of the logical switch.
- Centralized storage and exchange of MAC route information—Availability of MAC routes enables the hardware VTEP in the physical network and the software VTEP in the virtual network to forward VM traffic between entities in the physical and virtual networks.

This example explains how to configure a QFX5100 switch as a hardware VTEP, which serves as a Layer 2 gateway, and set up this device with an OVSDB connection to an NSX controller.

In this example, two VXLANs are deployed. Given this scenario, the packets exchanged between the applications running on a physical server and VMs in the VXLANs are tagged. As a result, the QFX5100 switch automatically configures logical trunk interfaces for the connection between the physical server and the switch. These logical interfaces handle the tagged packets.

Requirements

This example includes the following hardware and software components:

- A physical server on which software applications directly run.
- A QFX5100 switch running Junos OS Release 14.1X53-D30.
- The topology for this example includes two VXLANs, and as a result, the packets exchanged between the switch and the physical server on switch interface ge-1/0/0 are tagged. Interface ge-1/0/0 is a trunk interface, which accepts tagged packets only.
- 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 in the VXLAN used in this example.
- 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 need to perform the following tasks:

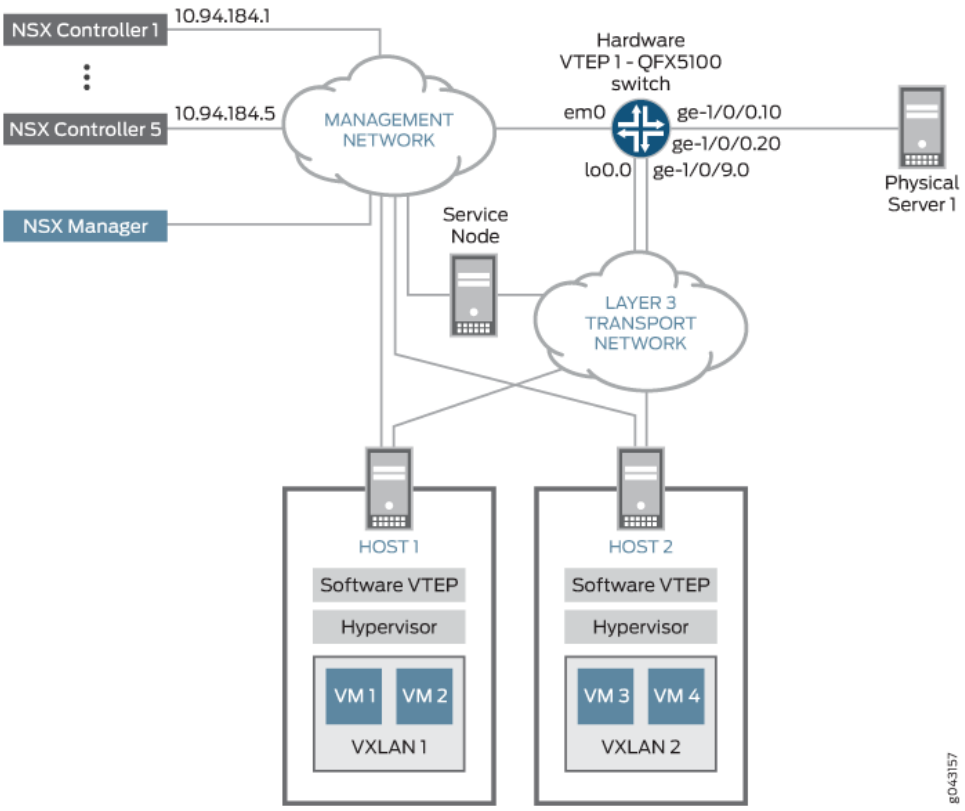
- Using NSX Manager, specify the IP address of the service node.
- Create an SSL private key and certificate, and install them in the **/var/db/certs** directory of the QFX5100 switch. See "Creating and Installing an SSL Key and Certificate on a Juniper Networks Device for Connection with SDN Controllers" on page 36.

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

Overview and Topology

Figure 5 on page 48 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. To enable this communication, a QFX5100 switch is configured as hardware VTEP 1.

Figure 6: VXLAN/OVSDB Layer 2 Gateway Topology



Based on the configuration of the two logical switches in NSX Manager, the QFX5100 switch automatically creates a Junos OS configuration for VXLAN 1 and VXLAN 2. Table 14 on page 56 provides the relevant configuration of the logical switches and the resulting VXLANs that the QFX5100 switch automatically configures.

Table 14: Logical Switch and Corresponding VXLAN Configurations

Relevant Logical Switch Configuration	Automatically Created Junos OS Configuration
For VXLAN 1: UUID: 28805c1d-0122-495d-85df-19abd647d772 VNI: 100 VLAN ID: 10	For VXLAN 1: **set vlans 28805c1d-0122-495d-85df-19abd647d772 vxlan vni 100** **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**

Table 14: Logical Switch and Corresponding VXLAN Configurations *(continued)*

Relevant Logical Switch Configuration	Automatically Created Junos OS Configuration
For VXLAN 2: UUID: 9acc24b3-7b0a-4c2e-b572-3370c3e1acff VNI: 200 VLAN ID: 20	For VXLAN 2: **set vlans 9acc24b3-7b0a-4c2e-b572-3370c3e1acff vxlan vni 200** **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 the name of VXLANs 1 and 2, the switch uses the UUIDs of 28805c1d-0122-495d-85df-19abd647d772 and 9acc24b3-7b0a-4c2e-b572-3370c3e1acff. The QFX5100 switch also uses VNI values that were specified in the logical switch configurations.

Based on the VLAN ID values (10 and 20) specified in the logical switch configurations, the QFX5100 switch configures interface ge-1/0/0 as a trunk interface. The QFX5100 switch also associates logical interface 10 (which is created for VXLAN 1) and logical interface 20 (which is created for VXLAN 2) with their respective VXLANs.

Based on the configuration generated by the switch, the trunk interface accepts packets from VXLAN 1 with a VLAN tag of 10 and from VXLAN 2 with a VLAN tag of 20. 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.

NOTE: **You must also configure interface ge-1/0/0 to be OVSDB-managed by issuing the set protocols ovsdb interfaces ge-1/0/0 command in the Junos OS CLI. The procedure in this example includes the step for doing this.**

The purpose of VXLAN 28805c1d-0122-495d-85df-19abd647d772 and VXLAN 9acc24b3-7b0a-4c2e-b572-3370c3e1acff is to provide a means of mapping physical server 1 to VXLAN 1 using VXLAN 1's VNI of 100 and to VXLAN 2 using VXLAN 2's VNI of 100.

On the management interface em0 of the QFX5100 switch, a connection with an NSX controller is explicitly configured by using the Junos OS CLI.

The components of the topology for this example are shown in Table 13 on page 49.

Table 15: Components of the Topology for Setting Up a VXLAN Layer 2 Gateway and OVSDB Connections

Property	Settings
Junos OS configurations for VXLAN 1 and VXLAN 2	NOTE: The QFX5100 switch automatically creates these VXLAN configurations, which are based on the NSX-equivalent logical switch configurations in NSX Manager. Therefore, no manual configuration is required. However, before the switch can automatically create these configuration, you must enter the **set switch-options ovsdb-managed** configuration mode command in the Junos OS CLI on the switch. For VXLAN 1: VXLAN name: 28805c1d-0122-495d-85df-19abd647d772 VLAN ID: 10 VNI: 100 For VXLAN 2: VXLAN name: VXLAN 9acc24b3-7b0a-4c2e-b572-3370c3e1acff VLAN ID: 20 VNI: 200
OVSDB-managed interface	NOTE: The QFX5100 switch automatically creates these interface configurations, which are based on the NSX-equivalent logical switch configurations in NSX Manager. Therefore, no manual configuration is required. However, after the switch automatically creates these configurations, you must specify interface ge-1/0/0 as an OVSDB-managed interface by entering the **set protocols ovsdb interface ge-1/0/0** configuration mode command in the Junos OS CLI on the switch. Interface name: ge-1/0/0 Interface type: trunk Logical interface 10: associated with VXLAN 28805c1d-0122-495d-85df-19abd647d772 Logical interface 20: associated with VXLAN 9acc24b3-7b0a-4c2e-b572-3370c3e1acff
NSX controller	IP address: 10.94.184.1
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.
Hardware VTEP source identifier	Source interface: loopback (lo0.0) Source IP address: 10.17.17.17/32

Table 15: Components of the Topology for Setting Up a VXLAN Layer 2 Gateway and OVSDB Connections *(continued)*

Property	Settings
OVSDB tracing operations	Filename: /var/log/ovsdb File size: 10 MB Flag: All

Non-OVSDB and Non-VXLAN Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your 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
```

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

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
```

Step-by-Step Procedure

To configure the QFX5100 switch as hardware VTEP 1 with an OVSDB connection to an NSX controller:

1. Enable the QFX5100 switch to automatically configure OVSDB-managed VXLANs and associated interfaces.

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

2. 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 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. See the VMware documentation that accompanies NSX Manager.

8. In NSX Manager, configure a gateway for the QFX5100 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 39.

Verification

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 state (**Flags**) of the logical switches is **Created by both**.

Action Issue the **show ovsdb logical-switch** operational mode 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 QFX5100 switch automatically created 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" on page 113.

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 Issue the **show ovsdb mac remote** operational mode 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 QFX5100 switch, which functions as a hardware VTEP.

Verifying the NSX Controller Connection

Purpose Verify that the connection with the NSX controller is up.

Action Issue the **show ovsdb controller** operational mode 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 QFX5100 switch.

Verifying the OVSDB-Managed Interface

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

Action Issue the **show ovsdb interface** operational mode command, and verify that interface ge-1/0/0 is 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 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 **10**, and VXLAN **9acc24b3-7b0a-4c2e-b572-3370c3e1acff**, which has a **VLAN ID** of **20**.

Related Documentation

- *Example: Setting Up a VXLAN Layer 2 Gateway and OVSDB Connections Between Virtual and Physical Entities in a Data Center (Access Interfaces)*
- Example: Setting Up a VXLAN Layer 2 Gateway and OVSDB Connections in a VMware NSX Environment (Trunk Interfaces Supporting Untagged Packets) on page 46

Example: Setting Up Inter-VXLAN Routing and OVSDB Connections in a Data Center

Supported Platforms MX Series

This example shows a data center in which virtual machines (VMs) in different Virtual Extensible LANs (VXLANs) need to communicate. The Juniper Networks device that is integrated into this environment functions as a hardware virtual tunnel endpoint (VTEP) that can route VM traffic from one VXLAN (Layer 2) environment to another.

The Juniper Networks device implements the Open vSwitch Database (OVSDB) management protocol and has a connection with a VMware NSX controller, both of which enable these two entities to exchange MAC routes to and from VMs in the physical and virtual networks. .

This example explains how to configure a Juniper Networks device as hardware VTEPs and set up OVSDB connections to an NSX controller.

Requirements

In this example, the topology includes the following hardware and software components:

- A cluster of five NSX controllers, each of which is running NSX software version 4.0.3.
- NSX Manager.
- A service node that handles broadcast, unknown unicast, and multicast (BUM) traffic within each of the two VXLANs.

- Two hosts, each of which includes VMs managed by a hypervisor. Each hypervisor includes a software VTEP. The VMs on each of the hosts belong to different VXLANs.
- A Juniper Networks device that routes VM traffic between the two VXLANs. For example, an MX Series router running Junos OS Release 14.1R2 or later. The Juniper Networks device must also run an OVSDB software package, and the release of this package must be the same as the Junos OS release running on the device. This device is configured to function as a hardware VTEP.

Before you start the configuration of the Juniper Networks device, you need to perform the following tasks:

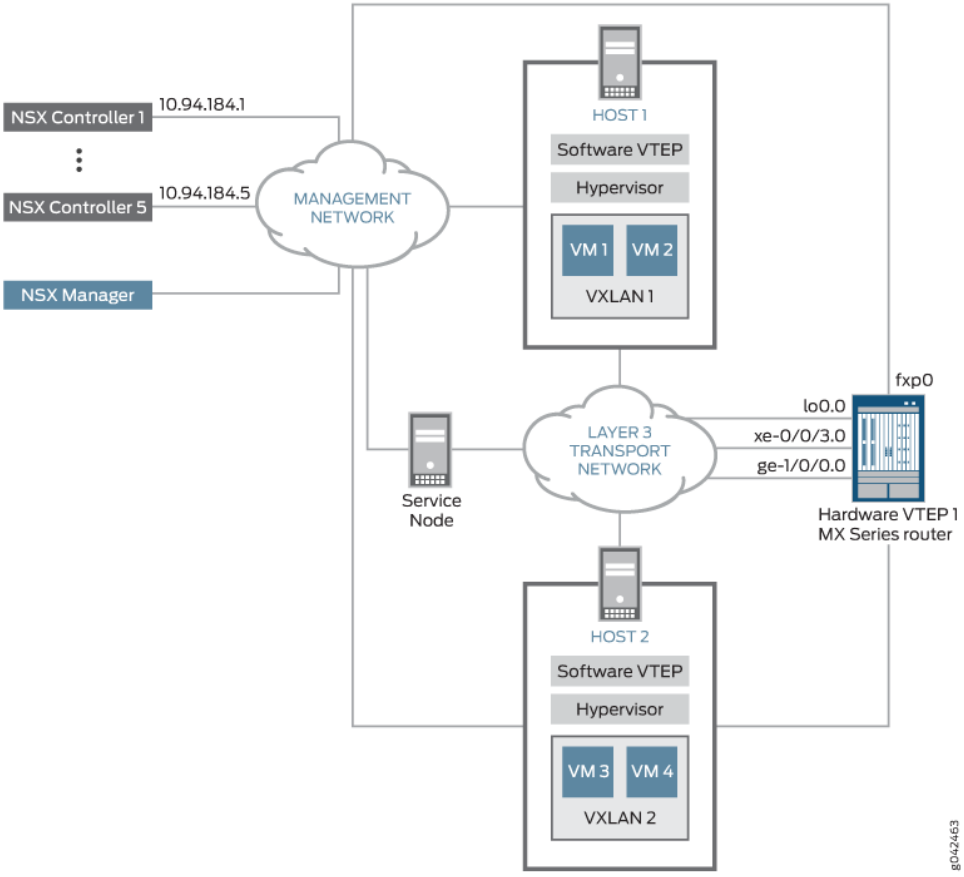
- In NSX Manageror the NSX API, specify the IP address of the service node.
- In NSX Manager or the NSX API, configure a logical switch for each VXLAN that OVSDB will manage. This example implements two OVSDB-managed VXLANs; therefore, you must configure two logical switches. After the configuration of each logical switch, NSX automatically generates a universally unique identifier (UUID) for the logical switch. If you have not already, retrieve the UUID for each logical switch. A sample UUID is 28805c1d-0122-495d-85df-19abd647d772. When configuring the equivalent VXLANs on the Juniper Networks device, you must use the UUID of the logical switch as the bridge domain or VLAN name.
- Create an SSL private key and certificate, and install them in the **/var/db/certs** directory of the Juniper Networks device. For more information, see "Creating and Installing an SSL Key and Certificate on a Juniper Networks Device for Connection with SDN Controllers" on page 36.

For information about using NSX Manager or the NSX API to perform these configuration tasks, see the documentation that accompanies these respective products.

Overview and Topology

In the topology shown in Figure 7 on page 65, VM 1 in VXLAN 1 needs to communicate with VM 3 in VXLAN 2. To enable this communication, hardware VTEP 1, which can be an MX Series router, is configured to route VM traffic between the two VXLANs.

Figure 7: Inter-VXLAN Routing and OVSDB Topology



On hardware VTEP 1, a routing instance (virtual switch) is set up. Within the routing instance, two VXLANs are configured: VXLAN 1 and VXLAN 2. Both of the VXLANs are associated with an integrated routing and bridging (IRB) interface, over which VM traffic is routed between the VXLANs.

On hardware VTEP 1, a connection with an NSX controller is configured on the management interface fxp0. This configuration enables the NSX controller to push MAC routes for VM 1 and VM 3 to the hardware VTEP by way of the table for remote unicast MAC addresses in the OVSDB schema for physical devices.

Each VXLAN-encapsulated packet must include a source IP address, which identifies the source hardware or software VTEP, in the outer IP header. In this example, for hardware VTEP 1, the IP address of the loopback interface (lo0) is used.

Within each of the two VXLANs, a service node replicates Layer 2 BUM packets then forwards the replicas to all interfaces in the VXLANs. Having the service node handle the Layer 2 BUM traffic is the default behavior, and no configuration is required for this Juniper Networks device.

Between the two VXLANs, ingress node replication is automatically implemented and does not need to be configured. With this feature, hardware VTEP 1 replicates a Layer 3 multicast packet, then the IRB interface associated with the VXLAN that originated the packet forwards the replicas to all hardware and software VTEPs, but not to service nodes, in the other OVSDB-managed VXLAN.

In this example, the tracing of all OVSDB events are configured. The output of the OVSDB events are placed in a file named **ovsdb**, which is stored in the **/var/log** directory. By default, a maximum of 10 trace files can exist, and the configured maximum size of each file is 50 MB.

Table 13 on page 49 describes the components of the example topology.

Table 16: Components of the Topology for Setting Up Inter-VXLAN Routing and OVSDB Connections in a Data Center

Property	Settings
Routing instance	Name: vx1 Type: virtual switch OVSDB-managed VXLANs included: VXLAN 1 and VXLAN 2
VXLAN 1	Bridge domain or VLAN associated with: 28805c1d-0122-495d-85df-19abd647d772 Interface: xe-0/0/2.0 VLAN ID: 100 VNI: 100 IRB for inter-VXLAN traffic: irb.0; 10.20.20.1/24
VXLAN 2	Bridge domain or VLAN associated with: 96a382cd-a570-4ac8-a77a-8bb8b16bde70 Interface: xe-1/2/0.0 VLAN ID: 200 VNI: 200 IRB for inter-VXLAN traffic: irb.1; 10.10.10.3/24
Handling of BUM traffic in each VXLAN	Service node NOTE: By default, one or more service nodes handle Layer 2 BUM traffic in a VXLAN; therefore, no configuration is required.
Handling of Layer 3 multicast traffic between VXLANs	Ingress node replication NOTE: On IRB interfaces that forward Layer 3 multicast traffic from one OVSDB-managed VXLAN to another, ingress node replication is automatically implemented; therefore, no configuration is required.

Table 16: Components of the Topology for Setting Up Inter-VXLAN Routing and OVSDB Connections in a Data Center *(continued)*

Property	Settings
Hardware VTEP source identifier	Source interface: loopback (lo0.0) Source IP address: 10.19.19.19/32
NSX controller	IP address: 10.94.184.1
OVSDB tracing operations	Filename: /var/log/ovsdb File size: 50 m Flag: All

Configuration

An MX Series router can function as hardware VTEP 1 in this example.

To configure inter-VXLAN routing and OVSDB connections in a data center topology, you need to perform this task:

- Configuring an MX Series Router as a Hardware VTEP with an OVSDB Connection on page 68

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.

NOTE: After completing this configuration, you must configure a gateway, which is the NSX-equivalent of a hardware VTEP. This example implements one hardware VTEP; therefore, you must configure one gateway, a gateway service, and a logical switch port using NSX Manager or the NSX API. For more information about the tasks you must perform and key NSX Manager configuration details, see "VMware NSX Configuration for Juniper Networks Devices Functioning as Virtual Tunnel Endpoints" on page 39.

MX Series router configuration:

```
set chassis network-services enhanced-ip
set interfaces xe-0/0/3 unit 0 family inet address 10.50.50.2/24
set interfaces ge-1/0/0 unit 0 family inet address 10.100.100.99/24
set routing-options router-id 10.19.19.19
set protocols ospf area 0.0.0.0 interface xe-0/0/3.0
set protocols ospf area 0.0.0.0 interface ge-1/0/0.0
set protocols ospf area 0.0.0.0 interface lo0.0
set interfaces xe-0/0/2 unit 0 family bridge interface-mode access
set interfaces xe-0/0/2 unit 0 family bridge vlan-id 100
```

```
set interfaces xe-1/2/0 unit 0 family bridge interface-mode access
set interfaces xe-1/2/0 unit 0 family bridge vlan-id 200
set interfaces irb unit 0 family inet address 10.20.20.1/24
set interfaces irb unit 1 family inet address 10.10.10.3/24
set routing-instances vx1 vtep-source-interface lo0.0
set routing-instances vx1 instance-type virtual-switch
set routing-instances vx1 interface xe-0/0/2.0
set routing-instances vx1 interface xe-1/2/0.0
set routing-instances vx1 bridge-domains 28805c1d-0122-495d-85df-19abd647d772
  vlan-id 100
set routing-instances vx1 bridge-domains 28805c1d-0122-495d-85df-19abd647d772
  routing-interface irb.0
set routing-instances vx1 bridge-domains 28805c1d-0122-495d-85df-19abd647d772
  vxlan ovsdb-managed
set routing-instances vx1 bridge-domains 28805c1d-0122-495d-85df-19abd647d772
  vxlan vni 100
set routing-instances vx1 bridge-domains 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  vlan-id 200
set routing-instances vx1 bridge-domains 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  routing-interface irb.1
set routing-instances vx1 bridge-domains 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  vxlan ovsdb-managed
set routing-instances vx1 bridge-domains 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  vxlan vni 200
set interfaces lo0 unit 0 family inet address 10.19.19.19/32 primary
set interfaces lo0 unit 0 family inet address 10.19.19.19/32 preferred
set protocols ovsdb traceoptions file ovsdb
set protocols ovsdb traceoptions file size 50m
set protocols ovsdb traceoptions flag all
set protocols ovsdb controller 10.94.184.1
set protocols ovsdb interfaces xe-0/0/2.0
set protocols ovsdb interfaces xe-1/2/0.0
```

Configuring an MX Series Router as a Hardware VTEP with an OVSDB Connection

Step-by-Step Procedure

To configure an MX Series router as hardware VTEP 1 with an OVSDB connection to an NSX controller, follow these steps:

1. Create the Layer 3 network.

```
[edit chassis]
user@router# set network-services enhanced-ip
[edit interfaces]
user@router# set xe-0/0/3 unit 0 family inet address 10.50.50.2/24
user@router# set ge-1/0/0 unit 0 family inet address 10.100.100.99/24
[edit routing-options]
user@router# set router-id 10.19.19.19
[edit protocols]
user@router# set ospf area 0.0.0.0 interface xe-0/0/3.0
user@router# set ospf area 0.0.0.0 interface ge-1/0/0.0
user@router# set ospf area 0.0.0.0 interface lo0.0
```

2. Create an access interface for VXLAN 1, and associate the interface with the VXLAN.

```
[edit interfaces]
user@router# set xe-0/0/2 unit 0 family bridge interface-mode access
```

```
user@router# set xe-0/0/2 unit 0 family bridge vlan-id 100
```

3. Create an access interface for VXLAN 2, and associate the interface with the VXLAN.

```
[edit interfaces]
user@router# set xe-1/2/0 unit 0 family bridge interface-mode access
user@router# set xe-1/2/0 unit 0 family bridge vlan-id 200
```

4. Create an IRB interface to handle inter-VXLAN traffic for VXLAN 1.

```
[edit interfaces]
user@router# set irb unit 0 family inet address 10.20.20.1/24
```

5. Create an IRB interface to handle inter-VXLAN traffic for VXLAN 2.

```
[edit interfaces]
user@router# set irb unit 1 family inet address 10.10.10.3/24
```

6. Set up the virtual switch routing instance.

```
[edit routing-instances]
user@router# set vx1 vtep-source-interface lo0.0
user@router# set vx1 instance-type virtual-switch
user@router# set vx1 interface xe-0/0/2.0
user@router# set vx1 interface xe-1/2/0.0
user@router# set vx1 bridge-domains 28805c1d-0122-495d-85df-19abd647d772
  vlan-id 100
user@router# set vx1 bridge-domains 28805c1d-0122-495d-85df-19abd647d772
  routing-interface irb.0
user@router# set vx1 bridge-domains 28805c1d-0122-495d-85df-19abd647d772
  vxlan ovsdb-managed
user@router# set vx1 bridge-domains 28805c1d-0122-495d-85df-19abd647d772
  vxlan vni 100
user@router# set vx1 bridge-domains 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  vlan-id 200
user@router# set vx1 bridge-domains 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  routing-interface irb.1
user@router# set vx1 bridge-domains 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  vxlan ovsdb-managed
user@router# set vx1 bridge-domains 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  vxlan vni 200
```

7. 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@router# set lo0 unit 0 family inet address 10.19.19.19/32 primary
user@router# set lo0 unit 0 family inet address 10.19.19.19/32 preferred
```

8. Set up OVSDB tracing operations.

