JUNIPER NETWORKS | Engineering Simplicity

Ethernet Switching User Guide

Published
2025-12-11

JUNOS

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

Ethernet Switching User Guide

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

YEAR 2000 NOTICE

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

END USER LICENSE AGREEMENT

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

Table of Contents

16 Stacking and Rewriting Gigabit Ethernet VLAN Tags

About This Guide

Use this guide to configure and monitor Layer 2 features.

1

CHAPTER

Understanding Layer 2 Networking

IN THIS CHAPTER

Layer 2 Networking

IN THIS SECTION

Overview of Layer 2 Networking

Layer 2, also known as the Data Link Layer, is the second level in the seven-layer OSI reference model for network protocol design. Layer 2 is equivalent to the link layer (the lowest layer) in the TCP/IP network model. Layer2 is the network layer used to transfer data between adjacent network nodes in a wide area network or between nodes on the same local area network.

A *frame* is a protocol data unit, the smallest unit of bits on a Layer 2 network. Frames are transmitted to and received from devices on the same local area network (LAN). Unlike bits, frames have a defined structure and can be used for error detection, control plane activities and so forth. Not all frames carry user data. The network uses some frames to control the data link itself.

At Layer 2, *unicast* refers to sending frames from one node to a single other node, whereas *multicast* denotes sending traffic from one node to multiple nodes, and *broadcasting* refers to the transmission of frames to all nodes in a network. A *broadcast domain* is a logical division of a network in which all nodes of that network can be reached at Layer 2 by a broadcast.

Segments of a LAN can be linked at the frame level using *bridges*. Bridging creates separate broadcast domains on the LAN, creating VLANs, which are independent logical networks that group together related devices into separate network segments. The grouping of devices on a VLAN is independent of where the devices are physically located in the LAN. Without bridging and VLANs, all devices on the Ethernet LAN are in a single broadcast domain, and all the devices detect all the packets on the LAN.

Forwarding is the relaying of packets from one network segment to another by nodes in the network. On a VLAN, a frame whose origin and destination are in the same VLAN are forwarded only within the local VLAN. A network segment is a portion of a computer network wherein every device communicates using the same physical layer.

Layer 2 contains two sublayers:

- Logical link control (LLC) sublayer, which is responsible for managing communications links and handling frame traffic.
- Media access control (MAC) sublayer, which governs protocol access to the physical network medium. By using the MAC addresses that are assigned to all ports on a switch, multiple devices on the same physical link can uniquely identify one another.

  The ports, or interfaces, on a switch operate in either access mode, tagged-access, or trunk mode:

  - *Access mode* ports connect to a network device such as a desktop computer, an IP telephone, a printer, a file server, or a security camera. The port itself belongs to a single VLAN. The frames transmitted over an access interface are normal Ethernet frames. By default, all ports on a switch are in access mode.
  - *Tagged-Access mode* ports connect to a network device such as a desktop computer, an IP telephone, a printer, a file server, or a security camera. The port itself belongs to a single VLAN. The frames transmitted over an access interface are normal Ethernet frames. By default, all ports on a switch are in access mode. Tagged-access mode accommodates cloud computing, specifically scenarios including virtual machines or virtual computers. Because several virtual computers can be included on one physical server, the packets generated by one server can contain an aggregation of VLAN packets from different virtual machines on that server. To accommodate this situation, tagged-access mode reflects packets back to the physical server on the same downstream port when the destination address of the packet was learned on that downstream port. Packets are also reflected back to the physical server on the downstream port when the destination has not yet been learned. Therefore, the third interface mode, tagged access, has some characteristics of access mode and some characteristics of trunk mode:
  - *Trunk mode* ports handle traffic for multiple VLANs, multiplexing the traffic for all those VLANs over the same physical connection. Trunk interfaces are generally used to interconnect switches to other devices or switches.

    With native VLAN configured, frames that do not carry VLAN tags are sent over the trunk interface. If you have a situation where packets pass from a device to a switch in access mode,

and you want to then send those packets from the switch over a trunk port, use native VLAN mode. Configure the single VLAN on the switch's port (which is in access mode) as a native VLAN. The switch's trunk port will then treat those frames differently than the other tagged packets. For example, if a trunk port has three VLANs, 10, 20, and 30, assigned to it with VLAN 10 being the native VLAN, frames on VLAN 10 that leave the trunk port on the other end have no 802.1Q header (tag). There is another native VLAN option. You can have the switch add and remove tags for untagged packets. To do this, you first configure the single VLAN as a native VLAN on a port attached to a device on the edge. Then, assign a VLAN ID tag to the single native VLAN on the port connected to a device. Last, add the VLAN ID to the trunk port. Now, when the switch receives the untagged packet, it adds the ID you specified and sends and receives the tagged packets on the trunk port configured to accept that VLAN.

Including the sublayers, Layer 2 on the QFX Series supports the following functionality:

- Unicast, multicast, and broadcast traffic.
- Bridging.
- VLAN 802.1Q—Also known as *VLAN tagging*, this protocol allows multiple bridged networks to transparently share the same physical network link by adding VLAN tags to an Ethernet frame.
- Extension of Layer 2 VLANs across multiple switches using Spanning Tree Protocol (STP) prevents looping across the network.
- *MAC learning*, including per-VLAN MAC learning and Layer 2 learning suppression–This process obtains the MAC addresses of all the nodes on a network
- Link aggregation—This process groups of Ethernet interfaces at the physical layer to form a single link layer interface, also known as a *link aggregation group (LAG)* or LAG bundle

  NOTE: Link aggregation is not supported on NFX150 devices.

- Storm control on the physical port for unicast, multicast, and broadcast

  NOTE: Storm control is not supported on NFX150 devices.

- STP support, including 802.1d, RSTP, MSTP, and Root Guard

SEE ALSO

Understanding Bridging and VLANs on Switches | 140

Understanding VLANs

A VLAN (virtual LAN) is a collection of network nodes grouped together to form separate broadcast domains. On an Ethernet network that is a single LAN, all traffic is forwarded to all nodes on the LAN. On VLANs, frames whose origin and destination are in the same VLAN are forwarded only within the local VLAN. Frames that are not destined for the local VLAN are the only ones forwarded to other broadcast domains. VLANs thus limit the amount of traffic flowing across the entire LAN, reducing the possible number of collisions and packet retransmissions within a VLAN and on the whole LAN.

On an Ethernet LAN, all network nodes must be physically connected to the same network. On VLANs, the physical location of the nodes is not important; therefore, you can group network devices in any way that makes sense for your organization, such as by department or business function, by types of network nodes, or by physical location. Each VLAN is identified by a single IP subnetwork and by standardized IEEE 802.1Q encapsulation.

To identify which VLAN the traffic belongs to, all frames on an Ethernet VLAN are identified by a tag, as defined in the IEEE 802.1Q standard. These frames are tagged and are encapsulated with 802.1Q tags.

For a simple network that has only a single VLAN, all traffic has the same 802.1Q tag. When an Ethernet LAN is divided into VLANs, each VLAN is identified by a unique 802.1Q tag. The tag is applied to all frames so that the network nodes receiving the frames know to which VLAN a frame belongs. Trunk ports, which multiplex traffic among a number of VLANs, use the tag to determine the origin of frames and where to forward them.

SEE ALSO

Understanding Bridging and VLANs on Switches | **140**

Ethernet Switching and Layer 2 Transparent Mode Overview

IN THIS SECTION

Layer 2 transparent mode provides the ability to deploy the firewall without making changes to the existing routing infrastructure. The firewall is deployed as a Layer 2 switch with multiple VLAN segments and provides security services within VLAN segments. Secure wire is a special version of Layer 2 transparent mode that allows bump-in-wire deployment.

A device operates in transparent mode when there are interfaces defined as Layer 2 interfaces. The device operates in route mode (the default mode) if there are no physical interfaces configured as Layer 2 interfaces.

For SRX Series Firewalls, transparent mode provides full security services for Layer 2 switching capabilities. On these SRX Series Firewalls, you can configure one or more VLANs to perform Layer 2 switching. A VLAN is a set of logical interfaces that share the same flooding or broadcast characteristics. Like a virtual LAN (VLAN), a VLAN spans one or more ports of multiple devices. Thus, the SRX Series Firewall can function as a Layer 2 switch with multiple VLANs that participate in the same Layer 2 network.

In transparent mode, the SRX Series Firewall filters packets that traverse the device without modifying any of the source or destination information in the IP packet headers. Transparent mode is useful for protecting servers that mainly receive traffic from untrusted sources because there is no need to reconfigure the IP settings of routers or protected servers.

In transparent mode, all physical ports on the device are assigned to Layer 2 interfaces. Do not route Layer 3 traffic through the device. Layer 2 zones can be configured to host Layer 2 interfaces, and security policies can be defined between Layer 2 zones. When packets travel between Layer 2 zones, security policies can be enforced on these packets.

Table 1 on page 7 lists the security features that are supported and are not supported in transparent mode for Layer 2 switching.

Table 1: Security Features Supported in Transparent Mode

Mode Type	Supported	Not Supported
Transparent mode	• Application Layer Gateways (ALGs) • Firewall User Authentication (FWAUTH) • Intrusion Detection and Prevention (IDP) • Screen • AppSecure • Content Security	• Network Address Translation (NAT) • VPN

NOTE: On SRX300, SRX320, SRX340, SRX345, and SRX550M devices, the DHCP server propagation is not supported in Layer 2 transparent mode.

In addition, the SRX Series Firewalls do not support the following Layer 2 features in Layer 2 transparent mode:

- Spanning Tree Protocol (STP), RSTP, or MSTP—It is the user's responsibility to ensure that no flooding loops exist in the network topology.
- Internet Group Management Protocol (IGMP) snooping—Host-to-router signaling protocol for IPv4 used to report their multicast group memberships to neighboring routers and determine whether group members are present during IP multicasting.
- Double-tagged VLANs or IEEE 802.1Q VLAN identifiers encapsulated within 802.1Q packets (also called "Q in Q" VLAN tagging)—Only untagged or single-tagged VLAN identifiers are supported on SRX Series Firewalls.
- Nonqualified VLAN learning, where only the MAC address is used for learning within the VLAN—VLAN learning on SRX Series Firewalls is qualified; that is, both the VLAN identifier and MAC address are used.

Also, on SRX100, SRX110, SRX210, SRX220, SRX240, SRX300, SRX320, SRX340, SRX345, SRX550, or SRX650 devices, some features are not supported. (Platform support depends on the Junos OS release in your installation.) The following features are not supported for Layer 2 transparent mode on the mentioned devices:

- G-ARP on the Layer 2 interface
- IP address monitoring on any interface
- Transit traffic through IRB
- IRB interface in a routing instance
- IRB interface handling of Layer 3 traffic

> **NOTE**: The IRB interface is a pseudointerface and does not belong to the reth interface and redundancy group.

Layer 2 Transparent Mode on the SRX5000 Line Module Port Concentrator

The SRX5000 line Module Port Concentrator (SRX5K-MPC) supports Layer 2 transparent mode and processes the traffic when the SRX Series Firewall is configured in Layer 2 transparent mode.

When the SRX5K-MPC is operating in Layer 2 mode, you can configure all interfaces on the SRX5K-MPC as Layer 2 switching ports to support Layer 2 traffic.

The security processing unit (SPU) supports all security services for Layer 2 switching functions, and the MPC delivers the ingress packets to the SPU and forwards the egress packets that are encapsulated by the SPU to the outgoing interfaces.

When the SRX Series Firewall is configured in Layer 2 transparent mode, you can enable the interfaces on the MPC to work in Layer 2 mode by defining one or more logical units on a physical interface with the family address type as `Ethernet switching`. Later you can proceed with configuring Layer 2 security zones and configuring security policies in transparent mode. Once this is done, next-hop topologies are set up to process ingress and egress packets.

Understanding IPv6 Flows in Transparent Mode on Security Devices

In transparent mode, the SRX Series Firewall filters packets that traverse the device without modifying any of the source or destination information in the packet MAC headers. Transparent mode is useful for protecting servers that mainly receive traffic from untrusted sources because there is no need to reconfigure the IP settings of routers or protected servers.