```
[edit protocols]
user@router# set ovsdb traceoptions file ovsdb
user@router# set ovsdb traceoptions file size 50m
user@router# set ovsdb traceoptions flag all
```

9. Configure a connection with an NSX controller.

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

10. Configure interfaces xe-0/0/2.0 and xe-1/2/0.0 to be managed by OVSDB.

```
[edit protocols]
user@router# set ovsdb interfaces xe-0/0/2.0
user@router# set ovsdb interfaces xe-1/2/0.0
```

NOTE: After completing this configuration, you must configure a gateway, which is the NSX-equivalent of a hardware VTEP. This example implements one hardware VTEP; therefore, you must configure one gateway, a gateway service, and a logical switch port by using NSX Manager or the NSX API. For more information about the tasks you must perform and key NSX Manager configuration details, see "VMware NSX Configuration for Juniper Networks Devices Functioning as Virtual Tunnel Endpoints" on page 39.

Verification

Verifying the Logical Switches

Purpose Verify that logical switches with the UUIDs of 28805c1d-0122-495d-85df-19abd647d772 and 96a382cd-a570-4ac8-a77a-8bb8b16bde70 are configured in NSX Manager or the NSX API, and that information about the logical switches is published in the OVSDB schema.

Action Issue the **show ovsdb logical-switch** operational mode command.

```
user@host> 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: 96a382cd-a570-4ac8-a77a-8bb8b16bde70
Flags: Created by both
VNI: 200
Num of Remote MAC: 1
Num of Local MAC: 1
```

Meaning The output verifies that information about the logical switches is published in the OVSDB schema. The **Created by both** state indicates that the logical switches are configured in NSX Manager or the NSX API, and the corresponding VXLANs are configured on the Juniper Networks device. 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" on page 113.

Verifying the MAC Addresses of VM 1 and VM 3

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

Action Issue the **show ovsdb mac remote** operational mode command, and verify that the MAC addresses for VM 1 and VM 3 are present.

```
user@host> show ovsdb mac remote
Logical Switch Name: 28805c1d-0122-495d-85df-19abd647d772
  Mac                   IP                 Encapsulation      Vtep
  Address               Address                               Address
  08:33:9d:5f:a7:f1     0.0.0.0            Vxlan over Ipv4    10.19.19.19
Logical Switch Name: 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  Mac                   IP                 Encapsulation      Vtep
  Address               Address                               Address
  a8:59:5e:f6:38:90     0.0.0.0            Vxlan over Ipv4    10.19.19.10
```

Meaning The output shows that the MAC addresses for VM 1 and VM 3 are present and are associated with logical switches with the UUIDs of 28805c1d-0122-495d-85df-19abd647d772 and 96a382cd-a570-4ac8-a77a-8bb8b16bde70, respectively. Given that the MAC addresses are present, VM 1 and VM 3 are reachable through hardware VTEP 1.

Verifying the NSX Controller Connection

Purpose Verify that the connection with the NSX controller is up.

Action Issue the **show ovsdb controller** operational mode command, and verify that the controller connection state is **up**.

```
user@host> 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 device.

Related Documentation

- Understanding How Layer 2 BUM and Layer 3 Routed Multicast Traffic Are Handled with OVSDB on page 15

Example: Manually Configuring VXLANs on MX Series Routers

Supported Platforms MX Series

Virtual Extensible Local Area Network (VXLAN) is a Layer 3 encapsulation protocol that enables MX Series routers to push Layer 2 or Layer 3 packets through a VXLAN tunnel to a virtualized data center or the Internet. Communication is established between two virtual tunnel endpoints (VTEPs). VTEPs encapsulate the virtual machine traffic into a VXLAN header and strip off the encapsulation.

This example shows how to configure VXLAN on MX Series routers using switch options in a default bridge domain.

Requirements

This example uses the following hardware and software components:

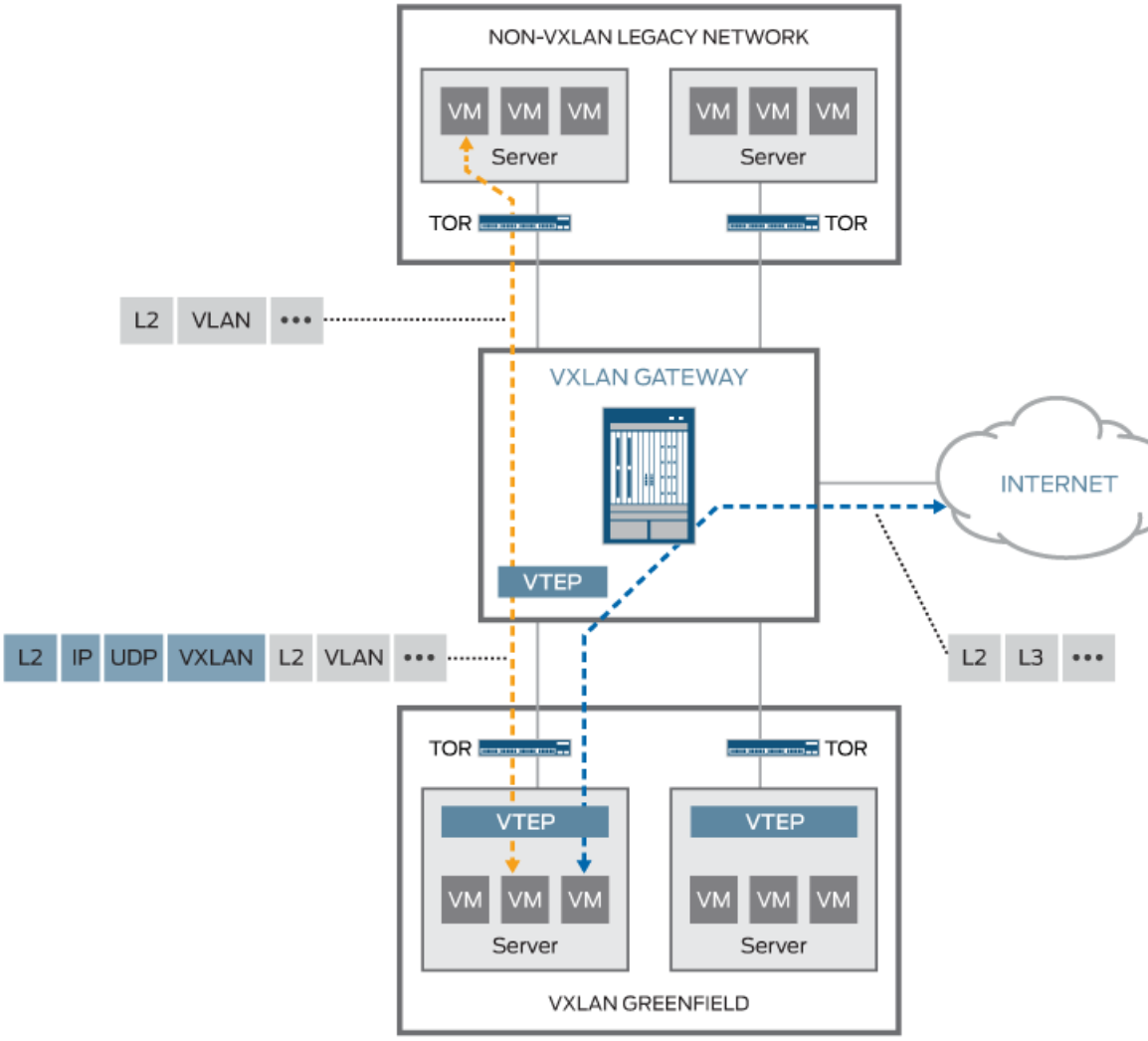
- An MX Series router
- A VXLAN capable peer router
- Junos OS Release 14.1

Overview

In this example, VXLAN is configured to run on a default bridge domain. VTEP interfaces sources are configured to the loopback address, and VLAN groups are configured under bridge domains with VXLAN enabled. Interfaces are configured for VLAN tagging and encapsulation, and IRB is enabled. OSPF and PIM protocols are configured to facilitate unicast and multicast routing. The chassis is configured for GRES and enhanced IP services.

Topology

Figure 1: VXLAN Topology



Configuring VXLAN on MX Series Routers

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 switch-options vtep-source-interface lo0.0
set bridge-domains vlan-5 vxlan vni 100
set bridge-domains vlan-5 vxlan multicast-group 239.1.1.1
set bridge-domains vlan-5 vlan-id 100
set bridge-domains vlan-5 routing-interface irb.0
set bridge-domains vlan-5 interface xe-1/0/0.0
```

```
set bridge-domains vlan-6 vxlan vni 200
set bridge-domains vlan-6 vxlan multicast-group 239.1.1.1
set bridge-domains vlan-6 vlan-id 200
set bridge-domains vlan-6 routing-interface irb.1
set bridge-domains vlan-6 interface xe-2/0/0.0
set interfaces xe-1/0/0 vlan-tagging
set interfaces xe-1/0/0 encapsulation flexible-ethernet-services
set interfaces xe-1/0/0 unit 0 encapsulation vlan-bridge
set interfaces xe-1/0/0 unit 0 vlan-id 100
set interfaces xe-2/0/0 vlan-tagging
set interfaces xe-2/0/0 encapsulation flexible-ethernet-services
set interfaces xe-2/0/0 unit 0 encapsulation vlan-bridge
set interfaces xe-2/0/0 unit 0 vlan-id 200
set interface irb unit 0 family inet address 5.5.5.1/24
set interface irb unit 1 family inet address 6.6.6.1/24
set interfaces lo0 unit 0 family inet address 3.3.3.3/32
set protocols ospf area 0.0.0.0 interface ge-8/3/8.0
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols ospf area 0.0.0.0 interface xe-0/1/3.0
set protocols ospf area 0.0.0.0 interface ge-8/3/2.0
set protocols pim rp static address 10.2.1.3
set protocols pim interface lo0.0 mode bidirectional-sparse
set protocols pim interface ge-8/3/8.0 mode bidirectional-sparse
set protocols pim interface xe-0/1/3.0 mode bidirectional-sparse
set protocols pim interface ge-8/3/2.0 mode bidirectional-sparse
set chassis redundancy graceful-switchover
set chassis aggregated-devices ethernet device-count 10
set chassis fpc 1 pic 0 tunnel-services bandwidth 10g
set chassis network-services enhanced-ip
```

Configuring VXLAN

Step-by-Step Procedure

The following example show how to set up a basic VXLAN configuration with default bridge domains and switch options. To configure VXLAN on an MX Series router, follow these steps:

1. Configure VTEP interface sources under **switch-options** for the default-switch.

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

2. Set up a VLAN group named **vlan-5** and set its VXLAN Network Identifier (VNI) to 100.

   ```
   [edit]
   user@router# set bridge-domains vlan-5 vxlan vni 100
   ```

3. Configure the **vlan-5** multicast group address for VXLAN.

   ```
   [edit]
   user@router# set bridge-domains vlan-5 vxlan multicast-group 239.1.1.1
   ```

4. Set the VLAN ID to 100 for **vlan-5**.

   ```
   [edit]
   user@router# set bridge-domains vlan-5 vlan-id 100
   ```

5. Configure integrated bridging and routing (IRB) for **vlan-5**.

```
[edit]
user@router# set bridge-domains vlan-5 routing-interface irb.0
```

6. Assign the xe-1/0/0.0 interface to **vlan-5**.

```
[edit]
user@router# set bridge-domains vlan-5 interface xe-1/0/0.0
```

7. Set up a VLAN group named **vlan-6** and set its VXLAN Network Identifier (VNI) to 200.

```
[edit]
user@router# set bridge-domains vlan-6 vxlan vni 200
```

8. Configure the **vlan-6** multicast group address for VXLAN.

```
[edit]
user@router# set bridge-domains vlan-6 vxlan multicast-group 239.1.1.1
```

9. Set the VLAN ID to 100 for **vlan-6**.

```
[edit]
user@router# set bridge-domains vlan-6 vlan-id 200
```

10. Configure IRB for **vlan-6**.

```
[edit]
user@router# set bridge-domains vlan-6 routing-interface irb.1
```

11. Assign the xe-2/0/0.0 interface to **vlan-6**.

```
[edit]
user@router# set bridge-domains vlan-6 interface xe-2/0/0.0
```

12. Set up VLAN tagging for xe-1/0/0.

```
[edit]
user@router# set interfaces xe-1/0/0 vlan-tagging
```

13. Configure flexible Ethernet service encapsulation on xe-1/0/0.

```
[edit]
user@router# set interfaces xe-1/0/0 encapsulation flexible-ethernet-services
```

14. Set up VLAN bridging encapsulation for xe-1/0/0 unit 0 ˙.

```
[edit]
user@router# set interfaces xe-1/0/0 unit 0 encapsulation vlan-bridge
```

15. Set the xe-1/0/0 unit 0 VLAN ID to 100.

```
[edit]
user@router# set interfaces xe-1/0/0 unit 0 vlan-id 100
```

16. Configure VLAN tagging for xe-2/0/0

```
[edit]
user@router# set interfaces xe-2/0/0 vlan-tagging
```


17. Set up flexible Ethernet service encapsulation on xe-2/0/0.

```
[edit]
user@router# set interfaces xe-2/0/0 encapsulation flexible-ethernet-services
```

18. Configure VLAN bridging encapsulation for xe-2/0/0 unit 0 .

```
[edit]
user@router# set interfaces xe-2/0/0 unit 0 encapsulation vlan-bridge
```

19. Set the xe-2/0/0 unit 0 VLAN ID to 200.

```
[edit]
user@router# set interfaces xe-2/0/0 unit 0 vlan-id 200
```

20. Configure the IRB unit 0 family inet address.

```
[edit]
user@router# set interface irb unit 0 family inet address 5.5.5.1/24
```

21. Configure the IRB unit 1 family inet address.

```
[edit]
user@router# set interface irb unit 1 family inet address 6.6.6.1/24
```

22. Set the family inet address for the loopback unit 0.

```
[edit]
user@router# set interfaces lo0 unit 0 family inet address 3.3.3.3/32
```

23. Set up OSPF for the ge-8/3/8.0 interface.

```
[edit]
user@router# set protocols ospf area 0.0.0.0 interface ge-8/3/8.0
```

24. Configure OSPF for the loopback interface.

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

25. Set up OSPF for the xe-0/1/3.0 interface.

```
[edit]
user@router# set protocols ospf area 0.0.0.0 interface xe-0/1/3.0
```

26. Configure OSPF for the ge-8/3/2.0 interface.

```
[edit]
user@router# set protocols ospf area 0.0.0.0 interface ge-8/3/2.0
```

27. Set up the static address for the physical interface module (PIM) rendezvous point (RP).

```
[edit]
user@router# set protocols pim rp static address 10.2.1.3
```

28. Configure the loopback interface to bidirectional sparse mode for the PIM protocol.

```
[edit]
user@router# set protocols pim interface lo0.0 mode bidirectional-sparse
```

29. Set the ge-8/3/8.0 interface to bidirectional sparse mode for the PIM protocol.

```
[edit]
user@router# set protocols pim interface ge-8/3/8.0 mode bidirectional-sparse
```

30. Configure the xe-0/1/3.0 interface to bidirectional sparse mode for the PIM protocol.

```
[edit]
user@router# set protocols pim interface xe-0/1/3.0 mode bidirectional-sparse
```

31. Set the ge-8/3/2.0 interface to bidirectional sparse mode for the PIM protocol.

```
[edit]
user@router# set protocols pim interface ge-8/3/2.0 mode bidirectional-sparse
```

32. Configure redundant graceful switchover on the chassis.

```
[edit]
user@router# set chassis redundancy graceful-switchover
```

33. Set the aggregated ethernet device count to 10.

```
[edit]
user@router# set chassis aggregated-devices ethernet device-count 10
```

34. Configure the tunnel services bandwidth for FPC 1/PIC 0.

```
[edit]
user@router# set chassis fpc 1 pic 0 tunnel-services bandwidth 10g
```

35. Enable enhanced IP for network services on the chassis.

```
[edit]
user@router# set chassis network-services enhanced-ip
```

Results

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

user@router# show switch-options

```
switch-options {
  vtep-source-interface lo0.0;
}
```

user@router# show bridge-domains

```
bridge-domains {
  vlan-5 {
    vxlan {
      vni 100;
      multicast-group 239.1.1.1;
    }
    vlan-id 100;
    routing-interface irb.0;
    interface xe-1/0/0.0;
  }
```

```
    vlan-6 {
      vxlan {
        vni 200;
        multicast-group 239.2.1.1;
      }
      vlan-id 200;
      routing-interface irb.1;
      interface xe-2/0/0.0;
    }
  }
```

user@router# show interfaces

```
  interfaces {
    xe-1/0/0 {
      vlan-tagging;
      encapsulation flexible-ethernet-services;
      unit 0 {
        encapsulation vlan-bridge;
        vlan-id 100;
      }
    }
    xe-2/0/0 {
      vlan-tagging;
      encapsulation flexible-ethernet-services;
      unit 0 {
        encapsulation vlan-bridge;
        vlan-id 200;
      }
    }
    irb {
      unit 0 {
        family inet {
          address 5.5.5.1/24;
        }
      }
      unit 1 {
        family inet {
          address 6.6.6.1/24;
        }
      }
    }
    lo0 {
      unit 0 {
        family inet {
          address 3.3.3.3/32;
        }
      }
    }
  }
```

user@router# show protocols ospf

```
  area 0.0.0.0 {
    interface ge-8/3/8.0;
    interface lo0.0;
    interface xe-0/1/3.0;
    interface ge-8/3/2.0;
```

```
  }
user@router# show protocols pim
  rp {
    static {
      address 10.2.1.3;
    }
  }
user@router# show chassis
  redundancy {
    graceful-switchover;
  }
  aggregated-devices {
    ethernet {
      device-count 10;
    }
  }
  fpc 1 {
    pic 0 {
      tunnel-services {
        bandwidth 10g;
      }
    }
  }
  network-services enhanced-ip;
```

Verification

Confirm that the configuration is working properly.

- Verifying Reachability on page 79
- Verifying VXLAN on page 80

Verifying Reachability

Purpose Verify that the network is up and running with the proper interfaces and routes installed.

Action

```
user@router> show interfaces terse irb
Interface               Admin Link Proto    Local                 Remote
irb                     up    up
irb.0                   up    up   inet     5.5.5.1/24
                                  multiservice
irb.1                   up    up   inet     6.6.6.1/24
                                  multiservice
```

```
user@router> ping 5.5.5.1/24
PING 5.5.5.1 (5.5.5.1): 56 data bytes
64 bytes from 5.5.5.1: icmp_seq=0 ttl=64 time=0.965 ms
64 bytes from 5.5.5.1: icmp_seq=1 ttl=64 time=0.960 ms
64 bytes from 5.5.5.1: icmp_seq=2 ttl=64 time=0.940 ms
^C
--- 1.1.1.1 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.940/0.955/0.965/0.011 ms
```

Meaning Use the **show interfaces terse irb** command to verify that the IRB interface has been properly configured. The **irb.0** and **irb.1** interfaces should display the proper multiservice inet addresses.

Use the **ping** command to confirm that the network is connected to the IRB multiservice address.

Verifying VXLAN

Purpose Verify that VXLAN is working and the proper protocols are enabled.

Action

```
user@router> show interfaces vtep
Physical interface: vtep, Enabled, Physical link is Up
  Interface index: 133, SNMP ifIndex: 575
  Type: Software-Pseudo, Link-level type: VxLAN-Tunnel-Endpoint, MTU: 1600, Speed:
 Unlimited
  Device flags   : Present Running
  Interface flags: SNMP-Traps
  Link type      : Full-Duplex
  Link flags     : None
  Last flapped   : Never
    Input packets : 0
    Output packets: 0