A device operates in transparent mode when all physical interfaces on the device are configured as Layer 2 interfaces. A physical interface is a Layer 2 interface if its *logical interface* is configured with the `ethernet-switching` option at the [`edit interfaces` *`interface-name`* `unit` *`unit-number`* `family`] hierarchy level. There is no command to define or enable transparent mode on the device. The device operates in transparent mode when there are interfaces defined as Layer 2 interfaces. The device operates in route mode (the default mode) if all physical interfaces are configured as Layer 3 interfaces.

By default, IPv6 flows are dropped on security devices. To enable processing by security features such as zones, screens, and firewall policies, you must enable flow-based forwarding for IPv6 traffic with the `mode flow-based` configuration option at the [`edit security forwarding-options family inet6`] hierarchy level. You must reboot the device when you change the mode.

In transparent mode, you can configure Layer 2 zones to host Layer 2 interfaces, and you can define security policies between Layer 2 zones. When packets travel between Layer 2 zones, security policies can be enforced on these packets. The following security features are supported for IPv6 traffic in transparent mode:

- Layer 2 security zones and security policies. See Understanding Layer 2 Security Zones and Understanding Security Policies in Transparent Mode .
- Firewall user authentication. See Understanding Firewall User Authentication in Transparent Mode .
- Layer 2 transparent mode chassis clusters.
- *Class of service* functions. See Class of Service Functions in Transparent Mode Overview.

The following security features are *not* supported for IPv6 flows in transparent mode:

- Logical systems
- IPv6 GTPv2
- J-Web interface
- NAT
- IPsec VPN
- With the exception of DNS, FTP, and TFTP ALGs, all other ALGs are not supported.

Configuring VLANs and Layer 2 logical interfaces for IPv6 flows is the same as configuring VLANs and Layer 2 logical interfaces for IPv4 flows. You can optionally configure an integrated routing and bridging (IRB) interface for management traffic in a VLAN. The IRB interface is the only Layer 3 interface allowed in transparent mode. The IRB interface on the SRX Series Firewall does not support traffic forwarding or routing. The IRB interface can be configured with both IPv4 and IPv6 addresses. You can assign an IPv6 address for the IRB interface with the `address` *configuration statement* at the [`edit interfaces irb unit` *`number`* `family inet6`] hierarchy level. You can assign an IPv4 address for the IRB interface with the `address` configuration statement at the [`edit interfaces irb unit` *`number`* `family inet`] hierarchy level.

The Ethernet Switching functions on SRX Series Firewalls are similar to the switching features on Juniper Networks MX Series routers. However, not all Layer 2 networking features supported on MX Series routers are supported on SRX Series Firewalls. See Ethernet Switching and Layer 2 Transparent Mode Overview.

The SRX Series Firewall maintains forwarding tables that contain MAC addresses and associated interfaces for each Layer 2 VLAN. The IPv6 flow processing is similar to IPv4 flows. See "Layer 2 Learning and Forwarding for VLANs Overview" on page 102.

Understanding Layer 2 Transparent Mode Chassis Clusters on Security Devices

A pair of SRX Series Firewalls in Layer 2 transparent mode can be connected in a *chassis cluster* to provide network node redundancy. When configured in a chassis cluster, one node acts as the primary device and the other as the secondary device, ensuring stateful failover of processes and services in the event of system or hardware failure. If the primary device fails, the secondary device takes over processing of traffic.

NOTE: If the primary device fails in a Layer 2 transparent mode chassis cluster, the physical ports in the failed device become inactive (go down) for a few seconds before they become active (come up) again.

To form a chassis cluster, a pair of the same kind of supported SRX Series Firewalls combines to act as a single system that enforces the same overall security.

Devices in Layer 2 transparent mode can be deployed in active/backup and active/active chassis cluster configurations.

The following chassis cluster features are not supported for devices in Layer 2 transparent mode:

- Gratuitous ARP—The newly elected primary in a redundancy group cannot send gratuitous ARP requests to notify network devices of a change in primary role on the redundant Ethernet interface links.
- IP address monitoring—Failure of an upstream device cannot be detected.

A redundancy group is a construct that includes a collection of objects on both nodes. A redundancy group is primary on one node and backup on the other. When a redundancy group is primary on a node, its objects on that node are active. When a redundancy group fails over, all its objects fail over together.

You can create one or more redundancy groups numbered 1 through 128 for an active/active chassis cluster configuration. Each redundancy group contains one or more redundant Ethernet interfaces. A redundant Ethernet interface is a pseudointerface that contains physical interfaces from each node of the cluster. The physical interfaces in a redundant Ethernet interface must be the same kind—either Fast Ethernet or Gigabit Ethernet. If a redundancy group is active on node 0, then the child links of all associated redundant Ethernet interfaces on node 0 are active. If the redundancy group fails over to the node 1, then the child links of all redundant Ethernet interfaces on node 1 become active.

NOTE: In the active/active chassis cluster configuration, the maximum number of redundancy groups is equal to the number of redundant Ethernet interfaces that you configure. In the active/backup chassis cluster configuration, the maximum number of redundancy groups supported is two.

Configuring redundant Ethernet interfaces on a device in Layer 2 transparent mode is similar to configuring redundant Ethernet interfaces on a device in Layer 3 route mode, with the following difference: the redundant Ethernet interface on a device in Layer 2 transparent mode is configured as a Layer 2 *logical interface*.

The redundant Ethernet interface may be configured as either an access interface (with a single VLAN ID assigned to untagged packets received on the interface) or as a trunk interface (with a list of VLAN IDs accepted on the interface and, optionally, a native-vlan-id for untagged packets received on the interface). Physical interfaces (one from each node in the chassis cluster) are bound as child interfaces to the parent redundant Ethernet interface.

In Layer 2 transparent mode, MAC learning is based on the redundant Ethernet interface. The MAC table is synchronized across redundant Ethernet interfaces and Services Processing Units (SPUs) between the pair of chassis cluster devices.

The IRB interface is used only for management traffic, and it cannot be assigned to any redundant Ethernet interface or redundancy group.

All Junos OS screen options that are available for a single, nonclustered device are available for devices in Layer 2 transparent mode chassis clusters.