  Logical interface vtep.32768 (Index 334) (SNMP ifIndex 607)
    Flags: Up SNMP-Traps Encapsulation: ENET2
    VXLAN Endpoint Type: Source, VXLAN Endpoint Address: 10.255.187.32, L2 Routing
 Instance: default-switch, L3 Routing Instance: default
    Input packets : 0
    Output packets: 0
```

```
user@router> show l2-learning vxlan-tunnel-end-point remote mac-table
MAC flags (S -static MAC, D -dynamic MAC, L -locally learned, C -Control MAC
           SE -Statistics enabled, NM -Non configured MAC, R -Remote PE MAC)

Logical system   : <default>
Routing instance : default-switch
 Bridging domain : vlan-5+100, VLAN : 100, VNID : 100
 Bridging domain : vlan-6+200, VLAN : 200, VNID : 200
```

```
user@router> show l2-learning vxlan-tunnel-end-point source
Logical System Name       Id  SVTEP-IP         IFL   L3-Idx
<default>                 0   10.255.187.32    lo0.0    0
    L2-RTT                   Bridge Domain                VNID     MC-Group-IP
    default-switch           vlan-5+100                   100      239.1.1.1

    default-switch           vlan-6+200                   200      239.1.1.1
```

Meaning

Use the **show interface vtep** command to displays information about VXLAN endpoint configuration. Make sure the routing instance is assigned to the default-switch..

Use the **show l2-learning vxlan-tunnel-end-point remote mac-table** command to confirm that the bridging domain VLAN groups were configured correctly.

Use the **show l2-learning vxlan-tunnel-end-point source** command to confirm the multicast IP addresses for bridging domain VLAN groups.

Related Documentation

- Understanding VXLANs on page 3
- show bridge mac-table on page 172
- show vpls mac-table on page 176

Examples: Manually Configuring VXLANs on QFX Series Switches

Supported Platforms QFX Series standalone switches

These examples show how to configure VXLANs on QFX Series Switches for several use cases.

Example: Configuring a VXLAN Transit Switch

Supported Platforms QFX Series standalone switches

If a QFX switch acts as a transit switch for downstream devices acting as VTEPs, you do not need to configure any VXLAN information on the QFX 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 14.1X53-D10

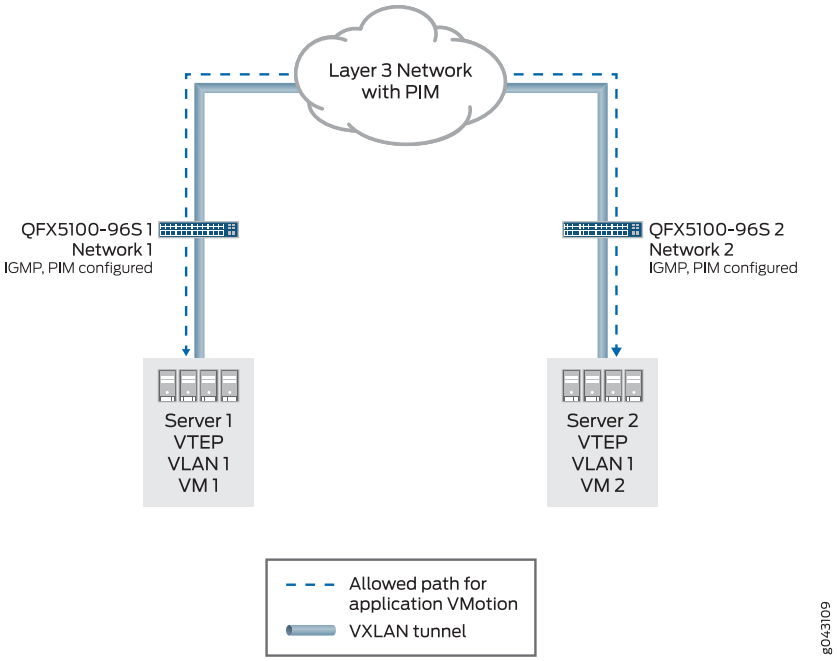
Overview

This example shows a simple use case in which QFX switches are connected to downstream servers acting as VTEPs and 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 83 shows a QFX 5100 switch configured to forward VXLAN packets for downstream VTEPs.

Figure 8: QFX5100 Acting as a VXLAN Transit Switch



Configuring PIM on the Transit Switches

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
```

Step-by-Step Procedure

If you are not using a 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 or 2 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
   ```

Example: Configuring a VXLAN Layer 2 Gateway

Supported Platforms QFX Series standalone switches

If a QFX 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 configure the switch to act as a VTEP—that is, a Layer 2 gateway for downstream Layer 2 devices. You also need to 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.

Requirements

This example uses the following hardware and software components:

- Two QFX5100 switches
- Junos OS 14.1X53-D10

Overview

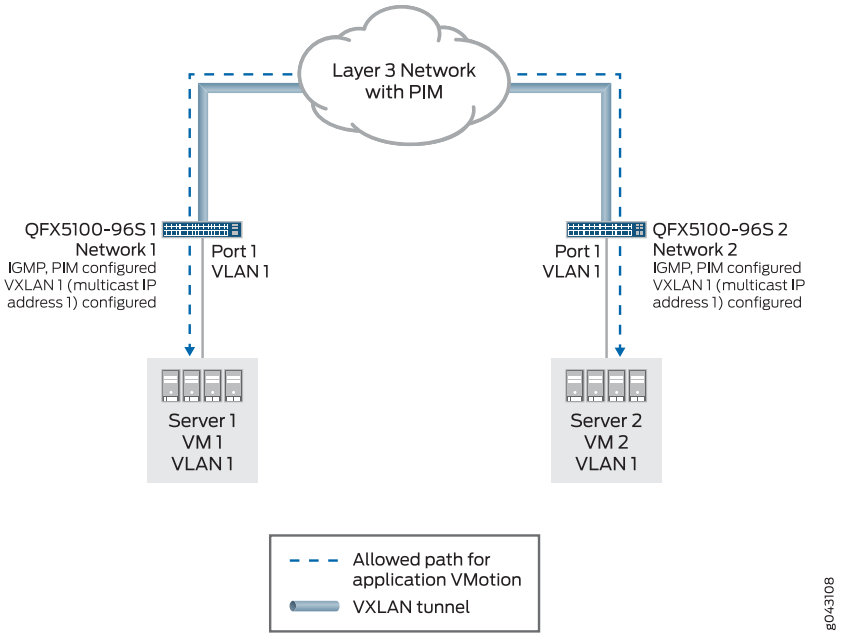
This example shows a use case in which QFX switches are connected to downstream VTEPs and need to allow Layer 2 connectivity between VM 1 on Server 1 and VM 2 on Server 2 so that VMotion can occur between the VMs. The servers in this example can be in the same or different data centers—the only constraint is that there must be Layer 3 connectivity between the QFX 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 85 shows a QFX 5100 switch configured to act as a VTEP.

Figure 9: QFX5100 Acting as a VTEP



Configuring the Switches

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 pim 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 224.2.2.2
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
```

The configuration for QFX5100-96S 2 is almost identical. The only changes 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 pim 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 224.2.2.2
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
```

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

Step-by-Step Procedure

Perform the following procedure on both switches to set up the example configuration. Note that you do not need to configure VLAN 1 or 2 on either switch.

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

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

5. Configure the address of a PIM rendezvous point:

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

6. 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 224.2.2.2
   ```

   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.

7. (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
   ```

8. (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
```

9. 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.**

10. 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.

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 224.2.2.2;
      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

Confirm that the configuration is working properly.

- Verifying VXLAN Reachability on page 88
- Verifying That the Local VTEP is Configured Correctly on page 89
- Verifying MAC Learning from the Remote VTEP on page 89
- Monitor the Remote Interface on page 89

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              224.2.2.2
```

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      224.2.2.2
```

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 This 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 This 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 on page 3
- VXLAN Constraints on QFX5100 Switches on page 9

Example: Configuring VXLAN to VPLS Stitching with OVSDB

Supported Platforms EX9200, MX Series

Virtual Extensible LAN (VXLAN) can be utilized with the Open vSwitch Database (OVSDB) management protocol in a VPLS-enabled network to stitch a virtualized data center into a Layer 2 VPN network. This configuration allows for seamless interconnection between different data centers using Layer 2 VPN regardless of whether it is virtualized, physical, or both.

- Requirements on page 91
- Overview on page 91
- Configuration on page 92
- Verification on page 99

Requirements

This example uses the following hardware and software components:

- Two MX Series routers running Junos OS 14.1R2 or later
- Two MX Series routers running Junos OS 14.1R2 or later with an OVSDB software package. The release of this package must be the same as the Junos OS release running on the device.
- One EX9200 switch
- One VMware NSX controller
- NSX Manager

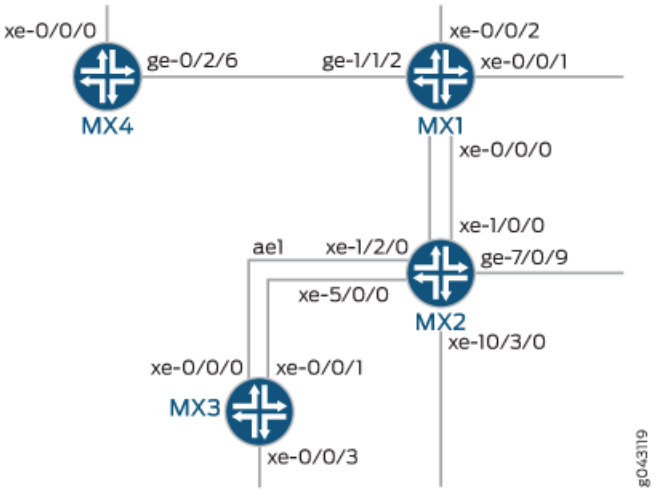
Before you start the configuration, you must perform the following tasks:

- In NSX Manager or the NSX API, configure a logical switch for each VXLAN that OVSDB will manage. This example implements two OVSDB-managed VXLANs, so you must configure two logical switches. After the configuration of each logical switch, NSX automatically generates a universally unique identifier (UUID) for the logical switch. If you have not done so already, retrieve the UUID for each logical switch. A sample UUID is 28805c1d-0122-495d-85df-19abd647d772. When configuring the equivalent VXLANs on the Juniper Networks device, you must use the UUID of the logical switch as the bridge domain name.
- Create an SSL private key and certificate, and install them in the **/var/db/certs** directory of the Juniper Networks device. For more information, see "Creating and Installing an SSL Key and Certificate on a Juniper Networks Device for Connection with SDN Controllers" on page 36.

Overview

In this example, four MX Series routers are configured to function together for VXLAN to virtual private LAN service (VPLS) stitching. Each router performs a different role in the configuration. The following diagram shows the topology of these MX Series routers. MX1 is the core router that handles Layer 3 traffic and protocols. MX2 is the VXLAN gateway router that functions as a virtual tunnel endpoint (VTEP) and handles switching for Layer 2, VPLS, and VXLAN. The MX3 router is configured to handle VPLS traffic. The MX4 router is configured as a VTEP to accept and decapsulate VXLAN packets.

Topology



Configuration

To configure VXLAN to VPLS stitching with OVSDB:

- Configuring MX1 on page 94
- Configuring MX2 on page 95
- Configuring MX3 on page 96
- Configuring MX4 on page 98
- Results on page 99

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.

MX1

```
set interfaces lo0 unit 0 family inet address 10.255.181.13/32 primary
set interfaces ge-1/1/2 unit 0 family inet address 30.30.30.2/30
set interfaces xe-0/0/0 unit 0 family inet address 20.20.20.2/30
set protocols ospf area 0.0.0.0 interface all
```

MX2

```
set interfaces lo0 unit 0 family inet address 10.255.181.72/32 primary
set lag-options interfaces <ae*> mtu 9192
set lag-options interfaces <ae*> aggregated ether-options minimum-links 1
set chassis aggregated-devices ethernet device-count 40
set chassis fpc 1 pic 0 tunnel-services bandwidth 10g
set chassis network-services enhanced-ip
set interfaces xe-1/2/0 gigether-options 802.3ad ae1
set interfaces xe-0/0/0 unit 0 family inet address 20.20.20.1/30
set interfaces ge-7/0/9 vlan tagging
set interfaces ge-7/0/9 unit 1 vlan-id 3
set interfaces ge-7/0/9 unit 1 family vpls
```

```
set interfaces xe-10/3/0 vlan tagging
set interfaces xe-10/3/0 unit 1 vlan-id 3
set interfaces xe-10/3/0 unit 0 family vpls
set interfaces ae1 unit 0 family inet address 1.1.1.1/30
set interfaces ae1 unit 0 family mpls
set routing-options autonomous-system 100
set protocols rsvp interface all
set protocols mpls no cspf
set protocols mpls label-switched-path-to-mx3 to 10.255.181.98
set protocols mpls interface all
set protocols bgp family l2vpn signaling
set protocols bgp group ibgp type internal
set protocols bgp group ibgp neighbor 10.255.181.98 local-address 10.255.181.72
set protocols ospf area 0.0.0.0 interface xe-0/0/0.0
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ae1.0
set protocols ovsdb traceoptions file ovsdb.log size 100m files 10
set protocols ovsdb traceoptions file ovsdb.level all
set protocols ovsdb traceoptions file ovsdb.flag all
set protocols ovsdb interfaces xe-10/3/0.1
set protocols ovsdb interfaces ge-7/0/9.1
set protocols ovsdb controller 192.168.182.45 protocol ssl port 6632
set routing-instances 24a76aff-7e61-4520-a78d-3eca26ad7510 vtep-source-interface
  lo0.0
set routing-instances 24a76aff-7e61-4520-a78d-3eca26ad7510 instance-type vpls
set routing-instances 24a76aff-7e61-4520-a78d-3eca26ad7510 vlan-id 3
set routing-instances 24a76aff-7e61-4520-a78d-3eca26ad7510 interface ge-7/0/9.1
set routing-instances 24a76aff-7e61-4520-a78d-3eca26ad7510 interface xe-10/3/0.1
set routing-instances 24a76aff-7e61-4520-a78d-3eca26ad7510 routing-interface irb.3
set routing-instances 24a76aff-7e61-4520-a78d-3eca26ad7510 vxlan ovsdb-managed
set routing-instances 24a76aff-7e61-4520-a78d-3eca26ad7510 vxlan vni 3
set routing-instances 24a76aff-7e61-4520-a78d-3eca26ad7510 route-distinguisher
  10.255.181.72:3
set routing-instances 24a76aff-7e61-4520-a78d-3eca26ad7510 vrf-target target:3:3
set routing-instances 24a76aff-7e61-4520-a78d-3eca26ad7510 protocols vpls site mx2
  site-identifier 1
```

MX3

```
set interfaces lo0 unit 0 family inet address 127.0.0.1/32
set interfaces lo0 unit 0 family inet address 10.255.181.98/32 primary
set lag-options interfaces <ae*> mtu 9192
set lag-options interfaces <ae*> aggregated-ether-options minimum-links 1
set lag-options interfaces <ae*> aggregated-ether-options lacp active
set interfaces xe-0/0/0 gigether-options 802.3ad ae1
set interfaces xe-0/0/1 gigether-options 802.3ad ae1
set interfaces xe-0/0/3 vlan tagging
set interfaces xe-0/0/3 unit 1 vlan-id 3
set interfaces xe-0/0/3 unit 1 family vpls
set interfaces ae1 unit 0 family inet address 1.1.1.2/30
set interfaces ae1 unit 0 family mpls
set routing-options autonomous-system 100
set protocols rsvp interface all
set protocols mpls no-cspf
set protocols mpls label-switched-path-to-mx2 to 10.255.181.72
set protocols mpls interface all
set protocols bgp family l2vpn signaling
```

```
set protocols bgp group ibgp type internal
set protocols bgp group ibgp neighbor 10.255.181.72 local-address 10.255.181.98
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ae1.0
set routing-instances vpls3 instance-type vpls
set routing-instances vpls3 vlan-id 3
set routing-instances vpls3 interface xe-0/0/3.1
set routing-instances vpls3 route-distinguisher 10.255.181.98:3
set routing-instances vpls3 vrf-target target:3:3
set routing-instances vpls3 protocols vpls no-tunnel-services
set routing-instances vpls3 protocols vpls site mx3 site-identifier 2
```

MX4

```
set interfaces lo0 unit 0 family inet address 10.255.181.43/32 primary
set interfaces xe-0/0/0 vlan-tagging
set interfaces xe-0/0/0 encapsulation flexible-ethernet-services
set interfaces xe-0/0/0 unit 0 family bridge interface-mode trunk
set interfaces xe-0/0/0 unit 0 family bridge vlan-id-list 1-10
set interfaces ge-0/2/6 unit 0 family inet address 30.30.30.1/30
set protocols ospf area 0.0.0.0 interface ge-0/2/6.0
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ovsdb traceoptions file ovsdb.log size 100m files 10
set protocols ovsdb traceoptions level all
set protocols ovsdb traceoptions flag all
set protocols ovsdb interfaces xe-0/0/0.0
set protocols ovsdb controller 192.168.182.45 protocol ssl port 6632
set routing-instances default-vs1 vtep-source-interface lo0.0
set routing-instances default-vs1 instance-type virtual-switch
set routing-instances default-vs1 interface xe-0/0/0.1
set bridge-domains 24a76aff-7e61-4520-a78d-3eca26ad7510 vlan-id 3
set bridge-domains 24a76aff-7e61-4520-a78d-3eca26ad7510 vxlan ovsdb-managed
set bridge-domains 24a76aff-7e61-4520-a78d-3eca26ad7510 vxlan vni 3
set bridge-domains 24a76aff-7e61-4520-a78d-3eca26ad7510 vxlan
  ingress-node-replication
set switch-options vtep-source-interface lo0.0
```

Configuring MX1

Step-by-Step Procedure

The first router to be configured is the core router. This MX Series router handles Layer 3 traffic and protocols for the rest of the network.

To configure the MX1 router:

1. Specify the IPv4 address for the loopback interface.

   ```
   [edit interfaces]
   user@MX1# set lo0 unit 0 family inet address 10.255.181.13/32 primary
   ```

2. Configure the Layer 3 network.

   ```
   [edit interfaces]
   user@MX1# set ge-1/1/2 unit 0 family inet address 30.30.30.2/30
   user@MX1# set xe-0/0/0 unit 0 family inet address 20.20.20.2/30
   ```

3. Enable OSPF on all interfaces.

   ```
   [edit protocols]
   ```

```
user@MX1# set ospf area 0.0.0.0 interface all
```

Configuring MX2

Step-by-Step Procedure

The second router to be configured is the VXLAN gateway router. This MX Series router is configured as a VTEP, and it handles switching for Layer 2, VPLS, and VXLAN.

To configure the MX2 router:

1. Configure interfaces for the VXLAN gateway.

```
[edit interfaces]
user@MX2# set lo0 unit 0 family inet address 10.255.181.72/32 primary
user@MX2# set xe-1/2/0 gigether-options 802.3ad ae1
user@MX2# set xe-0/0/0 unit 0 family inet address 20.20.20.1/30
user@MX2# set ge-7/0/9 vlan tagging
user@MX2# set ge-7/0/9 unit 1 vlan-id 3
user@MX2# set ge-7/0/9 unit 1 family vpls
user@MX2# set xe-10/3/0 vlan tagging
user@MX2# set xe-10/3/0 unit 1 vlan-id 3
user@MX2# set xe-10/3/0 unit 0 family vpls
user@MX2# set ae1 unit 0 family inet address 1.1.1.1/30
user@MX2# set ae1 unit 0 family mpls
```

2. Set up LAG options

```
[edit lag-options]
user@MX2# set interfaces <ae*> mtu 9192
user@MX2# set interfaces <ae*> aggregated ether-options minimum-links 1
```

3. Configure chassis settings.

```
[edit chassis]
user@MX2# set aggregated-devices ethernet device-count 40
user@MX2# set fpc 1 pic 0 tunnel-services bandwidth 10g
user@MX2# set network-services enhanced-ip
```

4. Configure routing options.

```
[edit routing-options]
user@MX2# set autonomous-system 100
```

5. Set up RSVP, MPLS, and BGP protocols.

```
[edit protocols]
user@MX2# set rsvp interface all
user@MX2# set mpls no cspf
user@MX2# set mpls label-switched-path-to-mx3 to 10.255.181.98
user@MX2# set mpls interface all
user@MX2# set bgp family l2vpn signaling
user@MX2# set bgp group ibgp type internal
user@MX2# set bgp group ibgp neighbor 10.255.181.98 local-address 10.255.181.72
```

6. Configure OSPF interface settings.

```
[edit protocols]
user@MX2# set ospf area 0.0.0.0 interface xe-0/0/0.0
user@MX2# set ospf area 0.0.0.0 interface fxp0.0 disable
user@MX2# set ospf area 0.0.0.0 interface lo0.0 passive
```

```
user@MX2# set ospf area 0.0.0.0 interface ae1.0
```

7. Set up OVSDB tracing operations.

```
[edit protocols]
user@MX2# set ovsdb traceoptions file ovsdb.log size 100m files 10
user@MX2# set ovsdb traceoptions file ovsdb.level all
user@MX2# set ovsdb traceoptions file ovsdb.flag all
```

8. Specify that interfaces xe-10/3/0.1 and ge-7/0/9.1 are managed by OVSDB.

```
[edit protocols]
user@MX2# set ovsdb interfaces xe-10/3/0.1
user@MX2# set ovsdb interfaces ge-7/0/9.1
```

9. Configure a connection with an NSX controller.

```
[edit protocols]
user@MX2# set ovsdb controller 192.168.182.45 protocol ssl port 6632
```

10. Create a VPLS routing instance with VXLAN functionality.

```
[edit routing-instances]
set 24a76aff-7e61-4520-a78d-3eca26ad7510 vtep-source-interface lo0.0
set 24a76aff-7e61-4520-a78d-3eca26ad7510 instance-type vpls
set 24a76aff-7e61-4520-a78d-3eca26ad7510 vlan-id 3
set 24a76aff-7e61-4520-a78d-3eca26ad7510 interface ge-7/0/9.1
set 24a76aff-7e61-4520-a78d-3eca26ad7510 interface xe-10/3/0.1
set 24a76aff-7e61-4520-a78d-3eca26ad7510 routing-interface irb.3
set 24a76aff-7e61-4520-a78d-3eca26ad7510 vxlan ovsdb-managed
set 24a76aff-7e61-4520-a78d-3eca26ad7510 vxlan vni 3
set 24a76aff-7e61-4520-a78d-3eca26ad7510 route-distinguisher 10.255.181.72:3
set 24a76aff-7e61-4520-a78d-3eca26ad7510 vrf-target target:3:3
set 24a76aff-7e61-4520-a78d-3eca26ad7510 protocols vpls site mx2 site-identifier
  1
```

NOTE: After completing this configuration, you must configure a gateway, which is the NSX equivalent of a hardware VTEP. This configuration implements one hardware VTEP, so you must configure one gateway, a gateway service, and a logical switch port using NSX Manager or the NSX API. For more information about the tasks you must perform as well as key NSX Manager configuration details, see "VMware NSX Configuration for Juniper Networks Devices Functioning as Virtual Tunnel Endpoints" on page 39.