NOTE: Spanning Tree Protocols (STPs) are not supported for Layer 2 transparent mode. You must ensure that there are no loop connections in the deployment topology.

Configuring Out-of-Band Management on SRX Series Firewalls

You can configure the **fxp0** out-of-band management interface on the SRX Series Firewall as a Layer 3 interface, even if Layer 2 interfaces are defined on the device. With the exception of the **fxp0** interface, you can define Layer 2 and Layer 3 interfaces on the device's network ports.

NOTE: There is no fxp0 out-of-band management interface on the SRX300, SRX320, and SRX550M devices. (Platform support depends on the Junos OS release in your installation.)

Ethernet Switching

Ethernet switching forwards the Ethernet frames within or across the LAN segment (or VLAN) using the Ethernet MAC address information. Ethernet switching on the SRX1500 device is performed in the hardware using ASICs.

Starting in Junos OS Release 15.1X49-D40, use the `set protocols l2-learning global-mode(transparent-bridge | switching)` command to switch between the Layer 2 transparent bridge mode and Ethernet switching mode. After switching the mode, you must reboot the device for the configuration to take effect. Table 2 on page 12 describes the default Layer 2 global mode on SRX Series Firewalls.

Table 2: Default Layer 2 Global Mode on SRX Series Devices

Junos OS Release	Platforms	Default Layer 2 Global Mode	Details
Prior to Junos OS Release 15.1X49-D50 and Junos OS Release 17.3R1 onwards	SRX300, SRX320, SRX340, and SRX345	Switching mode	None
Junos OS Release 15.1X49-D50 to Junos OS Release 15.1X49-D90	SRX300, SRX320, SRX340, and SRX345	Switching mode	When you delete the Layer 2 global mode configuration on a device, the device is in transparent bridge mode.
Junos OS Release 15.1X49-D100 onwards	SRX300, SRX320, SRX340, SRX345, SRX550, and SRX550M	Switching mode	When you delete the Layer 2 global mode configuration on a device, the device is in switching mode. Configure the `set protocols l2-learning global-mode transparent-bridge` command under the `[edit]` hierarchy level to switch to transparent bridge mode. Reboot the device for the configuration to take effect.
Junos OS Release 15.1X49-D50 onwards	SRX1500	Transparent bridge mode	None

The Layer 2 protocol supported in switching mode is Link Aggregation Control Protocol (LACP).

You can configure Layer 2 transparent mode on a redundant Ethernet interface. Use the following commands to define a redundant Ethernet interface:

- `set interfaces interface-name ether-options redundant-parent reth-interface-name`
- `set interfaces reth-interface-name redundant-ether-options redundancy-group number`

Layer 2 Switching Exceptions on SRX Series Devices

The switching functions on the SRX Series Firewalls are similar to the switching features on Juniper Networks MX Series routers. However, the following Layer 2 networking features on MX Series routers are not supported on SRX Series Firewalls:

- Layer 2 control protocols—These protocols are used on MX Series routers for Rapid Spanning Tree Protocol (RSTP) or Multiple Spanning Tree Protocol (MSTP) in customer edge interfaces of a VPLS routing instance.
- Virtual switch routing instance—The virtual switching routing instance is used on MX Series routers to group one or more VLANs.
- Virtual private LAN services (VPLS) routing instance—The VPLS routing instance is used on MX Series routers for point-to-multipoint LAN implementations between a set of sites in a VPN.

SEE ALSO

> *global-mode (Protocols)*
>
> *l2-learning*

Understanding Unicast

Unicasting is the act of sending data from one node of the network to another. In contrast, multicast transmissions send traffic from one data node to multiple other data nodes.

Unknown unicast traffic consists of unicast frames with unknown destination MAC addresses. By default, the switch floods these unicast frames that are traveling in a VLAN to all interfaces that are members of the VLAN. Forwarding this type of traffic to interfaces on the switch can trigger a security issue. The LAN is suddenly flooded with packets, creating unnecessary traffic that leads to poor network performance or even a complete loss of network service. This is known as a traffic storm.

To prevent a storm, you can disable the flooding of unknown unicast packets to all interfaces by configuring one VLAN or all VLANs to forward any unknown unicast traffic to a specific trunk interface. (This channels the unknown unicast traffic to a single interface.)

SEE ALSO

Understanding Bridging and VLANs on Switches | 140

Understanding Layer 2 Broadcasting on Switches

In a Layer 2 network, *broadcasting* refers to sending traffic to all nodes on a network.

Layer 2 broadcast traffic stays within a local area network (LAN) boundary; known as the *broadcast domain*. Layer 2 broadcast traffic is sent to the broadcast domain using a MAC address of FF:FF:FF:FF:FF:FF. Every device in the broadcast domain recognizes this MAC address and passes the broadcast traffic on to other devices in the broadcast domain, if applicable. Broadcasting can be compared to unicasting (sending traffic to a single node) or multicasting (delivering traffic to a group of nodes simultaneously).

Layer 3 broadcast traffic, however, is sent to all devices in a network using a broadcast network address. For example, if your network address is 10.0.0.0, the broadcast network address is 10.255.255.255. In this case, only devices that belong to the 10.0.0.0 network receive the Layer 3 broadcast traffic. Devices that do not belong to this network drop the traffic.

Broadcasting is used in the following situations:

- Address Resolution Protocol (ARP) uses broadcasting to map MAC addresses to IP addresses. ARP dynamically binds the IP address (the logical address) to the correct MAC address. Before IP unicast packets can be sent, ARP discovers the MAC address used by the Ethernet interface where the IP address is configured.
- Dynamic Host Configuration Protocol (DHCP) uses broadcasting to dynamically assign IP addresses to hosts on a network segment or subnet.
- Routing protocols use broadcasting to advertise routes.

Excessive broadcast traffic can sometimes create a broadcast storm. A broadcast storm occurs when messages are broadcast on a network and each message prompts a receiving node to respond by broadcasting its own messages on the network. This, in turn, prompts further responses that create a snowball effect. The LAN is suddenly flooded with packets, creating unnecessary traffic that leads to poor network performance or even a complete loss of network service.