Configuring MX3

Step-by-Step Procedure

The third MX Series router must be configured to handle VPLS traffic.

To configure the MX3 router:

1. Specify the IPv4, IPv6, and ISO addresses for the loopback interface.

```
[edit interfaces]
user@MX3# set lo0 unit 0 family inet address 127.0.0.1/32
```

```
user@MX3# set lo0 unit 0 family inet address 10.255.181.98/32 primary
```

2. Configure the network interfaces.

```
[edit interfaces]
user@MX3# set xe-0/0/0 gigether-options 802.3ad ae1
user@MX3# set xe-0/0/1 gigether-options 802.3ad ae1
user@MX3# set xe-0/0/3 vlan tagging
user@MX3# set xe-0/0/3 unit 1 vlan-id 3
user@MX3# set xe-0/0/3 unit 1 family vpls
user@MX3# set ae1 unit 0 family inet address 1.1.1.2/30
user@MX3# set ae1 unit 0 family mpls
```

3. Set up LAG options

```
[edit lag-options]
user@MX3# set interfaces <ae*> mtu 9192
user@MX3# set interfaces <ae*> aggregated-ether-options minimum-links 1
user@MX3# set interfaces <ae*> aggregated-ether-options lacp active
```

4. Configure routing options.

```
[edit routing-options]
user@MX3# set autonomous-system 100
```

5. Set up RSVP, MPLS, and BGP protocols.

```
[edit protocols]
user@MX3# set rsvp interface all
user@MX3# set mpls no cspf
user@MX3# set mpls label-switched-path-to-mx2 to 10.255.181.72
user@MX3# set mpls interface all
user@MX3# set bgp family l2vpn signaling
user@MX3# set bgp group ibgp type internal
user@MX3# set bgp group ibgp neighbor 10.255.181.72 local-address 10.255.181.98
```

6. Configure OSPF interface settings.

```
[edit protocols]
user@MX3# set ospf area 0.0.0.0 interface lo0.0 passive
user@MX3# set ospf area 0.0.0.0 interface ae1.0
```

7. Create a VPLS routing instance.

```
[edit routing-instances]
set vpls3 instance-type vpls
set vpls3 vlan-id 3
set vpls3 interface xe-0/0/3.1
set vpls3 route-distinguisher 10.255.181.98:3
set vpls3 vrf-target target:3:3
set vpls3 protocols vpls no-tunnel-services
set vpls3 protocols vpls site mx3 site-identifier 2
```

Configuring MX4

Step-by-Step Procedure

The fourth MX Series router is configured as a VTEP to accept and decapsulate VXLAN packets.

To configure the MX4 router:

1. Specify the IPv4, IPv6, and ISO addresses for the loopback interface.

```
[edit interfaces]
user@MX4# set lo0 unit 0 family inet address 10.255.181.43/32 primary
```

2. Configure the interfaces.

```
[edit interfaces]
user@MX4# set xe-0/0/0 vlan-tagging
user@MX4# set xe-0/0/0 encapsulation flexible-ethernet-services
user@MX4# set xe-0/0/0 unit 0 family bridge interface-mode trunk
user@MX4# set xe-0/0/0 unit 0 family bridge vlan-id-list 1-10
user@MX4# set ge-0/2/6 unit 0 family inet address 30.30.30.1/30
```

3. Configure OSPF interface settings.

```
[edit protocols]
user@MX4# set ospf area 0.0.0.0 interface ge-0/2/6.0
user@MX4# set ospf area 0.0.0.0 interface fxp0.0 disable
user@MX4# set ospf area 0.0.0.0 interface lo0.0 passive
```

4. Set up OVSDB tracing operations.

```
[edit protocols]
user@MX4# set ovsdb traceoptions file ovsdb.log size 100m files 10
user@MX4# set ovsdb traceoptions level all
user@MX4# set ovsdb traceoptions flag all
```

5. Specify that the xe-0/0/0.0 interface is managed by OVSDB.

```
[edit protocols]
user@MX4# set ovsdb interfaces xe-0/0/0.0
```

6. Configure a connection with an NSX controller.

```
[edit protocols]
user@MX4# set ovsdb controller 192.168.182.45 protocol ssl port 6632
```

7. Configure the VPLS interface.

```
[edit routing-instances]
user@MX4# set default-vs1 vtep-source-interface lo0.0
user@MX4# set default-vs1 instance-type virtual-switch
user@MX4# set default-vs1 interface xe-0/0/0.1
```

8. Configure a set of VXLAN-enabled bridge domains.

```
[edit bridge-domains]
user@MX4# set 24a76aff-7e61-4520-a78d-3eca26ad7510 vlan-id 3
user@MX4# set 24a76aff-7e61-4520-a78d-3eca26ad7510 vxlan ovsdb-managed
user@MX4# set 24a76aff-7e61-4520-a78d-3eca26ad7510 vxlan vni 3
user@MX4# set 24a76aff-7e61-4520-a78d-3eca26ad7510 vxlan
  ingress-node-replication
```

9. Configure the loopback interface to be used as the tunnel source address.

```
[edit switch-options]
user@MX4# set vtep-source-interface lo0.0
```

NOTE: After completing this configuration, you must configure a gateway, which is the NSX equivalent of a hardware VTEP. This configuration implements one hardware VTEP, so you must configure one gateway, a gateway service, and a logical switch port using NSX Manager or the NSX API. For more information about the tasks you must perform as well as key NSX Manager configuration details, see "VMware NSX Configuration for Juniper Networks Devices Functioning as Virtual Tunnel Endpoints" on page 39.

Results

From configuration mode, confirm your configuration by entering the following commands on each router. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

Verification

Confirm that the configuration is working properly.

Verifying MX1

Purpose Verify your configuration on MX1.

Action Verify that the interfaces are configured properly.

```
user@MX1# show interface

    lo0 {
      unit 0 {
        family inet {
          address 10.255.181.13/32 {
            primary;
          }
        }
      }
    }
  ge-1/1/2 {
    unit 0 {
      family inet {
        address 30.30.30.2/30;
      }
    }
  }
  xe-0/0/0 {
    unit 0 {
      family inet {
        address 20.20.20.2/30;
      }
    }
  }
```

Verify that OSPF is configured correctly.

```
user@MX1# show protocols

    ospf {
      area 0.0.0.0 {
        interface all;
      }
    }
```

Verifying MX2

Purpose Verify your configuration on MX2.

Action Verify that the interfaces are configured properly.

```
user@MX2# show interfaces
```

```
    lo0 {
      unit 0 {
        family inet {
          address 10.255.181.72/32 {
            primary;
          }
        }
      }
    }
  xe-1/2/0 {
    gigether-options {
      802.3ad ae1;
    }
  }
  xe-0/0/0 {
    unit 0 {
      family inet {
        address 20.20.20.1/30;
      }
    }
  }
  ge-7/0/9 {
    vlan-tagging;
    unit 0 {
      family vpls;
    }
    unit 1 {
      vlan-id 3;
    }
  }
  xe-10/3/0 {
    vlan-tagging;
    unit 0 {
      family vpls;
    }
    unit 1 {
      vlan-id 3;
    }
  }
  ae1 {
    unit 0 {
      family inet {
        address 1.1.1.1/30;
      }
      family mpls;
    }
  }
```

Verify that OSPF is configured properly.

```
user@MX2# show protocols ospf
```

```
    ospf {
    area 0.0.0.0 {
      interface xe-0/0/0.0;
      interface fxp0.0 {
        disable;
      }
      interface lo0.0 {
        passive;
      }
      interface ae1.0;
    }
  }
```

Verify that OVSDB is configured properly.

```
user@MX2# show protocols ovsdb
```

```
  ovsdb {
    traceoptions {
      file ovsdb.log size 100m files 10;
      level all;
      flag all;
    }
    interfaces {
      xe-10/3/0.1;
      ge-7/0/9.0;
      ge-7/0/9.1;
    }
    controller 192.168.182.45 {
      protocol {
        ssl port 6632;
      }
    }
  }
```

Verify the **default-VS1** routing instance configuration.

```
user@MX2# show routing-instances
```

```
  routing-instances {
    24a76aff-7e61-4520-a78d-3eca26ad7510 {
      vtep-source-interface lo0.0;
      instance-type vpls;
      vlan-id 3;
      interface ge-7/0/9.1;
      interface xe-10/3/0.1;
      routing-interface irb.3;
      vxlan {
        ovsdb-managed;
        vni 3;
        encapsulate-inner-vlan;
        decapsulate-accept-inner-vlan;
        ingress-node-replication;
      }
      route-distinguisher 10.255.181.72:3;
      vrf-target target:3:3;
      protocols {
        vpls {
```



```
            traceoptions {
            file vpls.log;
            flag all;
          }
          site moneycar {
            site-identifier 1;
          }
        }
      }
    }
    cadbc185-f60f-48a6-93fd-dc14a6420c60 {
      vtep-source-interface lo0.0;
      instance-type vpls;
      vlan-id 2;
      interface ge-7/0/9.0;
      interface xe-10/3/0.0;
      routing-interface irb.2;
      vxlan {
        ovsdb-managed;
        vni 2;
        encapsulate-inner-vlan;
        decapsulate-accept-inner-vlan;
        ingress-node-replication;
      }
      route-distinguisher 10.255.181.72:10;
      vrf-target target:10:10;
      protocols {
        vpls {
          traceoptions {
            file vpls.log;
            flag all;
          }
          site moneycar {
            site-identifier 1;
          }
        }
      }
    }
    vpls11 {
      vtep-source-interface lo0.1;
      instance-type vpls;
      vlan-id 11;
      interface ge-7/0/9.11;
      interface xe-10/3/0.11;
      routing-interface irb.11;
      vxlan {
        ovsdb-managed;
        vni 11;
        ingress-node-replication;
      }
      route-distinguisher 10.255.181.72:11;
      vrf-target target:11:11;
      protocols {
        vpls {
          traceoptions {
            file vpls.log;
```

```
            flag all;
        }
        site moneycar {
          site-identifier 1;
        }
      }
    }
  }
  vpls12 {
    vtep-source-interface lo0.1;
    instance-type vpls;
    vlan-id 12;
    interface ge-7/0/9.12;
    interface xe-10/3/0.12;
    routing-interface irb.12;
    vxlan {
      ovsdb-managed;
      vni 12;
      ingress-node-replication;
    }
    route-distinguisher 10.255.181.72:12;
    vrf-target target:12:12;
    protocols {
      vpls {
        traceoptions {
          file vpls.log;
          flag all;
        }
        site moneycar {
          site-identifier 1;
        }
      }
    }
  }
  vpls13 {
    vtep-source-interface lo0.1;
    instance-type vpls;
    vlan-id 13;
    interface ge-7/0/9.13;
    interface xe-10/3/0.13;
    routing-interface irb.13;
    vxlan {
      vni 13;
      multicast-group 228.1.1.13;
    }
    route-distinguisher 10.255.181.72:13;
    vrf-target target:13:13;
    protocols {
      vpls {
        traceoptions {
          file vpls.log;
          flag all;
        }
        site moneycar {
          site-identifier 1;
        }
```

```
        }
    }
}
vpls14 {
    vtep-source-interface lo0.1;
    instance-type vpls;
    vlan-id 14;
    interface ge-7/0/9.14;
    interface xe-10/3/0.14;
    routing-interface irb.14;
    vxlan {
        vni 14;
        multicast-group 228.1.1.14;
    }
    route-distinguisher 10.255.181.72:14;
    vrf-target target:14:14;
    protocols {
        vpls {
            traceoptions {
                file vpls.log;
                flag all;
            }
            site moneycar {
                site-identifier 1;
            }
        }
    }
}
vpls15 {
    vtep-source-interface lo0.1;
    instance-type vpls;
    vlan-id 15;
    interface ge-7/0/9.15;
    interface xe-10/3/0.15;
    routing-interface irb.15;
    vxlan {
        vni 15;
        multicast-group 228.1.1.15;
    }
    route-distinguisher 10.255.181.72:15;
    vrf-target target:15:15;
    protocols {
        vpls {
            traceoptions {
                file vpls.log;
                flag all;
            }
            site moneycar {
                site-identifier 1;
            }
        }
    }
}
vpls4 {
    vtep-source-interface lo0.0;
    instance-type vpls;
```

```
    vlan-id 4;
  interface ge-7/0/9.4;
  interface xe-10/3/0.4;
  routing-interface irb.4;
  vxlan {
    vni 4;
    multicast-group 228.1.1.4;
  }
  route-distinguisher 10.255.181.72:4;
  vrf-target target:4:4;
  protocols {
    vpls {
      traceoptions {
        file vpls.log;
        flag all;
      }
      site moneycar {
        site-identifier 1;
      }
    }
  }
}
vpls5 {
  vtep-source-interface lo0.0;
  instance-type vpls;
  vlan-id 5;
  interface ge-7/0/9.5;
  interface xe-10/3/0.5;
  routing-interface irb.5;
  vxlan {
    vni 5;
    multicast-group 228.1.1.5;
  }
  route-distinguisher 10.255.181.72:5;
  vrf-target target:5:5;
  protocols {
    vpls {
      traceoptions {
        file vpls.log;
        flag all;
      }
      site moneycar {
        site-identifier 1;
      }
    }
  }
}
vrf1 {
  instance-type vrf;
  interface ae2.0;
  interface lo0.1;
  route-distinguisher 100:100;
  vrf-target target:100:100;
  protocols {
    ospf {
      area 0.0.0.0 {
```

```
            interface ae2.0;
          interface lo0.1 {
            passive;
          }
        }
      }
      pim {
        rp {
          static {
            address 10.255.181.13;
          }
        }
        interface all;
      }
    }
  }
}
```

Verify the **vrf1** routing instance configuration.

user@MX2# show routing-instances

```
24a76aff-7e61-4520-a78d-3eca26ad7510 {
  vtep-source-interface lo0.0;
  instance-type vpls;
  vlan-id 3;
  interface ge-7/0/9.1;
  interface xe-10/3/0.1;
  routing-interface irb.3;
  vxlan {
    ovsdb-managed;
    vni 3;
  }
  route-distinguisher 10.255.181.72:3;
  vrf-target target:3:3;
  protocols {
    vpls {
      site mx2 {
        site-identifier 1;
      }
    }
  }
}
```

Verifying MX3

Purpose Verify your configuration on MX3.

Action Verify that the interfaces are configured properly.

```
user@MX3# show interfaces
```

```
xe-0/0/0 {
    gigether-options {
        802.3ad ae1;
    }
}
xe-0/0/1 {
    gigether-options {
        802.3ad ae1;
    }
}
xe-0/0/3 {
    vlan-tagging;
    unit 1 {
        vlan-id 3;
        family vpls;
    }
}
ae1 {
    unit 0 {
        family inet {
            address 1.1.1.2/30;
        }
        family mpls;
    }
}
```

Verify the RSVP, MPLS, BGP and OSPF protocols are configured properly.

```
user@MX3# show protocols
```

```
protocols {
    rsvp {
        interface all;
    }
    mpls {
        no-cspf;
        label-switched-path to-mx2 {
            to 10.255.181.72;
        }
        interface all;
    }
    bgp {
        family l2vpn {
            signaling;
        }
        group ibgp {
            type internal;
            neighbor 10.255.181.72 {
                local-address 10.255.181.98;
            }
        }
    }
```

```
        ospf {
        area 0.0.0.0 {
          interface lo0.0 {
            passive;
          }
          interface ae1.0;
        }
      }
    }
```

Verify the VPLS routing instance configuration.

user@MX3# show routing-instances

```
  routing-instances {
    vpls3 {
      instance-type vpls;
      vlan-id 3;
      interface xe-0/0/3.1;
      route-distinguisher 10.255.181.98:3;
      vrf-target target:3:3;
      protocols {
        vpls {
          no-tunnel-services;
          site mx3 {
            site-identifier 2;
          }
        }
      }
    }
  }
```

Verifying MX4

Purpose Verify your configuration on MX4.

Action Verify that the global group interfaces are configured properly.

user@MX4# show groups global interfaces

Verify that the interfaces are configured properly.

user@MX4# show interfaces

```
    lo0 {
      unit 0 {
        family inet {
          address 10.255.181.43/32 {
            primary;
          }
        }
      }
    }
  xe-0/0/0 {
    vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 0 {
      family bridge {
        interface-mode trunk;
        vlan-id-list 1-10;
      }
    }
  }
  ge-0/2/6 {
    unit 0 {
      family inet {
        address 30.30.30.1/30;
      }
    }
  }
```

Verify that the OSPF interface settings are configured properly.

user@MX4# show protocols ospf

```
        area 0.0.0.0 {
            interface ge-0/2/6.0;
            interface fxp0.0 {
                disable;
            }
            interface lo0.0 {
                passive;
            }
        }
```

Verify that OVSDB is configured properly.

user@MX4# show protocols ovsdb

```
        traceoptions {
            file ovsdb.log size 100m files 10;
            level all;
            flag all;
        }
        interfaces {
            xe-0/0/0.0;
```



```
        }
        controller 192.168.182.45 {
            protocol {
                ssl port 6632;
            }
        }
```

Verify the **default-VS1** routing instance configuration and bridge domains.

```
user@MX4# show routing-instances default-VS1
        vtep-source-interface lo0.0;
        instance-type virtual-switch;
        interface xe-0/0/0.1;
```

Verify that the bridge domains are configured properly.

```
user@MX4# show bridge-domains
    24a76aff-7e61-4520-a78d-3eca26ad7510 {
        vlan-id 3;
        vxlan {
            ovsdb-managed;
            vni 3;
            ingress-node-replication;
        }
    }
```

Verify that the loopback interface is used as the tunnel source address.

```
user@MX4# show switch-options
    vtep-source-interface lo0.0;
```

Related Documentation

- Creating and Installing an SSL Key and Certificate on a Juniper Networks Device for Connection with SDN Controllers on page 36

Example: Configuring Storm Control on an OVSDB-Managed Interface

Supported Platforms QFX Series standalone switches

You can configure a QFX5100 switch to act as a virtual tunnel endpoint (VTEP) in a Virtual Extensible LAN (VXLAN) managed by an Open vSwitch Database (OVSDB) controller such as a VMWare NSX controller. In this environment, you must use a service node to replicate and forward Layer 2 broadcast, unknown unicast, and multicast (BUM) traffic that originates in the VXLAN. Because service nodes can be overloaded if they receive too much BUM traffic, you might want to configure storm control on server-facing VXLAN interfaces to control how much of this traffic is allowed into a VXLAN (and therefore into a service node).

The OVSDB controller automatically configures these interfaces but cannot supply storm-control configuration. To enable storm control on an OVSDB-managed interface, you must add configuration statements to it manually.

Requirements

This example assumes you have the following hardware and software components:

- One QFX Series switch running Junos OS with ELS and configured as a VXLAN VTEP
- Junos OS Release 14.1X53-D26
- One OVSDB-based VXLAN controller
- One service node for replicating and forwarding BUM traffic

Overview and Topology

You can use storm control to prevent the service node from being overloaded by specifying the amount of BUM traffic to be allowed on a server-facing Layer 2 interface that accepts traffic that will be VXLAN encapsulated. This controls the amount of BUM traffic that reaches the service node. You specify the amount of traffic—or *storm control level*—as the traffic rate in kilobits per second (Kbps) of the combined applicable traffic streams or as the percentage of available bandwidth used by the combined applicable traffic streams.

NOTE: **To configure storm control on an OVSDB-managed aggregated Ethernet interface, you must configure it on each member interface (not the aggregated interface) and the storm-control level applies to each member individually.**

Storm control monitors the level of applicable incoming traffic and compares it with the level that you specify. If the combined level of the applicable traffic exceeds the specified level, the switch drops packets for the controlled traffic types. (You cannot use the *action-shutdown* statement to temporarily disable an OVSDB-managed interface if the controlled traffic exceeds the storm-control level.)

The topology used in this example consists of a switch configured as a VTEP that connects to servers that will send Layer 2 traffic over a VXLAN that is configured on the switch. This example shows how to configure the storm control level on OVSDB-managed interface xe-0/0/0 by setting the level to a traffic rate of 15,000 Kbps for the combined applicable traffic streams. If the combined traffic exceeds this level, the switch drops packets for the controlled traffic types to prevent the service node from being overloaded.

Configuration

CLI Quick Configuration

To quickly configure storm control based on the traffic rate in kilobits per second of the combined traffic streams on an OVSDB-managed interface, copy the following command and paste it into the switch terminal window:

```
[edit]
set forwarding-options storm-control-profiles sc-profile all bandwidth-level 15000
set interfaces xe-0/0/0 ether-options ethernet-switch-profile storm-control sc-profile
```

Step-by-Step Procedure

To configure storm control:

1. Configure a storm control profile, **sc-profile**, and specify the traffic rate in kilobits per second of the combined traffic streams:

   ```
   [edit]
   user@switch# set forwarding-options storm-control-profiles sc-profile all bandwidth-level 15000
   ```

2. Bind the storm control profile, **sc-profile**, to a Layer 2 interface:

   ```
   [edit]
   user@switch# set interfaces xe-0/0/0 ether-options ethernet-switch-profile storm-control sc-profile
   ```

Results

Display the results of the configuration:

```
[edit forwarding-options]
user@switch# show storm-control-profiles sc-profile
all {
  bandwidth 15000;
}
```

```
[edit]
user@switch# show interfaces xe-0/0/0
flexible-vlan-tagging;
encapsulation extended-vlan-bridge;
  ether-options {
    ethernet-switch-profile {
      storm-control sc-profile;
  }
}
```

Note that the statements **flexible-vlan-tagging** and **encapsulation extended-vlan-bridge** are automatically configured on the VXLAN interface by the OVSDB controller.

Related Documentation

- *Understanding Storm Control*
- *storm-control*
- *storm-control-profiles*

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

Supported Platforms MX Series, QFX Series standalone switches

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 a Contrail environment.
- If the **Flags** field displays **Tunnel key mismatch**, the VXLAN network identifier (VNI) specified in the logical switch or the VXLAN identifier specified in virtual network 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 QFX5100 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 automatically 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 automatically 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. For this situation, enter the **show ovsdb commit failures** operational mode command. In the output that displays, determine which configurations are erroneous. 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 an automatically configured VXLAN and in a VXLAN that was previously configured using the Junos OS CLI. 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.
- On all other Juniper Networks devices that support VXLAN and OVSDB, determine whether a corresponding VXLAN is configured on the device. If the VXLAN is not configured, configure it using the procedure in "Configuring OVSDB-Managed VXLANs"

on page 42. 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 QFX5100 switch, two issues exist. First, despite the fact that the Juniper Networks device automatically creates OVSDB-managed VXLANs, this VXLAN was manually 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 automatically creates a corresponding VXLAN and deletes the manually configured VXLAN.
- On all other Juniper Networks devices that support VXLAN and OVSDB, determine whether a corresponding logical switch is configured in the NSX environment or 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 QFX5100 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 created the corresponding VXLAN. If it was changed, update the VNI on the QFX5100 switch, using the Junos OS CLI.
- On all other Juniper Networks devices that support VXLAN and OVSDB, check the values of the VNI in the NSX environment or the VXLAN Identifier in the Contrail environment, and the Junos OS CLI. Change the incorrect value.

Related Documentation

- Understanding Automatically Configured VXLANs in an OVSDB Environment on page 20
- show ovsdb logical-switch on page 154
- show ovsdb commit failures on page 148
- clear ovsdb commit failures on page 146

PART 3

Configuration Statements and Operational Commands

CHAPTER 3

OVSDB Configuration Statements

controller (OVSDB)

Supported Platforms MX Series, QFX Series standalone switches

Syntax

```
controller ip-address {
  inactivity-probe-duration milliseconds;
  maximum-backoff-duration milliseconds;
  protocol protocol {
    port number;
  }
}
```

Hierarchy Level [edit protocols ovsdb]

Release Information Statement introduced in Junos OS Release 14.1R2.
Statement introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Statement introduced in Junos OS Release 14.2 for EX Series switches.

Description Configure a connection between a Juniper Networks device running the Open vSwitch Database (OVSDB) management protocol and a software-defined networking (SDN) controller. 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 NSX controller, using the Junos OS CLI. If the NSX 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.

Options ***ip-address*** —IPv4 address of the SDN controller.

The remaining statements are explained separately.

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

Related Documentation

- Understanding How to Set Up Open vSwitch Database Connections Between Juniper Networks Devices and SDN Controllers on page 14

inactivity-probe-duration

Supported Platforms MX Series, QFX Series standalone switches

Syntax inactivity-probe-duration *milliseconds*;

Hierarchy Level [edit protocols ovsdb controller]

Release Information Statement introduced in Junos OS Release 14.1R2.
Statement introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Statement introduced in Junos OS Release 14.2 for EX Series switches.

Description Configure the maximum amount of time, in milliseconds, that the connection between a Juniper Networks device that supports the Open vSwitch Database (OVSDB) management protocol and a software-defined networking (SDN) controller can be inactive before an inactivity probe is sent.

Options ***milliseconds***—Number of milliseconds that the connection can be inactive before an inactivity probe is sent.

Range: 0 through 4,294,967,295

Default: 0. This value indicates that an inactivity probe is never sent.

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

Related Documentation

- Understanding How to Set Up Open vSwitch Database Connections Between Juniper Networks Devices and SDN Controllers on page 14

ingress-node-replication

Supported Platforms EX Series, MX Series

Syntax ingress-node-replication;

Hierarchy Level [edit bridge-domains *bridge-domain-name* vxlan],
[edit routing-instances *routing-instance-name* bridge-domains *bridge-domain-name* vxlan],
[edit routing-instances *routing-instance-name* vlans *vlan-name* vxlan],
[edit routing-instances *routing-instance-name* vxlan]
[edit vlans *vlan-name* vxlan]

Release Information Statement introduced in Junos OS Release 14.1R2.
Statement introduced in Junos OS Release 14.2 for EX Series switches.

Description Enable ingress node replication for a specified Virtual Extensible LAN (VXLAN) that is managed by the Open vSwitch Database (OVSDB) management protocol.

With this feature enabled, instead of service nodes, Juniper Networks devices with OVSDB implemented handle incoming broadcast, unknown unicast, or multicast (BUM) traffic. For more information about the scenarios in which you can use ingress node replication and how it works, see "Understanding How Layer 2 BUM and Layer 3 Routed Multicast Traffic Are Handled with OVSDB" on page 15.

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.

Default If you do not include the **ingress-node-replication** statement, one or more service nodes handle BUM traffic.

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

Related Documentation

- Configuring OVSDB-Managed VXLANs on page 42

interfaces (OVSDB)

Supported Platforms MX Series, QFX Series standalone switches

Syntax interfaces *interface-name*;

Hierarchy Level [edit protocols ovsdb]

Release Information Statement introduced in Junos OS Release 14.1R2.
Statement introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Statement introduced in Junos OS Release 14.2 for EX Series switches.

Description Specify the physical interfaces on a Juniper Networks device that you want the Open vSwitch Database (OVSDB) protocol to manage. Typically, the only interfaces that need to be managed by OVSDB are interfaces that are connected to physical servers.

Options *interface-name*—Name of the interface.

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

Related Documentation
- Configuring OVSDB-Managed VXLANs on page 42

maximum-backoff-duration

Supported Platforms MX Series, QFX Series standalone switches

Syntax maximum-backoff-duration *milliseconds*;

Hierarchy Level [edit protocols ovsdb controller]

Release Information Statement introduced in Junos OS Release 14.1R2.
Statement introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Statement introduced in Junos OS Release 14.2 for EX Series switches.

Description Specify (in milliseconds) how long a Juniper Networks device that supports the Open vSwitch Database (OVSDB) management protocol waits before it tries again to connect with a software-defined networking (SDN) controller after a previous attempt has failed.

Options *milliseconds*—Number of milliseconds a Juniper Networks device waits before it tries again to connect with an SDN controller.

Range: 1000 through 4,294,967,295

Default: 1000

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

Related Documentation
- Understanding How to Set Up Open vSwitch Database Connections Between Juniper Networks Devices and SDN Controllers on page 14

ovsdb

Supported Platforms MX Series, QFX Series standalone switches

Syntax

```
ovsdb {
  controller ip-address {
    inactivity-probe-duration milliseconds;
    maximum-backoff-duration milliseconds;
    protocol protocol {
      port number;
    }
  }
  interfaces interface-name;
  traceoptions {
    file <filename> <files number> <match regular-expression> <no-world-readable |
      world-readable> <size size>;
    flag flag;
    no-remote-trace;
  }
}
```

Hierarchy Level [edit protocols]

Release Information Statement introduced in Junos OS Release 14.1R2.
Statement introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Statement introduced in Junos OS Release 14.2 for EX Series switches.

Description Configure support for the Open vSwitch Database (OVSDB) management protocol on a Juniper Networks device.

The remaining statements are explained separately.

Default The OVSDB management protocol is disabled on Juniper Networks devices.

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

Related Documentation

- Understanding the OVSDB Protocol Running on Juniper Networks Devices on page 13
- Configuring OVSDB-Managed VXLANs on page 42

ovsdb-managed

Supported Platforms MX Series, QFX Series standalone switches

Syntax ovsdb-managed;

Hierarchy Level
[edit bridge-domains *bridge-domain-name* vxlan],
[edit routing-instances *routing-instance-name* bridge-domains *bridge-domain-name* vxlan],
[edit routing-instances *routing-instance-name* switch-options],
[edit routing-instances *routing-instance-name* vlans *vlan-name* vxlan],
[edit routing-instances *routing-instance-name* vxlan],
[edit switch-options]
[edit vlans *vlan-name* vxlan]

Release Information
Statement introduced in Junos OS Release 14.1R2.
Statement introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Statement introduced in Junos OS Release 14.2 for EX Series switches.

Description Disable a Juniper Networks device from learning about other Juniper Networks devices that function as hardware virtual tunnel endpoints (VTEPs) in a specified Virtual Extensible LAN (VXLAN) and the MAC addresses learned by the hardware VTEPs. Instead, the Juniper Networks device uses the Open vSwitch Database (OVSDB) management protocol to learn about the hardware VTEPs in the VXLAN and the MAC addresses learned by the hardware VTEPs.

The specified VXLAN must have a VXLAN network identifier (VNI) configured, using the **vni** statement in the **[edit bridge-domains *bridge-domain-name* vxlan]**, **[edit routing-instance *routing-instance-name* vxlan]**, or **[edit vlans *vlan-name* vxlan]** hierarchy.

Also, for a VMware NSX environment, this implementation of OVSDB uses the multicast scheme described in "Understanding How Layer 2 BUM and Layer 3 Routed Multicast Traffic Are Handled with OVSDB" on page 15. Therefore, specifying the **multicast-group** statement in the **[edit bridge-domains *bridge-domain-name* vxlan]**, **[edit routing-instances *routing-instance-name* vxlan]**, or **[edit vlans *vlan-name* vxlan]** hierarchy has no effect.

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

Related Documentation
- Configuring OVSDB-Managed VXLANs on page 42

port (OVSDB)

Supported Platforms MX Series, QFX Series standalone switches

Syntax port *number*;

Hierarchy Level [edit protocols ovsdb controller protocol]

Release Information Statement introduced in Junos OS Release 14.1R2.
Statement introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Statement introduced in Junos OS Release 14.2 for EX Series switches.

Description Specify the software-defined networking (SDN) controller port to which a Juniper Networks device that supports the Open vSwitch Database (OVSDB) management protocol connects.

Options *number*—Number of the SDN controller port.

Range: 1024 through 65,535

Default: 6632

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

Related Documentation

- Understanding How to Set Up Open vSwitch Database Connections Between Juniper Networks Devices and SDN Controllers on page 14

protocol (OVSDB)

Supported Platforms MX Series, QFX Series standalone switches

Syntax

```
protocol protocol {
  port number;
}
```

Hierarchy Level [edit protocols ovsdb controller]

Release Information Statement introduced in Junos OS Release 14.1R2.
Statement introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Statement introduced in Junos OS Release 14.2 for EX Series switches.

Description Configure the security protocol that protects the connection between a Juniper Networks device that supports the Open vSwitch Database (OVSDB) management protocol and a software-defined networking (SDN) controller.

The Secure Sockets Layer (SSL) connection requires a private key and certificates, which must be stored in the **/var/db/certs** directory of the Juniper Networks device. For more information, see "Creating and Installing an SSL Key and Certificate on a Juniper Networks Device for Connection with SDN Controllers" on page 36.

Options ***protocol***—Establish a secure connection to the SDN controller, using SSL or TCP.

NOTE: SSL is the only supported connection protocol.

Default: ssl

The remaining statement is explained separately.

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

Related Documentation

- Understanding How to Set Up Open vSwitch Database Connections Between Juniper Networks Devices and SDN Controllers on page 14

traceoptions (OVSDB)

Supported Platforms MX Series, QFX Series standalone switches

Syntax

```
traceoptions {
  file <filename> <files number> <match regular-expression> <no-world-readable |
    world-readable> <size size>;
  flag flag;
  no-remote-trace;
}
```

Hierarchy Level [edit protocols ovsdb]

Release Information Statement introduced in Junos OS Release 14.1R2.
Statement introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Statement introduced in Junos OS Release 14.2 for EX Series switches.

Description Define tracing operations for the Open vSwitch Database (OVSDB) management protocol, which is supported on Juniper Networks devices.

Default If you do not include this statement, OVSDB-specific tracing operations are not performed.

Options

file *filename*—Name of file in which the system places the output of the tracing operations. By default, the system places all files in the **/var/log** directory.

Default: **/var/log/vgd**

files *number*—(Optional) Maximum number of trace files. When a trace file reaches the size specified by the **size** option, the filename is appended with 0 and compressed. For example, a trace file named **trace-file.gz** would be renamed **trace-file.0.gz**. When **trace-file.0.gz** reaches the specified size, it is renamed **trace-file.1.gz** and its contents are compressed to **trace-file.0.gz**. This renaming scheme continues until the maximum number of trace files is reached. Then the oldest trace file is overwritten.

If you specify a maximum number of files, you also must specify a maximum file size with the **size** option and a filename.

Range: 2 through 1000 files

Default: 10 files

flag *flag*—Tracing operation to perform. You can include one or more of the following flags:

all—All OVSDB events.

configuration—OVSDB configuration events.

core—OVSDB core events.

function—OVSDB function events.

interface—OVSDB interface events.

l2-client—OVSDB Layer 2 client events.

netconf-client—(QFX5100 switches only) Events for the automatic configuration of Virtual Extensible LANs (VXLANs).

ovs-client—OVSDB client events.

match *regular-expression*—(Optional) Only log lines that match the regular expression.

no-remote-trace—(Optional) Disable tracing and logging operations that track normal operations, error conditions, and packets that are generated by or passed through the Juniper Networks device.

no-world-readable—Restrict access to the trace files to the owner.

Default: no-world-readable

size *size*—(Optional) Maximum size of each trace file in bytes, kilobytes (KB), megabytes (MB), or gigabytes (GB). If you do not specify a unit, the default is bytes. If you specify a maximum file size, you also must specify a maximum number of trace files by using the **files** option and a filename by using the **file** option.

Syntax: *size* to specify bytes, *size*k to specify KB, *size*m to specify MB, or *size*g to specify GB.

Range: 10,240 through 1,073,741,824 bytes

Default: 128 KB

world-readable—Enable any user to access the trace files.

Required Privilege Level

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

Related Documentation

- Example: Setting Up Inter-VXLAN Routing and OVSDB Connections in a Data Center on page 63

CHAPTER 4

VXLAN Configuration Statements

decapsulate-accept-inner-vlan

Supported Platforms QFX Series standalone switches

Syntax decapsulate-accept-inner-vlan

Hierarchy Level [edit protocols l2-learning]

Release Information Statement modified in Junos OS 14.1X53 for the QFX Series.

Description Configure the switch to de-encapsulate and accept original VLAN tags in VXLAN packets.

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

Related Documentation
- Understanding VXLANs on page 3
- **encapsulate-inner-vlan on page 132**

encapsulate-inner-vlan

Supported Platforms QFX Series standalone switches

Syntax encapsulate-inner-vlan

Hierarchy Level [edit vlans *VLAN* vxlan]

Release Information Statement introduced in Junos OS Release 14.1X53-D10.

Description Configure the switch to preserve the original VLAN tag (in the inner Ethernet packet) when performing VXLAN encapsulation.

Default The original tag is dropped when the packet is encapsulated.

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

Related Documentation

- Understanding VXLANs on page 3
- Configuring VXLANs on a QFX5100 Switch on page 45
- Examples: Manually Configuring VXLANs on QFX Series Switches on page 81
- **decapsulate-accept-inner-vlan on page 131**

multicast-group

Supported Platforms QFX Series standalone switches

Syntax multicast-group

Hierarchy Level [edit vlans *VLAN* vxlan]

Release Information Statement introduced in Junos OS Release 14.1X53-D10.

Description Assign a multicast group address to a VXLAN. All members of a VXLAN must use the same multicast group address

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

Related Documentation

- Understanding VXLANs on page 3
- Configuring VXLANs on a QFX5100 Switch on page 45
- Examples: Manually Configuring VXLANs on QFX Series Switches on page 81

ovsdb-managed

Supported Platforms MX Series, QFX Series standalone switches

Syntax ovsdb-managed;

Hierarchy Level
[edit bridge-domains *bridge-domain-name* vxlan],
[edit routing-instances *routing-instance-name* bridge-domains *bridge-domain-name* vxlan],
[edit routing-instances *routing-instance-name* switch-options],
[edit routing-instances *routing-instance-name* vlans *vlan-name* vxlan],
[edit routing-instances *routing-instance-name* vxlan],
[edit switch-options]
[edit vlans *vlan-name* vxlan]

Release Information
Statement introduced in Junos OS Release 14.1R2.
Statement introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Statement introduced in Junos OS Release 14.2 for EX Series switches.

Description Disable a Juniper Networks device from learning about other Juniper Networks devices that function as hardware virtual tunnel endpoints (VTEPs) in a specified Virtual Extensible LAN (VXLAN) and the MAC addresses learned by the hardware VTEPs. Instead, the Juniper Networks device uses the Open vSwitch Database (OVSDB) management protocol to learn about the hardware VTEPs in the VXLAN and the MAC addresses learned by the hardware VTEPs.

The specified VXLAN must have a VXLAN network identifier (VNI) configured, using the **vni** statement in the **[edit bridge-domains *bridge-domain-name* vxlan]**, **[edit routing-instance *routing-instance-name* vxlan]**, or **[edit vlans *vlan-name* vxlan]** hierarchy.

Also, for a VMware NSX environment, this implementation of OVSDB uses the multicast scheme described in "Understanding How Layer 2 BUM and Layer 3 Routed Multicast Traffic Are Handled with OVSDB" on page 15. Therefore, specifying the **multicast-group** statement in the **[edit bridge-domains *bridge-domain-name* vxlan]**, **[edit routing-instances *routing-instance-name* vxlan]**, or **[edit vlans *vlan-name* vxlan]** hierarchy has no effect.

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

Related Documentation
- Configuring OVSDB-Managed VXLANs on page 42

unreachable-vtep-aging-timer

Supported Platforms QFX Series standalone switches

Syntax unreachable-vtep-aging-timer [300–1800]

Hierarchy Level [edit vlans *VLAN* vxlan]

Release Information Statement introduced in Junos OS Release 14.1X53-D10.

Description Configure the system to age out the address for the remote VTEP 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.

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

Related Documentation

- Understanding VXLANs on page 3
- Configuring VXLANs on a QFX5100 Switch on page 45
- Examples: Manually Configuring VXLANs on QFX Series Switches on page 81

vni

Supported Platforms QFX Series standalone switches

Syntax vni [1–16777214]

Hierarchy Level [edit vlans *VLAN* vxlan]

Release Information Statement introduced in Junos OS Release 14.1X53-D10.

Description Assign a numeric value to identify a VXLAN. All members of a VXLAN must use the same VNI.

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

Related Documentation

- Understanding VXLANs on page 3
- Configuring VXLANs on a QFX5100 Switch on page 45
- Examples: Manually Configuring VXLANs on QFX Series Switches on page 81

vtep-source-interface

Supported Platforms QFX Series standalone switches

Syntax vtep-source-interface *logical-interface*;

Hierarchy Level [edit switch-options,

Release Information Statement introduced in Junos OS Release 14.1X53-D10.

Description Configure a source interface for a VXLAN tunnel. You must provide the name of a logical interface configured on the loopback interface.

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

Related Documentation

- Understanding VXLANs on page 3
- Configuring VXLANs on a QFX5100 Switch on page 45
- Examples: Manually Configuring VXLANs on QFX Series Switches on page 81

vxlan

Supported Platforms QFX Series standalone switches

Syntax

```
vxlan {
  encapsulate-inner-vlan
  ingress-node-replication
  multicast-group
  ovsdb-managed
  unreachable-vtep-aging-timer
  vni
  }
}
```

Hierarchy Level [edit vlans]

Release Information Statement introduced in Junos OS Release 14.1X53-D10.
ingress-node-replication option added for QFX Series switches in Junos OS Release 14.1X53-D30.

Description

Options The remaining statements are explained separately.

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

Related Documentation

- Understanding VXLANs on page 3
- Configuring VXLANs on a QFX5100 Switch on page 45
- Examples: Manually Configuring VXLANs on QFX Series Switches on page 81

CHAPTER 5

OVSDB Monitoring Commands

show bfd session

Supported Platforms ACX Series, EX Series, J Series, M Series, MX Series, PTX Series, QFabric System, QFX Series standalone switches, T Series

List of Syntax
Syntax on page 138
Syntax (EX Series Switch and QFX Series) on page 138

Syntax

```
show bfd session
<brief | detail | extensive | summary>
<address address>
<client rsvp-oam (brief | detail | extensive | summary) | vpls-oam (brief | detail | extensive |
  instance instance-name | summary)>
<discriminator discriminator>
<logical-system (all | logical-system-name)>
<prefix address>
```

Syntax (EX Series Switch and QFX Series)

```
show bfd session
<brief | detail | extensive | summary>
<address address>
<client rsvp-oam (brief | detail | extensive | summary) | vpls-oam (brief | detail | extensive |
  instance instance-name | summary)>
<discriminator discriminator>
<prefix address>
```

Release Information
Command introduced before Junos OS Release 7.4.
Options **discriminator** and **address** introduced in Junos OS Release 8.2.
Option **prefix** introduced in Junos OS Release 9.0.
Command introduced in Junos OS Release 12.1 for the QFX Series.
Option **client** introduced in Junos OS Release 12.3R3.

Description Display information about active Bidirectional Forwarding Detection (BFD) sessions.

Options

none—(Same as **brief**) Display information about active BFD sessions.

brief | detail | extensive | summary—(Optional) Display the specified level of output.

address *address*—(Optional) Display information about the BFD session for the specified neighbor address.

client rsvp-oam
(brief | detail | extensive | summary)
| vpls-oam
(brief | detail | extensive | instance *instance-name* | summary)—(Optional) Display information about RSVP-OAM or VPLS-OAM BFD sessions in the specified level of output. For VPLS-OAM, display the specified level of output or display information about all of the BFD sessions for the specified VPLS routing instance.

discriminator *discriminator*—(Optional) Display information about the BFD session using the specified local discriminator.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

prefix *address*—(Optional) Display information about all of the BFD sessions for the specified LDP forwarding equivalence class (FEC).