SEE ALSO

Understanding Bridging and VLANs on Switches | 140

Using the Enhanced Layer 2 Software CLI

IN THIS SECTION

Enhanced Layer 2 Software (ELS) provides a uniform CLI for configuring and monitoring Layer 2 features on QFX Series switches, EX Series switches, and other Juniper Networks devices. With ELS, you configure Layer 2 features in the same way on all these Juniper Networks devices.

This topic explains how to know if your platform is running ELS. It also explains how to perform some common tasks using the ELS style of configuration.

Understanding Which Devices Support ELS

ELS is automatically supported if your device is running a Junos OS release that supports it. You do not need to take any action to enable ELS, and you cannot disable ELS. See Feature Explorer for information about which platforms and releases support ELS.

Understanding How to Configure Layer 2 Features Using ELS

IN THIS SECTION

Because ELS provides a uniform CLI, you can now perform the following tasks on supported devices in the same way:

Configuring a VLAN

You can configure one or more VLANs to perform Layer 2 bridging. The Layer 2 bridging functions include integrated routing and bridging (IRB) for support for Layer 2 bridging and Layer 3 IP routing on the same interface. EX Series and QFX Series switches can function as Layer 2 switches, each with multiple bridging, or broadcast, domains that participate in the same Layer 2 network. You can also configure Layer 3 routing support for a VLAN.

To configure a VLAN:

1. Create the VLAN by setting a unique VLAN name and configuring the VLAN ID:

```
[edit]
user@host# set vlans vlan-name vlan-id vlan-id-number
```

   Using the VLAN ID list option, you can optionally specify a range of VLAN IDs.

```
[edit]
user@host# set vlans vlan-name vlan-id-list vlan-ids | vlan-id--vlan-id
```

2. Assign at least one interface to the VLAN:

```
[edit]
user@host# set interface interface-name family ethernet-switching vlan members vlan-name
```

Configuring the Native VLAN Identifier

EX Series and QFX Series switches support receiving and forwarding routed or bridged Ethernet frames with 802.1Q VLAN tags. Typically, trunk ports, which connect switches to each other, accept untagged control packets, but do not accept untagged data packets. You can enable a trunk port to accept untagged data packets by configuring a native VLAN ID on the interface on which you want the untagged data packets to be received.

To configure the native VLAN ID:

1. On the interface on which you want untagged data packets to be received, set the interface mode to `trunk`, which specifies that the interface is in multiple VLANs and can multiplex traffic between different VLANs.

```
[edit interfaces]
user@host# set interface-name unit logical-unit-number family ethernet-switching interface-mode trunk
```

2. Configure the native VLAN ID and assign the interface to the native VLAN ID:

```
[edit interfaces]
user@host# set interface-name native-vlan-id number
```

3. Assign the interface to the native VLAN ID:

```
[edit interfaces]
user@host# set interface-name unit logical-unit-number family ethernet-switching vlan members native-vlan-id-number
```

Configuring Layer 2 Interfaces

To ensure that your high-traffic network is tuned for optimal performance, explicitly configure some settings on the switch's network interfaces.

To configure a Gigabit Ethernet interface or a 10-Gigabit Ethernet interface as a `trunk` interface:

```
[edit]
user@host# set interfaces interface-name unit logical-unit-number family ethernet-switching interface-mode trunk
```

To configure a Gigabit Ethernet interface or a 10-Gigabit Ethernet interface as a `access` interface:

```
[edit]
 user@host# set interfaces interface-name unit logical-unit-number family ethernet-switching interface-mode access
```

To assign an interface to VLAN:

```
[edit interfaces]
 user@host# set interface-name unit logical-unit-number family ethernet-switching vlan members
[all | vlan-names | vlan-ids]
```

Configuring Layer 3 Interfaces

To configure a Layer 3 interface, you must assign an IP address to the interface. You assign an address to an interface by specifying the address when you configure the protocol family. For the `inet` or `inet6` family, configure the interface IP address.

You can configure interfaces with a 32-bit IP version 4 (IPv4) address and optionally with a destination prefix, sometimes called a subnet mask. An IPv4 address utilizes a 4-octet dotted decimal address syntax (for example, 192.168.1.1). An IPv4 address with destination prefix utilizes a 4-octet dotted decimal address syntax with a destination prefix appended (for example, 192.168.1.1/16).

To specify an IP4 address for the logical unit:

```
[edit]
user@host# set interfaces interface-name unit logical-unit-number family inet address ip-address
```

You represent IP version 6 (IPv6) addresses in hexadecimal notation by using a colon-separated list of 16-bit values. You assign a 128-bit IPv6 address to an interface.

To specify an IP6 address for the logical unit:

```
[edit]
user@host# set interfaces interface-name unit logical-unit-number family inet6 address ip-address
```

Configuring an IRB Interface

Integrated routing and bridging (IRB) provides support for Layer 2 bridging and Layer 3 IP routing on the same interface. IRB enables you to route packets to another routed interface or to another VLAN that has a Layer 3 protocol configured. IRB interfaces enable the device to recognize packets that are being sent to local addresses so that they are bridged (switched) whenever possible and are routed only when necessary. Whenever packets can be switched instead of routed, several layers of processing are eliminated. An interface named irb functions as a logical router on which you can configure a Layer 3 logical interface for VLAN. For redundancy, you can combine an IRB interface with implementations of

the Virtual Router Redundancy Protocol (VRRP) in both bridging and virtual private LAN service (VPLS) environments.

To configure an IRB interface:

1. Create a Layer 2 VLAN by assigning it a name and a VLAN ID:

```
[edit]
 user@host# set vlans vlan-name vlan-id vlan-id
```

2. Create an IRB logical interface:

```
[edit]
user@host# set interface irb unit logical-unit-number family inet address ip-address
```

3. Associate the IRB interface with the VLAN:

```
[edit]
user@host# set vlans vlan-name l3-interface irb.logical-unit-number
```

Configuring an Aggregated Ethernet Interface and Configuring LACP on That Interface

Use the link aggregation feature to aggregate one or more links to form a virtual link or link aggregation group (LAG). The MAC client can treat this virtual link as if it were a single link to increase bandwidth, provide graceful degradation as failure occurs, and increase availability.

To configure an aggregated Ethernet interface:

1. Specify the number of aggregated Ethernet interfaces to be created:

```
[edit chassis]
user@host# set aggregated-devices ethernet device-count number
```

2. Specify the name of the link aggregation group interface:

```
[edit]
user@host# set interfaces aex
```

3. Specify the minimum number of links for the aggregated Ethernet interface (aex)– that is, the defined bundle– to be labeled *up*:

```
[edit interfaces]
user@host# set aex aggregated-ether-options minimum-links number
```

4. Specify the link speed for the aggregated Ethernet bundle:

```
[edit interfaces]
user@host# set aex aggregated-ether-options link-speed link-speed
```

5. Specify the members to be included within the aggregated Ethernet bundle:

```
[edit interfaces]
 user@host# set interface-name ether-options 802.3ad aex
```

6. Specify an interface family for the aggregated Ethernet bundle:

```
[edit interfaces]
user@host# set aex unit 0 family inet address ip-address
```

For aggregated Ethernet interfaces on the device, you can configure the Link Aggregation Control Protocol (LACP). LACP bundles several physical interfaces to form one logical interface. You can configure aggregated Ethernet with or without LACP enabled.

When LACP is enabled, the local and remote sides of the aggregated Ethernet links exchange protocol data units (PDUs), containing information about the state of the link. You can configure Ethernet links to actively transmit PDUs, or you can configure the links to passively transmit them, sending out LACP PDUs only when they receive them from another link. One side of the link must be configured as active for the link to be up.

To configure LACP:

1. Enable one side of the aggregated Ethernet link as active:

```
[edit interfaces]
user@host# set aex aggregated-ether-options lacp active
```

2. Specify the interval at which the interfaces send LACP packets:

```
[edit interfaces]
 user@host# set aex aggregated-ether-options lacp periodic interval
```

Understanding ELS Configuration Statement and Command Changes

IN THIS SECTION

ELS was introduced in Junos OS Release 12.3R2 for EX9200 switches. ELS changes the CLI for some of the Layer 2 features on supported EX Series and QFX Series switches.

The following sections provide a list of existing commands that were moved to new hierarchy levels or changed on EX Series switches as part of this CLI enhancement effort. These sections are provided as a high-level reference only. For detailed information about these commands, use the links to the configuration statements provided or see the technical documentation.

Changes to the ethernet-switching-options Hierarchy Level

This section outlines the changes to the `ethernet-switching-options` hierarchy level.

NOTE: The `ethernet-switching-options` hierarchy level has been renamed as **`switch-options`**.

Table 3: Renaming the ethernet-switching-options hierarchy

Original Hierarchy	Changed Hierarchy
ethernet-switching-options { authentication-whitelist { ... } }	**switch-options** { ... authentication-whitelist { ... } }
ethernet-switching-options { interfaces *interface-name* { no-mac-learning; ... } }	switch-options { interfaces *interface-name* { no-mac-learning; ... } }
ethernet-switching-options { unknown-unicast-forwarding { (...) } }	switch-options { unknown-unicast-forwarding { (...) } }
ethernet-switching-options { voip { interface (all | [*interface-name* | access-ports]) { **forwarding-class** (assured-forwarding | best-effort | expedited-forwarding | network-control); **vlan** *vlan-name*; ... } } }	**switch-options** { voip { interface (all | [*interface-name* | access-ports]) { **forwarding-class** (assured-forwarding | best-effort | expedited-forwarding | network-control); **vlan** *vlan-name*; ... } } }

Table 4: RTG Statements

Original Hierarchy	Changed Hierarchy
ethernet-switching-options { redundant-trunk-group { group *name* { **description**; **interface** *interface-name* { **primary**; } preempt-cutover-timer *seconds*; ... } } }	switch-options { redundant-trunk-group { **group** *name* { **description**; **interface** *interface-name* { **primary**; } **preempt-cutover-timer** *seconds*; ... } } }

Table 5: Deleted Statements

Original Hierarchy	Changed Hierarchy
ethernet-switching-options { **mac-notification** { notification-interval *seconds*; ... } }	The statements have been removed from the switch-options hierarchy.
ethernet-switching-options { traceoptions { **file** *filename* <files *number*> <no-stamp> <replace> <size *size*> <world-readable | no-world-readable>; **flag** *flag* <disable>; ... } }	The statements have been removed from the switch-options hierarchy.

Table 5: Deleted Statements *(Continued)*

Original Hierarchy	Changed Hierarchy
ethernet-switching-options { **port-error-disable** { **disable-timeout** ***timeout***; ... } }	**NOTE**: The port-error-disable statement has been replaced with a new statement. **interfaces** *interface-name* family **ethernet-switching** { recovery-timeout *seconds*; }

Changes to the Port Mirroring Hierarchy Level

NOTE: Statements have moved from the ethernet-switching-options hierarchy level to the forwarding-options hierarchy level.

Table 6: Port Mirroring hierarchy

Original Hierarchy	Changed Hierarchy
ethernet-switching-options { analyzer { *name* { ... } } }	forwarding-options { analyzer { *name* { ... } } }

Changes to the Layer 2 Control Protocol Hierarchy Level

The Layer 2 control protocol statements have moved from the ethernet-switching-options hierarchy to the protocols hierarchy.

Table 7: Layer 2 Control Protocol

Original Hierarchy	Changed Hierarchy
ethernet-switching-options { bpdu-block { ... } }	protocols { layer2-control { **bpdu-block** { ... } } }

Changes to the dot1q-tunneling Statement

The dot1q-tunneling statement has been replaced with a new statement and moved to a different hierarchy level.

Table 8: dot1q-tunneling

Original Hierarchy	Changed Hierarchy
ethernet-switching-options { **dot1q-tunneling** { **ether-type** (0x8100 | 0x88a8 | 0x9100); ... } }	interfaces *interface-name* { aggregated-ether-options ether-options { ethernet-switch-profile { tag-protocol-id [*tpids*]; } } } interfaces *interface-name* { aggregated-ether-options { ethernet-switch-profile { tag-protocol-id [*tpids*]; } } }

Changes to the L2 Learning Protocol

The `mac-table-aging-time` statement has been replaced with a new statement and moved to a different hierarchy level.