Required Privilege Level view

Related Documentation

- *clear bfd session*
- *Examples: Configuring BFD for Static Routes*
- *Example: Configuring BFD for OSPF*
- *Example: Configuring BFD for BGP*
- *Configuring PIM and the Bidirectional Forwarding Detection (BFD) Protocol*
- *Example: Configuring BFD for IS-IS*

List of Sample Output

show bfd session on page 142
show bfd session brief on page 143
show bfd session detail on page 143
show bfd session detail (with Authentication) on page 143
show bfd session address extensive on page 143
show bfd session client rsvp-oam on page 144
show bfd session client vpls-oam summary on page 144
show bfd session client vpls-oam instance instance-name on page 144
show bfd session extensive on page 144
show bfd session extensive (with Authentication) on page 145
show bfd session summary on page 145

Output Fields Table 17 on page 139 describes the output fields for the **show bfd session** command. Output fields are listed in the approximate order in which they appear.

Table 17: show bfd session Output Fields

Field Name	Field Description	Level of Output
Address	Address on which the BFD session is active.	**brief detail extensive** none
State	State of the BFD session: **Up**, **Down**, **Init** (initializing), or **Failing**.	**brief detail extensive** none
Interface	Interface on which the BFD session is active.	**brief detail extensive** none
Detect Time	Negotiated time interval, in seconds, used to detect BFD control packets.	**brief detail extensive** none
Transmit Interval	Time interval, in seconds, used by the transmitting system to send BFD control packets.	**brief detail extensive** none

Table 17: show bfd session Output Fields *(continued)*

Field Name	Field Description	Level of Output
Multiplier	Negotiated multiplier by which the time interval is multiplied to determine the detection time for the transmitting system.	**detail extensive**
Session up time	Length of time a BFD session has been established.	**detail extensive**
Client	Protocol or process for which the BFD session is active: **ISIS**, **OSPF**, **Static**, or **VGD**.	**detail extensive**
TX interval	Time interval, in seconds, used by the host system to transmit BFD control packets.	**brief detail extensive** none
RX interval	Time interval, in seconds, used by the host system to receive BFD control packets.	**brief detail extensive** none
Authenticate	Indicates that BFD authentication is configured.	**detail extensive**
keychain	Name of the security authentication keychain being used by a specific client. BFD authentication information for a client is provided in a single line and includes the **keychain**, **algo**, and **mode** parameters. Multiple clients can be configured in a BFD session.	**extensive**
algo	BFD authentication algorithm being used for a specific client: **keyed-md5**, **keyed-sha-1**, **meticulous-keyed-md5**, **meticulous-keyed-sha-1**, or **simple-password**. BFD authentication information for a client is provided in a single line and includes the **keychain**, **algo**, and **mode** parameters. Multiple clients can be configured in a BFD session.	**extensive**
mode	Level of BFD authentication enforcement being used by a specific client: **strict** or **loose**. Strict enforcement indicates that authentication is configured at both ends of the session (the default). Loose enforcement indicates that one end of the session might not be authenticated. BFD authentication information for a client is provided in a single line and includes the **keychain**, **algo**, and **mode** parameters. Multiple clients can be configured in a BFD session.	**extensive**
Local diagnostic	Local diagnostic information about failing BFD sessions.	**detail extensive**
Remote diagnostic	Remote diagnostic information about failing BFD sessions.	**detail extensive**
Remote state	Status report on whether the remote system's BFD packets have been received and whether the remote system is receiving transmitted control packets.	**detail extensive**
Version	BFD version: **0** or **1**.	**extensive**
Replicated	The **replicated** flag appears when nonstop routing or graceful Routing Engine switchover is configured and the BFD session has been replicated to the backup Routing Engine.	**detail extensive**

Table 17: show bfd session Output Fields *(continued)*

Field Name	Field Description	Level of Output
Min async interval	Minimum amount of time, in seconds, between asynchronous control packet transmissions across the BFD session.	**extensive**
Min slow interval	Minimum amount of time, in seconds, between synchronous control packet transmissions across the BFD session.	**extensive**
Adaptive async TX interval	Transmission interval being used because of adaptation.	**extensive**
RX interval	Minimum required receive interval.	**extensive**
Local min TX interval	Minimum amount of time, in seconds, between control packet transmissions on the local system.	**extensive**
Local min RX interval	Minimum amount of time, in seconds, between control packet detections on the local system.	**extensive**
Remote min TX interval	Minimum amount of time, in seconds, between control packet transmissions on the remote system.	**extensive**
Remote min RX interval	Minimum amount of time, in seconds, between control packet detections on the remote system.	**extensive**
Threshold transmission interval	Threshold for notification if the transmission interval increases.	**extensive**
Threshold for detection time	Threshold for notification if the detection time increases.	**extensive**
Local discriminator	Authentication code used by the local system to identify that BFD session.	**extensive**
Remote discriminator	Authentication code used by the remote system to identify that BFD session.	**extensive**
Echo mode	Information about the state of echo transmissions on the BFD session.	**extensive**
Prefix	LDP FEC address associated with the BFD session.	All levels
Egress, Destination	Displays the LDP FEC destination address. This field is displayed only on a router at the egress of an LDP FEC, where the BFD session has an LDP Operation, Administration, and Maintenance (OAM) client.	All levels
Remote is control-plane independent	The BFD session on the remote peer is running on its Packet Forwarding Engine. In this case, when the remote node undergoes a graceful restart, the local peer can help the remote peer with the graceful restart. The following BFD sessions are not distributed to the Packet Forwarding Engine: tunnel-encapsulated sessions, and sessions over integrated routing and bridging (IRB) interfaces.	**extensive**

Table 17: show bfd session Output Fields *(continued)*

Field Name	Field Description	Level of Output
Authentication	Summary status of BFD authentication: • **status**—**enabled/active** indicates authentication is configured and active. **enabled/inactive** indicates authentication is configured but not active. This only occurs when the remote end of the session does not support authentication and loose checking is configured. • **keychain**—Name of the security authentication keychain associated with the specified BFD session. • **algo**—BFD authentication algorithm being used: **keyed-md5**, **keyed-sha-1**, **meticulous-keyed-md5**, **meticulous-keyed-sha-1**, or **simple-password**. • **mode**—Level of BFD authentication enforcement: **strict** or **loose**. Strict enforcement indicates authentication is configured at both ends of the session (the default). Loose enforcement indicates that one end of the session might not be authenticated. This information is only shown if BFD authentication is configured.	**extensive**
Session ID	The BFD session ID number that represents the protection using MPLS fast reroute (FRR) and loop-free alternate (LFA).	**detail extensive**
sessions	Total number of active BFD sessions.	All levels
clients	Total number of clients that are hosting active BFD sessions.	All levels
Cumulative transmit rate	Total number of BFD control packets transmitted per second on all active sessions.	All levels
Cumulative receive rate	Total number of BFD control packets received per second on all active sessions.	All levels
Multi-hop, min-recv-TTL	Minimum time to live (TTL) accepted if the session is configured for multihop.	**extensive**
route table	Route table used if the session is configured for multihop.	**extensive**
local address	Local address of the source used if the session is configured for multihop. The source IP address for outgoing BFD packets from the egress side of an MPLS BFD session is based on the outgoing interface IP address.	**extensive**

Sample Output

show bfd session

```
user@host> show bfd session
                                                    Transmit
Address              State     Interface     Detect Time  Interval  Multiplier
10.9.1.33            Up        so-7/1/0.0          0.600     0.200      3
10.9.1.29            Up        ge-4/0/0.0          0.600     0.200      3

2 sessions, 2 clients
Cumulative transmit rate 10.0 pps, cumulative receive rate 10.0 pps
```

show bfd session brief

The output for the **show bfd session brief** command is identical to that for the **show bfd session** command. For sample output, see show bfd session on page 142.

show bfd session detail

```
user@host> show bfd session detail
                                                    Transmit
Address              State     Interface     Detect Time  Interval  Multiplier
10.9.1.33            Up        so-7/1/0.0          0.600     0.200      3
  Client OSPF, TX interval 0.200, RX interval 0.200, multiplier 3
  Session up time 3d 00:34
  Local diagnostic None, remote diagnostic None
  Remote state Up, version 1
  Replicated
10.9.1.29            Up        ge-4/0/0.0          0.600     0.200      3
  Client ISIS L2, TX interval 0.200, RX interval 0.200, multiplier 3
  Session up time 3d 00:29, previous down time 00:00:01
  Local diagnostic NbrSignal, remote diagnostic AdminDown
  Remote state Up, version 1

2 sessions, 2 clients
Cumulative transmit rate 10.0 pps, cumulative receive rate 10.0 pps
```

show bfd session detail (with Authentication)

```
user@host> show bfd session detail
                                                    Transmit
Address              State     Interface     Detect Time  Interval  Multiplier
10.9.1.33            Up        so-7/1/0.0          0.600     0.200      3
  Client OSPF, TX interval 0.200, RX interval 0.200, multiplier 3, Authenticate
  Session up time 3d 00:34
  Local diagnostic None, remote diagnostic None
  Remote state Up, version 1
  Replicated
10.9.1.29            Up        ge-4/0/0.0          0.600     0.200      3
  Client ISIS L2, TX interval 0.200, RX interval 0.200, multiplier 3
  Session up time 3d 00:29, previous down time 00:00:01
  Local diagnostic NbrSignal, remote diagnostic AdminDown
  Remote state Up, version 1

2 sessions, 2 clients
Cumulative transmit rate 10.0 pps, cumulative receive rate 10.0 pps
```

show bfd session address extensive

```
user@host> show bfd session 10.255.245.212 extensive
                                                    Transmit
Address              State     Interface     Detect Time  Interval  Multiplier
10.255.245.212       Up                          1.200     0.400      3
  Client Static, TX interval 0.400, RX interval 0.400, multiplier 3
  Session up time 00:17:03, previous down time 00:00:14
  Local diagnostic CtlExpire, remote diagnostic NbrSignal
  Remote state Up, version 1
  Replicated
  Min async interval 0.400, min slow interval 1.000
  Adaptive async tx interval 0.400, rx interval 0.400
  Local min tx interval 0.400, min rx interval 0.400, multiplier 3
  Remote min tx interval 0.400, min rx interval 0.400, multiplier 3
```

```
  Threshold transmission interval 0.000, Threshold for detection time 0.000
  Local discriminator 6, remote discriminator 16
  Echo mode disabled/inactive
  Multi-hop, min-recv-TTL 255, route-table 0, local-address 10.255.245.205


1 sessions, 1 clients
Cumulative transmit rate 2.5 pps, cumulative receive rate 2.5 pps
```

show bfd session client rsvp-oam

```
user@host> show bfd session client rsvp-oam
                                                 Detect   Transmit
 Address                 State     Interface     Time     Interval  Multiplier

 192.168.0.223           Up                      540.000   180.000      3

 1 Up sessions, 0 Down sessions
 1 sessions, 1 clients
 Cumulative transmit rate 0.0 pps, cumulative receive rate 0.0 pps
```

show bfd session client vpls-oam summary

```
user@host> show bfd session client vpls-oam summary
1 Up sessions, 1 Down sessions
 2 sessions, 2 clients
 Cumulative transmit rate 2.0 pps, cumulative receive rate 1.0 pps
```

show bfd session client vpls-oam instance instance-name

```
user@host> show bfd session client vpls-oam instance vpls
                                                 Detect   Transmit
 Address                 State     Interface     Time     Interval  Multiplier

 127.0.0.1               Up        ae9.0         3.000     1.000        3

 1 Up Sessions, 0 Down Sessions
 1 sessions, 1 clients
 Cumulative transmit rate 1.0 pps, cumulative receive rate 1.0 pps
```

show bfd session extensive

```
user@host> show bfd session extensive
10.31.1.2               Up        ge-2/1/8.0     0.030     0.010        3
 Client OSPF realm ospf-v2 Area 0.0.0.0, TX interval 0.010, RX interval 0.010
 Session up time 00:10:13
 Local diagnostic None, remote diagnostic None
 Remote state Up, version 1
 Replicated
 Min async interval 0.010, min slow interval 1.000
 Adaptive async TX interval 0.010, RX interval 0.010
 Local min TX interval 0.010, minimum RX interval 0.010, multiplier 3
 Remote min TX interval 0.010, min RX interval 0.010, multiplier 3
 Local discriminator 12, remote discriminator 4
 Echo mode disabled/inactive
 Remote is control-plane independent
  Session ID: 0x201
  Micro-BFD Session
                                                 Detect   Transmit
Address                 State     Interface     Time     Interval  Multiplier
10.31.2.2               Up        ge-2/1/4.0     0.030     0.010        3
 Client OSPF realm ospf-v2 Area 0.0.0.0, TX interval 0.010, RX interval 0.010
```

```
 Session up time 00:10:14
 Local diagnostic None, remote diagnostic NbrSignal
 Remote state Up, version 1
 Replicated
 Min async interval 0.010, min slow interval 1.000
 Adaptive async TX interval 0.010, RX interval 0.010
 Local min TX interval 0.010, minimum RX interval 0.010, multiplier 3
 Remote min TX interval 0.010, min RX interval 0.010, multiplier 3
 Local discriminator 13, remote discriminator 5
 Echo mode disabled/inactive
 Remote is control-plane independent
  Session ID: 0x202

2 sessions, 2 clients
Cumulative transmit rate 200.0 pps, cumulative receive rate 200.0 pps
```

show bfd session extensive (with Authentication)

```
user@host> show bfd session extensive
                                                  Detect   Transmit
Address                 State     Interface       Time     Interval  Multiplier
192.168.208.26          Up        so-1/0/0.0      2.400     0.800        10
 Client Static, TX interval 0.600, RX interval 0.600, Authenticate
        keychain bfd, algo keyed-md5, mode loose
 Session up time 00:18:07
 Local diagnostic None, remote diagnostic NbrSignal
 Remote state Up, version 1
 Replicated
 Min async interval 0.600, min slow interval 1.000
 Adaptive async TX interval 0.600, RX interval 0.600
 Local min TX interval 0.600, minimum RX interval 0.600, multiplier 10
 Remote min TX interval 0.800, min RX interval 0.800, multiplier 3
 Local discriminator 2, remote discriminator 3
 Echo mode disabled/inactive
 Authentication enabled/active, keychain bfd, algo keyed-md5, mode loose


1 sessions, 1 clients
Cumulative transmit rate 1.2 pps, cumulative receive rate 1.2 pps
```

show bfd session summary

```
user@host> show bfd session summary
2 sessions, 2 clients
Cumulative transmit rate 10.0 pps, cumulative receive rate 10.0 pps
```

clear ovsdb commit failures

Supported Platforms QFX Series standalone switches

Syntax clear ovsdb commit failures
<transaction-id>

Release Information Command introduced in Junos OS Release 14.1X53-D26 for QFX Series switches.

Description Remove a transaction from a queue maintained by a Juniper Networks switch that supports the Open vSwitch Database (OVSDB) management protocol and Virtual Extensible LANs (VXLANs). The transaction includes OVSDB-managed VXLANs and associated logical interfaces that the Juniper Networks switch automatically configured and tried to commit but was unable to because of an issue with one or more of the configurations. In addition to removing the transaction, entering the **clear ovsdb commit failures** command causes the Juniper Networks switch to automatically retry committing all configurations in the transaction.

If there is an issue with one or more of the configurations in a transaction, this causes all configurations in the transaction, even the ones that are correctly configured, to remain uncommitted and in the queue until you troubleshoot and resolve the configuration issue(s).

You can display an erroneous transaction by entering the **show ovsdb commit failures** command. In the output that appears, you must determine which configuration(s) are erroneous and therefore prevent the Juniper Networks switch from committing the 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 an automatically configured VXLAN and in a VXLAN that was previously configured using the Junos OS CLI.

To monitor for issues with automatically configured OVSDB-managed VXLANs and their associated interfaces, we recommend checking for system log messages and traceoptions files for OVSDB.

After resolving the error(s), enter the **clear ovsdb commit failures** command to remove the transaction from the queue and retry committing all configurations in the transaction.

NOTE: While an erroneous transaction exists in the queue, the Juniper Networks switch cannot commit the configurations of additional VXLANs and their associated logical interfaces. The commitment of these VXLANs and logical interfaces remain in a pending state until all VXLAN and logical interface configurations in the erroneous transaction are resolved and successfully committed.

Options **none**—Remove the transaction that currently appears in the **show ovsdb commit failures** command output, and retry committing all configurations in the transaction.

transaction-id—Remove the transaction with the specified numerical ID, and retry committing the configurations in the transaction.

Required Privilege Level clear

Related Documentation
- show ovsdb commit failures on page 148

List of Sample Output
clear ovsdb commit failures on page 147
clear ovsdb commit failures (Specific Transaction) on page 147

Sample Output

clear ovsdb commit failures

```
user@host> clear ovsdb commit failures
```

clear ovsdb commit failures (Specific Transaction)

```
user@host> clear ovsdb commit failures 1
```

show ovsdb commit failures

Supported Platforms QFX Series standalone switches

Syntax show ovsdb commit failures
<transaction-id>

Release Information Command introduced in Junos OS Release 14.1X53-D26 for QFX Series switches.

Description Display configurations of Open vSwitch Database (OVSDB)-managed Virtual Extensible LANs (VXLANs) and associated logical interfaces that the Juniper Networks switch automatically configured but was unable to commit.

For each OVSDB-managed VXLAN and associated logical interface that you plan to implement in a Junos OS environment, you must configure equivalent entities in NSX Manager or in the NSX API for an NSX environment, or in the Contrail Web user interface for a Contrail environment. The software-defined networking (SDN) controller pushes these configurations to the connected Juniper Networks switch by way of the OVSDB schema for physical devices. After the Juniper Networks switch receives these configurations, it automatically configures a Junos OS-equivalent VXLAN and associated logical interface, and attempts to commit the configurations.

During the commitment of the automatic configurations, If there is an issue with one or more of the configurations, all configurations in the transaction, even the ones that are correctly configured, remain uncommitted and are saved in a queue. All configurations in the transaction remain uncommitted and in the queue until you troubleshoot and resolve the configuration issues. After you resolve the configuration issues, you must use the clear ovsdb commit failures command to remove the transaction from the queue and retry committing the configurations.

NOTE: **While an erroneous transaction exists in the queue, the Juniper Networks switch cannot commit the automatic configurations of additional VXLANs and their associated logical interfaces. The commitment of these VXLANs and logical interfaces remain in a pending state until all VXLAN and logical interface configurations in the erroneous transaction are resolved and successfully committed.**

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 automatically configured VXLAN and in a VXLAN that was previously configured using the Junos OS CLI.

To monitor for issues with automatically configured OVSDB-managed VXLANs and their associated interfaces, we recommend checking for system log messages and traceoptions files for OVSDB.

Options **none**—Display information about an erroneous transaction.

transaction-id—Display information about the transaction with the specified numerical ID.

Required Privilege Level admin

Related Documentation
- Understanding Automatically Configured VXLANs in an OVSDB Environment on page 20
- traceoptions (OVSDB) on page 128

List of Sample Output
show ovsdb commit failures on page 149
show ovsdb commit failure (Specific Transaction) on page 149

Output Fields Table 18 on page 149 lists the output fields for the **show ovsdb commit failures** command. Output fields are listed in the approximate order in which they appear.

Table 18: show ovsdb commit failures Output Fields

Field Name	Field Description
Txn ID	ID assigned to a transaction by the Juniper Networks switch.
Logical-switch	Name of the VXLAN that the Juniper Networks switch automatically configured but was unable to commit the configuration of.
Port	Name of an OVSDB-managed physical interface that is associated with the VXLAN.
VLAN ID	ID that is assigned to the VXLAN.

Sample Output

show ovsdb commit failures

```
user@host> show ovsdb commit failures
Txn ID      Logical-switch                          Port              VLAN ID
1           28805c1d-0122-495d-85df-19abd647d772
1                                                   xe-0/0/5:0        1016
1           9acc24b3-7b0a-4c2e-b572-3370c3e1acff
1                                                   xe-0/0/5:0        1017
...
```

show ovsdb commit failure (Specific Transaction)

```
user@host> how ovsdb commit failures 1
Txn ID      Logical-switch                          Port              VLAN ID
1           28805c1d-0122-495d-85df-19abd647d772
1                                                   xe-0/0/5:0        1016
1           9acc24b3-7b0a-4c2e-b572-3370c3e1acff
1                                                   xe-0/0/5:0        1017
...
```

show ovsdb controller

Supported Platforms MX Series, QFX Series standalone switches

Syntax
show ovsdb controller
<address *ip-address*>

Release Information
Command introduced in Junos OS Release 14.1R2.
Command introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Command introduced in Junos OS Release 14.2 for EX Series switches.

Description Display information and connection status for software-defined networking (SDN) controllers to which the Juniper Networks device is connected.

Options
none—Display information about all SDN controllers to which the Juniper Networks device is connected.

address *ip-address*—Display information about the SDN controller at the specified IP address.

Required Privilege Level admin

Related Documentation
- Understanding How to Set Up Open vSwitch Database Connections Between Juniper Networks Devices and SDN Controllers on page 14

List of Sample Output
show ovsdb controller on page 151
show ovsdb controller address on page 151

Output Fields Table 19 on page 150 lists the output fields for the **show ovsdb controller** command. Output fields are listed in the approximate order in which they appear.

Table 19: show ovsdb controller Output Fields

Field Name	Field Description
Controller IP address	IP address of the SDN controller to which the Juniper Networks device is connected.
Controller protocol	Protocol used by the Juniper Networks device to initiate the connection.
Controller port	Port to which the Juniper Networks device is connected.
Controller connection	State of the connection with the SDN controller.
Controller seconds-since-connect	Number of seconds since the connection with the SDN controller was established.
Controller seconds-since-disconnect	Number of seconds since the connection with the SDN controller was dropped.
Controller connection status	Status of the connection with the SDN controller.