Table 9: mac-table-aging-time statement

Original Hierarchy	Changed Hierarchy
`ethernet-switching-options {` `  mac-table-aging-time` *seconds*; `  ...` `}`	`protocols {` `  l2-learning {` `  global-mac-table-aging-time` *seconds*; `  ...` `  }` `}`

Changes to Nonstop Bridging

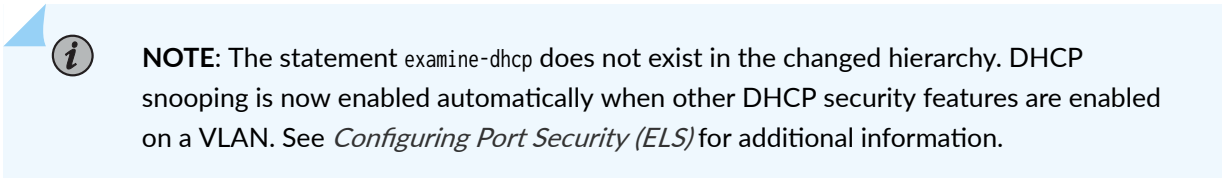
The `nonstop-bridging` statement has moved to a different hierarchy level.

Table 10: Nonstop Bridging statement

Original Hierarchy	Changed Hierarchy
`ethernet-switching-options {` `  nonstop-bridging;` `}`	`protocols {` `  layer2-control {` `  nonstop-bridging {` `  }` `  }` `}`

Changes to Port Security and DHCP Snooping

Port security and DHCP snooping statements have moved to different hierarchy levels.



NOTE: The statement `examine-dhcp` does not exist in the changed hierarchy. DHCP snooping is now enabled automatically when other DHCP security features are enabled on a VLAN. See *Configuring Port Security (ELS)* for additional information.

Table 11: Port Security statements

Original Hierarchy	Changed Hierarchy
ethernet-switching-options { secure-access-port { interface (all | *interface-name*) { (dhcp-trusted | no-dhcp-trusted); static-ip *ip-address* { mac *mac-address*; vlan *vlan-name*; } } vlan (all | *vlan-name*) { (arp-inspection | no-arp-inspection); dhcp-option82 { disable; circuit-id { prefix *hostname*; use-interface-description; use-vlan-id; } remote-id { prefix (*hostname* | mac | none); use-interface-description; use-string *string*; } vendor-id [*string*]; } (examine-dhcp | no-examine-dhcp); } (ip-source-guard | no-ip-source-guard); } }	vlans *vlan-name* forwarding-options{ dhcp-security { arp-inspection; group ***group-name*** { interface*interface-name* { static-ip *ip-address* { mac *mac-address*; } } overrides { no-option82; trusted; } } ip-source-guard; no-dhcp-snooping; option-82 { circuit-id { prefix { host-name; routing-instance-name; } use-interface-description (device | logical); use-vlan-id; } remote-id { host-name; use-interface-description (device | logical); use-string *string*; } vendor-id { use-string *string*; } } }

TIP: For allowed mac configuration, the original hierarchy statement `set ethernet-switching-options secure-access-port interface ge-0/0/2 allowed-mac 00:05:85:3A:82:8` is replaced by the ELS command `set interfaces ge-0/0/2 unit 0 accept-source-mac mac-address 00:05:85:3A:82:8`

NOTE: DHCP snooping statements have moved to a different hierarchy level.

Table 12: DHCP Snooping Statements

Original Hierarchy	Changed Hierarchy
ethernet-switching-options { secure-access-port { **dhcp-snooping-file** { location *local_pathname* | *remote_URL*; timeout *seconds*; write-interval *seconds*; }	system [processes [dhcp-service dhcp-snooping-file *local_pathname* | *remote_URL*; write-interval *interval*; } }

Changes to Configuring VLANs

The statements for configuring VLANs have moved to a different hierarchy level.

NOTE: Starting with Junos OS Release 14.1X53-D10 for EX4300 and EX4600 switches, when enabling *x*STP, you can enable it on some or all interfaces included in a VLAN. For example, if you configure VLAN 100 to include interfaces ge-0/0/0, ge-0/0/1, and ge-0/0/2, and you want to enable MSTP on interfaces ge-0/0/0 and ge-0/0/2, you can specify the `set protocols mstp interface ge-0/0/0` and `set protocols mstp interface ge-0/0/2` commands. In this example, you did not explicitly enable MSTP on interface ge-0/0/1; therefore, MSTP is not enabled on this interface.

Table 13: VLAN hierarchy

Original Hierarchy	Changed Hierarchy
ethernet-switching-options { secure-access-port vlan (all | *vlan-name*{ mac-move-limit } 	vlans *vlan-name* switch-options { mac-move-limit }
ethernet-switching-options { **static** { **vlan** *vlan-id* { mac *mac-address* next-hop *interface-name*; ... } } }	**NOTE**: Statement is replaced with a new statement and has moved to a different hierarchy level. vlans { *vlan-name* { switch-options { interface *interface-name* { static-mac *mac-address*; ... } } } }
vlans { *vlan-name* { interface *interface-name* { **egress**; ingress; **mapping** (native (push | swap) | policy | *tag* (push | swap)); **pvlan-trunk**; ... } } }	These statements have been removed. You can assign interfaces to a VLAN using the [edit interfaces *interface-name* unit *logical-unit-number* family ethernet-switching vlan members *vlan-name*] hierarchy.

Table 13: VLAN hierarchy *(Continued)*

Original Hierarchy	Changed Hierarchy
vlans { *vlan-name* { isolation-id *id-number*; ... } }	Statements have been removed.
vlans { *vlan-name* { interface **vlan**.*logical-interface-number*; ... } }	**NOTE**: Syntax is changed. vlans { *vlan-name* { interface **irb**.*logical-interface-number*; ... } }
vlans { *vlan-name* { l3-interface-ingress-counting *layer-3-interface-name*; ... } }	Statement is removed. Ingress traffic is automatically tracked.
vlans { *vlan-name* { no-local-switching; ... } }	Statement is removed.