Sample Output

show ovsdb controller

```
user@host> show ovsdb controller
VTEP controller information:
Controller IP address: 10.168.66.189
Controller protocol: ssl
Controller port: 6632
Controller connection: up
Controller seconds-since-connect: 56290
Controller seconds-since-disconnect: 0
Controller connection status: active

Controller IP address: 10.168.181.54
Controller protocol: ssl
Controller port: 6632
Controller connection: up
Controller seconds-since-connect: 56292
Controller seconds-since-disconnect: 0
Controller connection status: active

Controller IP address: 10.168.182.45
Controller protocol: ssl
Controller port: 6632
Controller connection: up
Controller seconds-since-connect: 56292
Controller seconds-since-disconnect: 0
Controller connection status: active
```

show ovsdb controller address

```
user@host> show ovsdb controller address 10.168.182.45
VTEP controller information:
Controller IP address: 192.168.182.45
Controller protocol: ssl
Controller port: 6632
Controller connection: up
Controller seconds-since-connect: 56347
Controller seconds-since-disconnect: 0
Controller connection status: active
```

show ovsdb interface

Supported Platforms MX Series, QFX Series standalone switches

Syntax
show ovsdb interface
<*interface-name*>

Release Information
Command introduced in Junos OS Release 14.1R2.
Command introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.

Description Display information about Open vSwitch Database (OVSDB)-managed interfaces configured by using the **interfaces** ***interface-name*** statement in the [**edit protocols ovsdb**] hierarchy.

Options
none—Display information about all OVSDB-managed interfaces.

interface-name—Display information about the specified OVSDB-managed interface.

Required Privilege Level admin

Related Documentation
- Configuring OVSDB-Managed VXLANs on page 42
- show ovsdb statistics interface on page 161

List of Sample Output
show ovsdb interface on page 152
show ovsdb (Specific Interface) on page 153

Output Fields Table 20 on page 152 lists the output fields for the **show ovsdb interface** command. Output fields are listed in the approximate order in which they appear.

Table 20: show ovsdb interface Output Fields

Field Name	Field Description
Interface	Name of interface.
VLAN ID	ID of Virtual Extensible LAN (VXLAN) with which the interface is associated. NOTE: This field is not supported by MX Series routers.
Bridge domain or VLAN	Bridge domain or VLAN under which the VXLAN is created. NOTE: This field is not supported by MX Series routers.

Sample Output

show ovsdb interface

```
user@host> show ovsdb interface
Interface              VLAN ID         Bridge-domain
ge-7/0/9.0
ge-7/0/9.1
```

```
irb.11
irb.12
irb.2
irb.3
xe-10/3/0.0
xe-10/3/0.1
```

show ovsdb (Specific Interface)

```
user@host> show ovsdb interface ge-7/0/9.0
Interface               VLAN ID         Bridge-domain
ge-7/0/9.0
```

show ovsdb logical-switch

Supported Platforms MX Series, QFX Series standalone switches

Syntax
show ovsdb logical-switch
<logical-switch-name>

Release Information
Command introduced in Junos OS Release 14.1R2.
Command introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Command introduced in Junos OS Release 14.2 for EX Series switches.

Description

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 was pushed to the OVSDB schema.

Display information about logical switches and the corresponding Virtual Extensible LANs (VXLANs), which were configured on the Juniper Networks device.

In the command output, each logical switch is identified by a universally unique identifier (UUID), which in the context of this command, is also known as a logical switch name.

The **show ovsdb logical-switch** command displays the state of the logical switch (**Flags**), which can be one of the following:

Created by Controller—A logical switch is configured. However, a corresponding VXLAN is not yet configured. In this state, the logical switch and corresponding VXLAN are not yet operational.

Created by L2ALD—A VXLAN is configured. However, a corresponding logical switch is not yet configured. In this state, the logical switch and corresponding VXLAN are not yet operational.

Created by both—A logical switch and a corresponding VXLAN are configured. In this state, the logical switch and corresponding VXLAN are operational.

Tunnel key mismatch—The VNIs specified in the logical switch and corresponding VXLAN configurations do not match. In this state, the logical switch and corresponding VXLAN are not yet operational.

Options

none—Display information about all logical switches that are present in the OVSDB schema for physical devices.

logical-switch-name—Display information about the specified logical switch.

Required Privilege Level admin

Related Documentation
- Troubleshooting a Nonoperational Logical Switch and Corresponding Junos OS OVSDB-Managed VXLAN on page 113

List of Sample Output
show ovsdb logical-switch on page 155
show ovsdb logical-switch (Specific Logical Switch) on page 156

Output Fields Table 21 on page 155 lists the output fields for the **show ovsdb logical-switch** command. Output fields are listed in the approximate order in which they appear.

Table 21: show ovsdb logical-switch Output Fields

Field Name	Field Description
Logical Switch Name	UUID that is automatically generated and assigned to the logical switch. When you configure the corresponding VXLAN in the Junos OS CLI, you must specify the same UUID as the VXLAN name.
Flags	State of the logical switch. For possible states, see the Description section of this topic.
VNI	VNI that is configured for the logical switch and corresponding VXLAN.
Num of Remote MAC	The total number of remote MAC addresses associated with the logical switch. These addresses are learned by software and hardware virtual tunnel endpoints (VTEPs).
Num of Local MAC	The total number of local MAC addresses associated with the logical switch. *Local MAC addresses* are addresses learned on the local physical ports.

Sample Output

show ovsdb logical-switch

```
user@host> show ovsdb logical-switch
Logical switch information:
Logical Switch Name: 24a76aff-7e61-4520-a78d-3eca26ad7510
Flags: Created by both
VNI: 3
Num of Remote MAC: 13
Num of Local MAC: 12
Logical Switch Name: 9b4f880e-dac8-4612-a832-97ad9dec270f
Flags: Created by Controller
VNI: 50
Num of Remote MAC: 0
Num of Local MAC: 0
Logical Switch Name: bc0da2da-6c16-44bf-b655-442484294ded
Flags: Created by Controller
VNI: 51
Num of Remote MAC: 0
Num of Local MAC: 0
```

show ovsdb logical-switch (Specific Logical Switch)

```
user@host> show ovsdb logical-switch 24a76aff-7e61-4520-a78d-3eca26ad7510
Logical switch information:
Logical Switch Name: 24a76aff-7e61-4520-a78d-3eca26ad7510
Flags: Created by both
VNI: 3
Num of Remote MAC: 13
Num of Local MAC: 12
```

show ovsdb mac

Supported Platforms MX Series, QFX Series standalone switches

Syntax
show ovsdb mac
<address *mac-address*>
<local>
<logical-switch *logical-switch-uuid*>
<multicast>
<remote>
<unicast>

Release Information
Command introduced in Junos OS Release 14.1R2.
Command introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Command introduced in Junos OS Release 14.2 for EX Series switches.

Description Display MAC addresses, as well as information about the MAC addresses, learned by a Juniper Networks device that functions as a hardware virtual tunnel endpoint (VTEP). Using the Open vSwitch Database (OVSDB) management protocol, this hardware VTEP can learn about MAC addresses directly or from other software or hardware VTEPs. The MAC addresses learned directly by the hardware VTEP are known as local addresses, while the addresses learned from other software or hardware VTEPs are known as remote addresses.

Options Use one or more of the following options to display a more specific list of MAC addresses and information about the MAC addresses. For example, to display a list of local unicast MAC addresses, you can issue the **show ovsdb mac local unicast** command.

none—Display all MAC addresses, which includes all local, remote, unicast, and multicast addresses associated with all logical switches.

address *mac-address*—Display the specified MAC address.

count—(All Juniper Networks devices that support OVSDB except EX9200 switches) (Optional) Display the number of MAC addresses learned by the Juniper Networks device. Using this option alone, the number includes all local, remote, unicast, and multicast MAC addresses associated with all logical switches in the logical switch table of the OVSDB schema for physical devices. You can use this option with one or more of the other options to display a more specific count of MAC addresses. For example, to display the number of local and remote unicast MAC addresses, you can issue the **show ovsdb mac count local remote unicast** command.

local—Display all local MAC addresses.

logical-switch *logical-switch-uuid*—Display all MAC addresses associated with the specified logical switch in the logical switch table of the OVSDB schema for physical devices.

multicast—Display all multicast MAC addresses.

remote—Display all remote MAC addresses.

unicast—Display all unicast MAC addresses.

Required Privilege Level admin

List of Sample Output

Output Fields Table 22 on page 158 lists the output fields for the **show ovsdb mac** command. Output fields are listed in the approximate order in which they appear.

Table 22: show ovsdb mac Output Fields

Field Name	Field Description
Logical Switch Name	Universally unique identifier (UUID) of the logical switch.
MAC Address	MAC addresses of virtual machines (VMs).
IP Address	IP address of VMs. NOTE: If the IP addresses of VMs are not published by the SDN controller, this field displays **0.0.0.0**.
Encapsulation	Encapsulation type.
VTEP Address	IP address of the hardware or software VTEP from which the MAC address was learned. Further, this VTEP can forward VM traffic to the associated host.
MAC Count	NOTE: This field is supported by all Juniper Networks devices that support OVSDB except EX9200 switches. Number of all or specified MAC addresses learned by the Juniper Networks device.

Sample Output

show ovsdb mac

```
user@host> show ovsdb mac
Logical Switch Name: 24a76aff-7e61-4520-a78d-3eca26ad7510
  Mac                 IP                Encapsulation      Vtep
  Address             Address                              Address
  02:00:00:00:03:01   0.0.0.0           Vxlan over Ipv4    10.255.18.22
  02:00:00:00:03:02   0.0.0.0           Vxlan over Ipv4    10.255.18.22
  02:00:00:00:03:03   0.0.0.0           Vxlan over Ipv4    10.255.18.22
  02:00:00:00:03:04   0.0.0.0           Vxlan over Ipv4    10.255.18.22
  02:00:00:00:03:05   0.0.0.0           Vxlan over Ipv4    10.255.18.22
  04:00:00:00:03:05   0.0.0.0           Vxlan over Ipv4    10.255.18.22
  06:00:00:00:03:01   0.0.0.0           Vxlan over Ipv4    10.255.18.22
  06:00:00:00:03:02   0.0.0.0           Vxlan over Ipv4    10.255.18.22
  06:00:00:00:03:03   0.0.0.0           Vxlan over Ipv4    10.255.18.22
  06:00:00:00:03:04   0.0.0.0           Vxlan over Ipv4    10.255.18.22
  06:00:00:00:03:05   0.0.0.0           Vxlan over Ipv4    10.255.18.22
```

```
  40:b4:f0:06:6f:f0      0.0.0.0           Vxlan over Ipv4    10.255.18.22
  ff:ff:ff:ff:ff:ff      0.0.0.0           Vxlan over Ipv4    10.100.100.1

Logical Switch Name: bf6d4fd4-f5f6-430c-8c37-4033ef1c55ab
  Mac                    IP                Encapsulation      Vtep
  Address                Address                              Address
  02:00:00:00:11:01      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  02:00:00:00:11:02      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  02:00:00:00:11:03      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  02:00:00:00:11:04      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  02:00:00:00:11:05      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  04:00:00:00:11:05      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  06:00:00:00:11:01      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  06:00:00:00:11:02      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  06:00:00:00:11:03      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  06:00:00:00:11:04      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  06:00:00:00:11:05      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  40:b4:f0:06:6f:f0      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  00:23:9c:5e:a7:f0      0.0.0.0           Vxlan over Ipv4    10.255.18.22
  08:00:00:00:11:01      0.0.0.0           Vxlan over Ipv4    10.255.18.22
  08:00:00:00:11:02      0.0.0.0           Vxlan over Ipv4    10.255.18.22
  08:00:00:00:11:03      0.0.0.0           Vxlan over Ipv4    10.255.18.22
  08:00:00:00:11:04      0.0.0.0           Vxlan over Ipv4    10.255.18.22
  08:00:00:00:11:05      0.0.0.0           Vxlan over Ipv4    10.255.18.22
  ff:ff:ff:ff:ff:ff      0.0.0.0           Vxlan over Ipv4    10.110.110.1
...
```

show ovsdb mac address

```
user@host> show ovsdb mac address 02:00:00:00:03:01

  Mac                    IP                Encapsulation      Vtep
  Address                Address                              Address
  02:00:00:00:03:01      0.0.0.0           Vxlan over Ipv4    10.255.18.22
```

show ovsdb mac logical-switch

```
user@host> show ovsdb mac logical-switch bf6d4fd4-f5f6-430c-8c37-4033ef1c55ab
Logical Switch Name: bf6d4fd4-f5f6-430c-8c37-4033ef1c55ab
  Mac                    IP                Encapsulation      Vtep
  Address                Address                              Address
  02:00:00:00:11:01      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  02:00:00:00:11:02      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  02:00:00:00:11:03      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  02:00:00:00:11:04      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  02:00:00:00:11:05      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  04:00:00:00:11:05      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  06:00:00:00:11:01      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  06:00:00:00:11:02      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  06:00:00:00:11:03      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  06:00:00:00:11:04      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  06:00:00:00:11:05      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  40:b4:f0:06:6f:f0      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  00:23:9c:5e:a7:f0      0.0.0.0           Vxlan over Ipv4    10.255.18.22
  08:00:00:00:11:01      0.0.0.0           Vxlan over Ipv4    10.255.18.22
  08:00:00:00:11:02      0.0.0.0           Vxlan over Ipv4    10.255.18.22
  08:00:00:00:11:03      0.0.0.0           Vxlan over Ipv4    10.255.18.22
  08:00:00:00:11:04      0.0.0.0           Vxlan over Ipv4    10.255.18.22
  08:00:00:00:11:05      0.0.0.0           Vxlan over Ipv4    10.255.18.22
  ff:ff:ff:ff:ff:ff      0.0.0.0           Vxlan over Ipv4    10.110.110.1
```

show ovsdb mac local unicast

```
user@host> show ovsdb mac local unicast
Logical Switch Name: 24a76aff-7e61-4520-a78d-3eca26ad7510
  Mac                 IP                Encapsulation      Vtep
  Address             Address                              Address
  02:00:00:00:03:01   0.0.0.0           Vxlan over Ipv4    10.255.181.72
  02:00:00:00:03:02   0.0.0.0           Vxlan over Ipv4    10.255.181.72
  02:00:00:00:03:03   0.0.0.0           Vxlan over Ipv4    10.255.181.72
  02:00:00:00:03:04   0.0.0.0           Vxlan over Ipv4    10.255.181.72
  02:00:00:00:03:05   0.0.0.0           Vxlan over Ipv4    10.255.181.72
  04:00:00:00:03:05   0.0.0.0           Vxlan over Ipv4    10.255.181.72
  06:00:00:00:03:01   0.0.0.0           Vxlan over Ipv4    10.255.181.72
  06:00:00:00:03:02   0.0.0.0           Vxlan over Ipv4    10.255.181.72
  06:00:00:00:03:03   0.0.0.0           Vxlan over Ipv4    10.255.181.72
  06:00:00:00:03:04   0.0.0.0           Vxlan over Ipv4    10.255.181.72
  06:00:00:00:03:05   0.0.0.0           Vxlan over Ipv4    10.255.181.72
  40:b4:f0:06:6f:f0   0.0.0.0           Vxlan over Ipv4    10.255.181.72
...
```

show ovsdb mac (Count of All Local, Remote, Unicast, and Multicast MAC Addresses for All Logical Switches)

```
user@host> show ovsdb mac count
MAC count: 6877
```

show ovsdb statistics interface

Supported Platforms MX Series, QFX Series standalone switches

Syntax
show ovsdb statistics interface
<*interface-name*>

Release Information
Command introduced in Junos OS Release 14.1R2.
Command introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.

Description
Display statistics for Open vSwitch Database (OVSDB)-managed interfaces configured by using the **interfaces** ***interface-name*** statement in the [**edit protocols ovsdb**] hierarchy.

When an interface is configured as OVSDB-managed, the collection of statistics for that interface begins, and the statistics displayed at any given time reflects the data collected up to that point.

Options
none—Display statistics for all configured OVSDB-managed interfaces.

interface-name—Display statistics for the specified interface.

Required Privilege Level
admin

Related Documentation
- interfaces on page 123

List of Sample Output
show ovsdb statistics interface on page 161
show ovsdb statistics interface (Specific Interface) on page 162

Output Fields
Table 23 on page 161 lists the output fields for the **show ovsdb statistics interface** command. Output fields are listed in the approximate order in which they appear.

Table 23: show ovsdb statistics interface Output Fields

Field Name	Field Description
Num of rx pkts	Number of packets received by the interface.
Num of tx pkts	Number of packets sent by the interface.
Num of rx bytes	Number of bytes received by the interface.
Num of tx bytes	Number of bytes sent by the interface.

Sample Output

show ovsdb statistics interface

```
user@host> show ovsdb statistics interface
Interface Name: ge-7/0/9.0
Num of rx pkts: 945                    Num of tx pkts: 113280890
```

```
Num of rx bytes: 56700                Num of tx bytes: 57531319540
Interface Name: ge-7/0/10.0
Num of rx pkts: 459                   Num of tx pkts: 473840856
Num of rx bytes: 84747                Num of tx bytes: 45830738532
Interface Name: ge-7/0/11.0
Num of rx pkts: 305                   Num of tx pkts: 367483456
Num of rx bytes: 98974                Num of tx bytes: 33495468092
```

show ovsdb statistics interface (Specific Interface)

```
user@host> show ovsdb statistics interface ge-7/0/9.0
Interface Name: ge-7/0/9.0
Num of rx pkts: 945                   Num of tx pkts: 113280890
Num of rx bytes: 56700                Num of tx bytes: 57531319540
```

show ovsdb tunnels

Supported Platforms QFX Series standalone switches

Syntax
show ovsdb tunnels
<destination-address *ip-address*>
<source-address *ip-address*>

Release Information Command introduced in Junos OS Release 14.1X53-D40 for QFX Series switches.

Description Display the status of tunnels established between Juniper Networks devices that act as hardware virtual tunnel endpoints (VTEPs), and software VTEPs and service nodes that replicate Layer 2 broadcast, unknown unicast, and multicast (BUM) packets within a Virtual Extensible LAN (VXLAN) managed by the Open vSwitch Database (OVSDB) protocol. (In this topic, the software VTEPs and service nodes are known collectively as *replicators*.)

In this topology with VMware NSX controllers deployed, the hardware VTEP uses the Bidirectional Forwarding Detection (BFD) protocol to monitor replicators learned by the NSX controllers. In this situation, BFD is used to decrease the possibility that BUM packets are forwarded to non-functional replicators and eventually dropped.

To monitor the status of the replicators, the hardware VTEP, replicators, and NSX controllers must all be BFD-capable. (Juniper Networks switches that support OVSDB and VXLAN are 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 a functional replicator to handle the packet.

In the output of the **show ovsdb tunnels** command, each tunnel is identified by the IP addresses of the source and destination entities at the ends of the tunnel. Each tunnel that appears is from the perspective of the hardware VTEP. The hardware VTEP provides information about its end of a particular tunnel, while the NSX controller collects the same information from the replicator at the other end of the tunnel. Both hardware VTEP and NSX controller push the information to the tunnel table in the OVSDB schema for physical devices.

Some settings of the **BFD Enabled** and **BFD Status** fields in the **show ovsdb tunnels** output can indicate issues:

- A tunnel with a **BFD Enabled** setting of **Enabled** and a **BFD Status** of **Down** usually indicates that there is an issue with the replicator.
- A tunnel with a **BFD Enabled** setting of **Disabled** and a **BFD Status** of **AdminDown** indicates that either the hardware VTEP, the NSX controller, or the replicator is not BFD-capable or is having an issue with BFD. As a result, a BFD session between the hardware VTEP and the replicator cannot be enabled.

Options Use one or more of the following options to display a more specific list of tunnels.

none—Display all tunnels.

destination-address ***ip-address*** **(Optional)**—Display tunnels that have the specified destination address.

source-address ***ip-address*** **(Optional)**—Display tunnels that have the specified source address.

Required Privilege Level admin

Related Documentation
- Understanding BFD in a VMware NSX Environment with OVSDB and VXLAN on page 17

List of Sample Output
show ovsdb tunnels on page 164
show ovsdb tunnels (Specific Destination) on page 165
show ovsdb tunnels (Specific Source) on page 165

Output Fields Table 24 on page 164 lists the output fields for the **show ovsdb tunnels** command. Output fields are listed in the approximate order in which they appear.

Table 24: show ovsdb tunnels Output Fields

Field Name	Field Descriptions
SRC IP Address	IP address of the OVSDB-managed physical interface from which the BFD control packet is received.
DST IP Address	IP address of the replicator to which the BFD control packet is sent.
BFD Enabled	Status of the BFD protocol on the hardware VTEP and replicator at either ends of the tunnel (**Enabled** or **Disabled**).
BFD Status	Status of a BFD session between a hardware VTEP and a replicator. Possible values include: • **Up**—The BFD session is enabled and up. • **Down**—The BFD session is considered to be down if a replicator does not respond to three BFD control messages. • **AdminDown**—The BFD session is disabled by the NSX controller.

Sample Output

show ovsdb tunnels

```
user@host> show ovsdb tunnels
SRC IP Address      DST IP Address      BFD Enabled     BFD Status
192.168.110.110     192.168.100.1       Enabled         Up
192.168.110.110     192.168.100.2       Disabled        AdminDown
192.168.110.120     192.168.100.20      Enabled         Down
```

show ovsdb tunnels (Specific Destination)

```
user@host> show ovsdb tunnels destination-address 192.168.100.1
SRC IP Address       DST IP Address       BFD Enabled     BFD Status
192.168.110.110      192.168.100.1        Enabled         Up
```

show ovsdb tunnels (Specific Source)

```
user@host> show ovsdb tunnels source-address 192.168.110.110
SRC IP Address       DST IP Address       BFD Enabled     BFD Status
192.168.110.110      192.168.100.1        Enabled         Up
192.168.110.110      192.168.100.2        Disabled        AdminDown
```

show ovsdb virtual-tunnel-end-point

Supported Platforms MX Series, QFX Series standalone switches

Syntax
show ovsdb virtual-tunnel-end-point
address <*ip-address*>
encapsulation <*encapsulation-type*>

Release Information Command introduced in Junos OS Release 14.1R2.
Command introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.

Description Display information about the following entities that the Juniper Networks device has learned:

- Other Juniper Networks devices that function as hardware virtual tunnel endpoints (VTEPs)
- Software VTEPs
- Service nodes
- Top-of-rack service nodes (TSNs)

Options
none—Display information about all VTEPs, service nodes, and TSNs that the Juniper Networks device has learned.

address *ip-address*—Display information about the entity with specified IP address.

encapsulation *encapsulation-type*—Display information about all entities with the specified encapsulation type.

Required Privilege Level admin

List of Sample Output
show ovsdb virtual-tunnel-end-point on page 167
show ovsdb virtual-tunnel-end-point address (Specific Address) on page 167
show ovsdb virtual-tunnel-end-point encapsulation (Specific Encapsulation) on page 167
show ovsdb virtual-tunnel-end-point address (Specific Address) encapsulation (Specific Encapsulation) on page 167

Output Fields Table 25 on page 166 lists the output fields for the **show ovsdb virtual-tunnel-end-point** command. Output fields are listed in the approximate order in which they appear.

Table 25: show ovsdb virtual-tunnel-end-point Output Fields

Field Name	Field Description
Encapsulation	Encapsulation type of entity.
IP Address	IP address of entity.
Num of MACs	Number of MAC addresses learned by the entity.

Sample Output

show ovsdb virtual-tunnel-end-point

```
user@host> show ovsdb virtual-tunnel-end-point
Encapsulation              Ip Address                      Num of MAC's
VXLAN over IPv4            10.255.181.43                   24
VXLAN over IPv4            10.255.181.50                   12
VXLAN over IPv4            10.255.181.72                   24
```

show ovsdb virtual-tunnel-end-point address (Specific Address)

```
user@host> show ovsdb virtual-tunnel-end-point address 10.255.181.43
Encapsulation              Ip Address                      Num of MAC's
VXLAN over IPv4            10.255.181.43                   24
```

show ovsdb virtual-tunnel-end-point encapsulation (Specific Encapsulation)

```
user@host> show ovsdb virtual-tunnel-end-point encapsulation vxlan-over-ipv4
Encapsulation              Ip Address                      Num of MAC's
VXLAN over IPv4            10.255.181.43                   24
VXLAN over IPv4            10.255.181.50                   12
VXLAN over IPv4            10.255.181.72                   24
```

show ovsdb virtual-tunnel-end-point address (Specific Address) encapsulation (Specific Encapsulation)

```
user@host> show ovsdb virtual-tunnel-end-point address 10.255.181.43 encapsulation
vxlan-over-ipv4
Encapsulation              Ip Address                      Num of MAC's
VXLAN over IPv4            10.255.181.43                   24
```

CHAPTER 6

VXLAN Monitoring Commands

Monitor a Remote VTEP Interface

Supported Platforms QFX Series standalone switches

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 This 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 on page 3
- Configuring VXLANs on a QFX5100 Switch on page 45
- Examples: Manually Configuring VXLANs on QFX Series Switches on page 81

ping overlay

Supported Platforms QFX Series standalone switches

Syntax ping overlay [count *value*] [hash-parameters source-mac *src-mac*] [source-host *ip-address*] [ttl *value*] tunnel-dst *ip-address* tunnel-src *ip-address* tunnel-type vxlan vni *id*

Release Information Command introduced in Junos OS Release 14.1X53-D30 for QFX Series switches.

Description On a QFX5100 switch, force the **ping** packets to follow the same path as data packets through a VXLAN tunnel. In other words, make the underlay packets (**ping** packets) take the same route as the overlay packets (data traffic).

Options

count *value*—Number of pings to send (1-65535).

hash-parameters source-mac *src-mac*—Not supported.

source-host *ip-address*—IP address of the host (virtual machine or bare metal server) that is the source of the tunnel. The default address is 127.0.0.1.

ttl *value*—TTL value to use in the ping packets (1-255).

tunnel-dst *ip-address*—IP address of the remote virtual tunnel endpoint (VTEP).

tunnel-src *ip-address*—IP address of the source VTEP.

tunnel-type vxlan—Value must be **vxlan**.

vni *id* —Value of the VXLAN network identifier that identifies the overlay segment (0-16777215).

Additional Information

Required Privilege Level

Related Documentation

- Understanding VXLANs on page 3
- Examples: Manually Configuring VXLANs on QFX Series Switches on page 81
- VXLAN Constraints on QFX5100 Switches on page 9
- **traceroute overlay on page 181**

show bridge mac-table

Supported Platforms MX Series

Syntax
show bridge mac-table
<brief | count | detail | extensive>
<bridge-domain (all | *bridge-domain-name*)>
<global-count>
<interface *interface-name*>
<*mac-address*>
<vlan-id (all-vlan | *vlan-id*)>

Release Information Command introduced in Junos OS Release 8.4.

Description (MX Series routers only) Display Layer 2 MAC address information.

Options
none—Display all learned Layer 2 MAC address information.

brief | count | detail | extensive—(Optional) Display the specified level of output.

bridge-domain (all | *bridge-domain-name*)—(Optional) Display learned Layer 2 MAC addresses for all bridging domains or for the specified bridging domain.

global-count—(Optional) Display the total number of learned Layer 2 MAC addresses on the system.

instance *instance-name*—(Optional) Display learned Layer 2 MAC addresses for the specified routing instance.

interface *interface-name*—(Optional) Display learned Layer 2 MAC addresses for the specified interface.

mac-address—(Optional) Display the specified learned Layer 2 MAC address information.

vlan-id (all-vlan | *vlan-id*)—(Optional) Display learned Layer 2 MAC addresses for all VLANs or for the specified VLAN.

Additional Information When Layer 2 protocol tunneling is enabled, the tunneling MAC address 01:00:0c:cd:cd:d0 is installed in the MAC table. When the Cisco Discovery Protocol (CDP), Spanning Tree Protocol (STP), or VLAN Trunk Protocol (VTP) is configured for Layer 2 protocol tunneling on an interface, the corresponding protocol MAC address is installed in the MAC table.

Required Privilege Level view

List of Sample Output
show bridge mac-table on page 174
show bridge mac-table (with PBB-EVPN enabled) on page 174
show bridge mac-table (with VXLAN enabled) on page 174
show bridge mac-table count on page 175
show bridge mac-table detail on page 175

Output Fields Table 26 on page 173 describes the output fields for the **show bridge mac-table** command. Output fields are listed in the approximate order in which they appear.

Table 26: show bridge mac-table Output fields

Field Name	Field Description
Routing instance	Name of the routing instance.
Bridging domain	Name of the bridging domain.
MAC address	MAC address or addresses learned on a logical interface.
MAC flags	Status of MAC address learning properties for each interface: • **S**—Static MAC address is configured. • **D**—Dynamic MAC address is configured. • **L**—Locally learned MAC address is configured. • **C**—Control MAC address is configured. • **SE**—MAC accounting is enabled. • **NM**—Non-configured MAC. • **R**—Remote PE MAC address is configured.
Logical interface	Name of the logical interface.
MAC count	Number of MAC addresses learned on the specific routing instance or interface.
Learning interface	Name of the logical interface on which the MAC address was learned.
Learning VLAN	VLAN ID of the routing instance or bridge domain in which the MAC address was learned.
VXLAN ID/VXLAN	VXLAN Network Identifier (VNI)
Layer 2 flags	Debugging flags signifying that the MAC address is present in various lists.
Epoch	Spanning Tree Protocol epoch number identifying when the MAC address was learned. Used for debugging.
Sequence number	Sequence number assigned to this MAC address. Used for debugging.
Learning mask	Mask of the Packet Forwarding Engines where this MAC address was learned. Used for debugging.
IPC generation	Creation time of the logical interface when this MAC address was learned. Used for debugging.

Sample Output

show bridge mac-table

```
user@host> show bridge mac-table
MAC flags (S -static MAC, D -dynamic MAC, L -locally learned, C -Control MAC
           SE -Statistics enabled, NM -Non configured MAC, R -Remote PE MAC)

Routing instance : default-switch
 Bridging domain : test1, VLAN : 1
   MAC                 MAC      Logical          NH     RTR
   address             flags    interface        Index  ID
   01:00:0c:cc:cc:cc   S,NM     NULL
   01:00:0c:cc:cc:cd   S,NM     NULL
   01:00:0c:cd:cd:d0   S,NM     NULL
   64:87:88:6a:17:d0   D        ae0.1
   64:87:88:6a:17:f0   D        ae0.1
```

show bridge mac-table (with PBB-EVPN enabled)

```
user@host> show bridge mac-table
MAC flags       (S -static MAC, D -dynamic MAC, L -locally learned, C -Control
MAC
    O -OVSDB MAC, SE -Statistics enabled, NM -Non configured MAC, R -Remote PE
MAC)

Routing instance : pbbn10
 Bridging domain : bda, VLAN : 100
   MAC                 MAC      Logical          NH     RTR
   address             flags    interface        Index  ID
   00:26:88:5f:67:b0   DC                        1048581 1048581
   00:51:51:51:51:51   DC                        1048581 1048581
   00:52:52:52:52:52   DC                        1048576 1048576
   01:1e:83:00:03:e8   DC                        1048580 0
   01:1e:83:00:07:d0   DC                        1048579 0
   a8:d0:e5:5b:01:c8   DC                        1048576 1048576
```

show bridge mac-table (with VXLAN enabled)

```
user@host> show bridge mac-table
MAC flags (S -static MAC, D -dynamic MAC, L -locally learned
           SE -Statistics enabled, NM -Non configured MAC, R -Remote PE MAC)

Routing instance : default-switch
 Bridging domain : vlan-1, VLAN : 1
  VXLAN: Id : 100, Multicast group: 226.1.1.1
   MAC                 MAC      Logical
   address             flags    interface
   00:01:01:00:01:f7   D,SE     vtep.1052010
   00:03:00:32:01:f7   D,SE     vtep.1052011
   00:00:21:11:11:10   DL       ge-1/0/0.0
   00:00:21:11:11:11   DL       ge-1/1/0.0


Routing instance : default-switch
 Bridging domain : vlan-2, VLAN : 2, VXLAN : 200
  VXLAN: Id : 200, Multicast group: 226.1.1.2
   MAC                 MAC      Logical
   address             flags    interface
   00:02:01:33:01:f7   D,SE     vtep.1052010
```

```
  00:04:00:14:01:f7   D,SE     vtep.1052011
  00:00:21:11:21:10   DL       ge-1/0/0.1
  00:00:21:11:21:11   DL       ge-1/1/0.1
```

show bridge mac-table count

```
user@host> show bridge mac-table count
2 MAC address learned in routing instance vs1 bridge domain vlan100

  MAC address count per interface within routing instance:
    Logical interface        MAC count
    ge-11/0/3.0                      1
    ge-11/1/4.100                    0
    ge-11/1/1.100                    0
    ge-11/1/0.100                    0
    xe-10/2/0.100                    1
    xe-10/0/0.100                    0

  MAC address count per learn VLAN within routing instance:
    Learn VLAN ID            MAC count
                0                    2

0 MAC address learned in routing instance vs1 bridge domain vlan200

  MAC address count per interface within routing instance:
    Logical interface        MAC count
    ge-11/1/0.200                    0
    ge-11/1/1.200                    0
    ge-11/1/4.200                    0
    xe-10/0/0.200                    0
    xe-10/2/0.200                    0

  MAC address count per learn VLAN within routing instance:
    Learn VLAN ID            MAC count
                0                    0
```

show bridge mac-table detail

```
user@host> show bridge mac-table detail
MAC address: 00:00:00:19:1c:db
  Routing instance: vs1
   Bridging domain: vlan100
   Learning interface: ge-11/0/3.0     Learning VLAN: 0
   Layer 2 flags: in_ifd, in_ifl, in_vlan, kernel
   Epoch: 4                            Sequence number: 0
   Learning mask: 0x800                IPC generation: 0

MAC address: 00:00:00:59:3a:2f
  Routing instance: vs1
   Bridging domain: vlan100
   Learning interface: xe-10/2/0.100   Learning VLAN: 0
   Layer 2 flags: in_ifd, in_ifl, in_vlan, kernel
   Epoch: 7                            Sequence number: 0
   Learning mask: 0x400                IPC generation: 0
```

show vpls mac-table

Supported Platforms MX960

Syntax
show vpls mac-table
<**brief | detail | extensive | summary**>
<**bridge-domain** *bridge-domain-name*>
<**instance** *instance-name*>
<**interface** *interface-name*>
<**logical-system (all |** *logical-system-name***)**>
<***mac-address***>
<**vlan-id** *vlan-id-number*>

Release Information Command introduced in Junos OS Release 8.5.

Description (MX960 routers only) Display learned VPLS MAC address information.

Options
none—Display all learned VPLS MAC address information.

brief | detail | extensive | summary—(Optional) Display the specified level of output.

bridge-domain ***bridge-domain-name***—(Optional) Display learned VPLS MAC addresses for the specified bridge domain.

instance ***instance-name***—(Optional) Display learned VPLS MAC addresses for the specified instance.

interface ***interface-name***—(Optional) Display learned VPLS MAC addresses for the specified instance.

logical-system (all | ***logical-system-name*****)**—(Optional) Display learned VPLS MAC addresses for all logical systems or for the specified logical system.

mac-address—(Optional) Display the specified learned VPLS MAC address information..

vlan-id ***vlan-id-number***—(Optional) Display learned VPLS MAC addresses for the specified VLAN.

Required Privilege Level view

List of Sample Output
show vpls mac-table on page 177
show vpls mac-table (with VXLAN enabled) on page 178
show vpls mac-table count on page 178
show vpls mac-table detail on page 179
show vpls mac-table extensive on page 179

Output Fields Table 27 on page 177 describes the output fields for the **show bridge mac-table** command. Output fields are listed in the approximate order in which they appear.

Table 27: show vpls mac-table Output fields

Field Name	Field Description
Routing instance	Name of the routing instance.
Bridging domain	Name of the bridging domain.
MAC address	MAC address or addresses learned on a logical interface.
MAC flags	Status of MAC address learning properties for each interface: • **S**—Static MAC address configured. • **D**—Dynamic MAC address learned. • **SE**—MAC accounting is enabled. • **NM**—Nonconfigured MAC.
Logical interface	Name of the logical interface.
MAC count	Number of MAC addresses learned on a specific routing instance or interface.
Learning interface	Logical interface or logical Label Switched Interface (LSI) the address is learned on.
Learn VLAN ID/VLAN	VLAN ID of the routing instance or bridge domain in which the MAC address was learned.
VXLAN ID/VXLAN	VXLAN Network Identifier (VNI)
Layer 2 flags	Debugging flags signifying that the MAC address is present in various lists.
Epoch	Spanning Tree Protocol epoch number identifying when the MAC address was learned. Used for debugging.
Sequence number	Sequence number assigned to this MAC address. Used for debugging.
Learning mask	Mask of Packet Forwarding Engines where this MAC address was learned. Used for debugging.
IPC generation	Creation time of the logical interface when this MAC address was learned. Used for debugging.

Sample Output

show vpls mac-table

```
user@host> show vpls mac-table
MAC flags (S -static MAC, D -dynamic MAC,
           SE -Statistics enabled, NM -Non configured MAC)

Routing instance : vpls_ldp1
VLAN : 223
   MAC                MAC      Logical
   address            flags    interface
   00:90:69:9c:1c:5d  D        ge-0/2/5.400

MAC flags (S -static MAC, D -dynamic MAC,
```

```
           SE -Statistics enabled, NM -Non configured MAC)

Routing instance : vpls_red
VLAN : 401
   MAC                MAC      Logical
   address            flags    interface
   00:00:aa:12:12:12  D        lsi.1051138
   00:05:85:74:9f:f0  D        lsi.1051138
```

show vpls mac-table (with VXLAN enabled)

```
user@host> show vpls mac-table
MAC flags (S -static MAC, D -dynamic MAC, L -locally learned
           SE -Statistics enabled, NM -Non configured MAC, R -Remote PE MAC)

Routing instance : vpls_4site:1000
 Bridging domain : __vpls_4site:1000__, VLAN : 4094,4093
  VXLAN: Id : 300, Multicast group: 226.1.1.3
   MAC                MAC      Logical
   address            flags    interface
   00:01:01:00:01:f4  D,SE     ge-4/2/0.1000
   00:02:01:33:01:f4  D,SE     lsi.1052004
   00:03:00:32:01:f4  D,SE     lsi.1048840
   00:04:00:14:01:f4  D,SE     lsi.1052005
   00:02:01:33:02:f7  D,SE     vtep.1052010
   00:04:00:14:02:f7  D,SE     vtep.1052011
```

show vpls mac-table count

```
user@host> show vpls mac-table count
0 MAC address learned in routing instance __juniper_private1__

  MAC address count per interface within routing instance:
    Logical interface        MAC count
    lc-0/0/0.32769                   0
    lc-0/1/0.32769                   0
    lc-0/2/0.32769                   0
    lc-2/0/0.32769                   0
    lc-0/3/0.32769                   0
    lc-2/1/0.32769                   0
    lc-9/0/0.32769                   0
    lc-11/0/0.32769                  0
    lc-2/2/0.32769                   0
    lc-9/1/0.32769                   0
    lc-11/1/0.32769                  0
    lc-2/3/0.32769                   0
    lc-9/2/0.32769                   0
    lc-11/2/0.32769                  0
    lc-11/3/0.32769                  0
    lc-9/3/0.32769                   0

  MAC address count per learn VLAN within routing instance:
    Learn VLAN ID            MAC count
                0                    0

1 MAC address learned in routing instance vpls_ldp1

  MAC address count per interface within routing instance:
    Logical interface        MAC count
    lsi.1051137                      0
    ge-0/2/5.400                     1
```

```
  MAC address count per learn VLAN within routing instance:
    Learn VLAN ID            MAC count
                 0                    1

1 MAC address learned in routing instance vpls_red

  MAC address count per interface within routing instance:
    Logical interface        MAC count
    ge-0/2/5.300                      1

  MAC address count per learn VLAN within routing instance:
    Learn VLAN ID            MAC count
                 0                    1
```

show vpls mac-table detail

```
user@host> show vpls mac-table detail
MAC address: 00:90:69:9c:1c:5d
  Routing instance: vpls_ldp1
   Learning interface: ge-0/2/5.400
   Layer 2 flags: in_ifd, in_ifl, in_vlan, kernel
   Epoch: 0                          Sequence number: 1
   Learning mask: 0x1                IPC generation: 0

MAC address: 00:90:69:9c:1c:5d
  Routing instance: vpls_red
   Learning interface: ge-0/2/5.300
   Layer 2 flags: in_ifd, in_ifl, in_vlan, kernel
   Epoch: 0                          Sequence number: 1
   Learning mask: 0x1                IPC generation: 0
```

show vpls mac-table extensive

```
user@host> show vpls mac-table extensive
MAC address: 00:00:aa:12:12:12
  Routing instance: vpls_ldp1
   Learning interface: lsi.1051137
   Layer 2 flags: in_ifd, in_ifl, in_vlan, kernel
   Epoch: 0                          Sequence number: 1
   Learning mask: 0x1                IPC generation: 0

MAC address: 00:05:85:74:9f:f0
  Routing instance: vpls_ldp1
   Learning interface: lsi.1051137
   Layer 2 flags: in_ifd, in_ifl, in_vlan, kernel
   Epoch: 0                          Sequence number: 1
   Learning mask: 0x1                IPC generation: 0

MAC address: 00:90:69:9c:1c:5d
  Routing instance: vpls_ldp1
   Learning interface: ge-0/2/5.400
   Layer 2 flags: in_ifd, in_ifl, in_vlan, kernel
   Epoch: 0                          Sequence number: 1
   Learning mask: 0x1                IPC generation: 0

MAC address: 00:00:aa:12:12:12
  Routing instance: vpls_red
   Learning interface: lsi.1051138
   Layer 2 flags: in_ifd, in_ifl, in_vlan, kernel
   Epoch: 0                          Sequence number: 0
```

```
    Learning mask: 0x1                  IPC generation: 0

MAC address: 00:05:85:74:9f:f0
  Routing instance: vpls_red
    Learning interface: lsi.1051138
    Layer 2 flags: in_ifd, in_ifl, in_vlan, kernel
    Epoch: 0                            Sequence number: 0
    Learning mask: 0x1                  IPC generation: 0
```

traceroute overlay

Supported Platforms QFX Series standalone switches

Syntax traceroute overlay [count *value*] [hash-parameters source-mac *src-mac*] [source-host *ip-address*] [ttl *value*] tunnel-dst *ip-address* tunnel-src *ip-address* tunnel-type vxlan vni *id*

Release Information Command introduced in Junos OS Release 14.1X53-D30 for QFX Series switches.

Description On a QFX5100 switch, force the **traceroute** packets to follow the same path as data packets through a VXLAN tunnel. In other words, make the underlay packets (**traceroute** packets) take the same route as the overlay packets (data traffic).

Options

count *value*—Number of packets to send (1-65535).

hash-parameters source-mac *src-mac*—Not supported.

source-host *ip-address*—IP address of the host (virtual machine or bare metal server) that is the source of the tunnel. The default address is 127.0.0.1.

ttl *value*—TTL value to use in the traceroute packets (1-255).

tunnel-dst *ip-address*—IP address of the remote virtual tunnel endpoint (VTEP).

tunnel-src *ip-address*—IP address of the source VTEP.

tunnel-type vxlan—Value must be **vxlan**.

vni *id* —Value of the VXLAN network identifier that identifies the overlay segment (0-16777215).

Additional Information

Required Privilege Level

Related Documentation

- Understanding VXLANs on page 3
- Examples: Manually Configuring VXLANs on QFX Series Switches on page 81
- VXLAN Constraints on QFX5100 Switches on page 9
- ping overlay on page 171

Verifying VXLAN Reachability

Supported Platforms QFX Series standalone switches

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              232.1.1.1
```

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 on page 3
- Configuring VXLANs on a QFX5100 Switch on page 45
- Examples: Manually Configuring VXLANs on QFX Series Switches on page 81

Verifying That a Local VXLAN VTEP is Configured Correctly

Supported Platforms QFX Series standalone switches

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      232.1.1.1
```

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

Related Documentation

- Understanding VXLANs on page 3
- Configuring VXLANs on a QFX5100 Switch on page 45
- Examples: Manually Configuring VXLANs on QFX Series Switches on page 81

Verifying MAC Learning from a Remote VTEP

Supported Platforms QFX Series standalone switches

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 This 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 on page 3
- Configuring VXLANs on a QFX5100 Switch on page 45
- Examples: Manually Configuring VXLANs on QFX Series Switches on page 81