Table 13: VLAN hierarchy *(Continued)*

Original Hierarchy	Changed Hierarchy
vlans { *vlan-name* { **no-mac-learning**; ... } }	Statement has been moved to different hierarchy. vlans { *vlan-name* { switch-options { no-mac-learning *limit* ... } } }
vlans { *vlan-name* { **primary-vlan** *vlan-name*; ... } }	Statement has been removed.
vlans { *vlan-name* { vlan-prune; ... } }	Statement is removed.
vlans { *vlan-name* { **vlan-range** *vlan-id-low-vlan-id-high*; ... } }	**NOTE**: Statement has been replaced with a new statement. vlans { *vlan-name* { vlan-id-list [*vlan-id-numbers*]; ... } }

Table 13: VLAN hierarchy *(Continued)*

Original Hierarchy	Changed Hierarchy
vlans { *vlan-name* { l3-interface **vlan**.*logical-interface-number*; ... } }	**NOTE**: Syntax is changed. vlans { *vlan-name* { interface **irb**.*logical-interface-number*; ... } }

Table 14: Statements Moved to a Different Hierarchy

Original Hierarchy	Changed Hierarchy
vlans { *vlan-name* { **dot1q-tunneling** { **customer-vlans** (*id* | native | *range*); **layer2-protocol-tunneling** all | *protocol-name* { drop-threshold *number*; shutdown-threshold *number*; ... } } } }	For **dot1q-tunneling**: interface *interface-name* { encapsulation extended-vlan-bridge; flexible-vlan-tagging; native-vlan-id *number*; unit *logical-unit-number* { input-vlan-map *action*; output-vlan-map *action*; vlan-id *number*; vlan-id-list [*vlan-id vlan-id-vlan-id*]; } } For **layer2-protocol-tunneling** (MAC rewrite enabled on an interface): protocols { layer2-control { mac-rewrite { interface *interface-name* { protocol { ... } } } } }

Table 14: Statements Moved to a Different Hierarchy *(Continued)*

Original Hierarchy	Changed Hierarchy
vlans { *vlan-name* { filter{ **input** *filter-name* **output** *filter-name*; ... } } }	vlans { *vlan-name* { forwarding-options { filter{ **input** *filter-name* **output** *filter-name*; ... } } } }
vlans { *vlan-name* { mac-limit *limit* **action** *action*; ... } }	vlans { *vlan-name* { switch-options { interface-mac-limit *limit* { packet-action *action*; ... } } } } vlans { *vlan-name* { switch-options { interface *interface-name* { interface-mac-limit *limit* { packet-action *action*; ... } } } } }

Table 14: Statements Moved to a Different Hierarchy *(Continued)*

Original Hierarchy	Changed Hierarchy
vlans { *vlan-name* { mac-table-aging-time *seconds*; ... } }	protocols { l2-learning { global-mac-table-aging-time *seconds*; ... } }

Changes to Storm Control Profiles

Storm control is configured in two steps. The first step is to create a storm control profile at the [edit forwarding-options] hierarchy level, and the second step is to bind the profile to a logical interface at the [edit interfaces] hierarchy level. See Example: Configuring Storm Control to Prevent Network Outages on EX Series Switches for the changed procedure.

Table 15: Changes to the Storm Control Profile hierarchy level

Original Hierarchy	Changed Hierarchy
ethernet-switching-options { storm-control { (...) } }	forwarding-options { storm-control-profiles *profile-name* { (...) } } interfaces *interface-name* unit *number* family ethernet-switching { storm-control *storm-control-profile*; }

Changes to the Interfaces Hierarchy

NOTE: Statements have been moved to a different hierarchy.

Table 16: Changes to the Interfaces hierarchy

Original Hierarchy	Changed Hierarchy
interfaces *interface-name* { ether-options { link-mode *mode*; speed (auto-negotiation | *speed*) } }	interfaces *interface-name* { link-mode *mode*; speed *speed*) }
interfaces *interface-name* { unit *logical-unit-number* { family ethernet-switching { **native-vlan-id** *vlan-id* } } }	interfaces *interface-name* { native-vlan-id *vlan-id* }
interfaces *interface-name* { unit *logical-unit-number* { family ethernet-switching { port-mode *mode* } } }	**NOTE**: Statement has been replaced with a new statement. interfaces *interface-name* { unit *logical-unit-number* { family ethernet-switching { interface-mode *mode* } } }
interfaces **vlan**	**NOTE**: Statement has been replaced with a new statement. **interfaces irb**

Changes to IGMP Snooping

Table 17: IGMP Snooping hierarchy

Original Hierarchy	Changed Hierarchy

```
protocols {
    igmp-snooping {
        traceoptions  {
            file filename <files number> <no-stamp>
<replace> <size maximum-file-size> <world-readable |
no-world-readable>;
            flag flag <flag-modifier> <disable>;
        }
        vlan (all | vlan-identifier) {
            disable;
            data-forwarding {
                receiver {
                    install;
                    source-vlans vlan-name;
                }
                source {
                    groups ip-address;
                }
            }
            immediate-leave;
            interface (all | interface-name) {
                multicast-router-interface;
                static {
                    group multicast-ip-address;
                }
            }
            proxy {
                source-address ip-address;
            }
            robust-count number;
        }
    }
}
```

```
protocols {
    igmp-snooping {
        vlan vlan-name {
                data-forwarding {
                    receiver {
                        install;
                        source-list vlan-name;
                        translate;
                    }
                    source {
                        groups ip-address;
                    }
                }
            immediate-leave;
                interface (all | interface-name) {
                    group-limit <1..65535>
                    host-only-interface
                    multicast-router-interface;
                    immediate-leave;
                    static {
                        group multicast-ip-address {
                            source <>
                        }
                    }
                }
            }
            l2-querier {
                source-address ip-address;
            }
            proxy {
                source-address ip-address;
            }
            query-interval number;
            query-last-member-interval number;
            query-response-interval number;
            robust-count number;
            traceoptions  {
                file filename <files number> <no-
stamp> <replace> <size maximum-file-size> <world-
```

Table 17: IGMP Snooping hierarchy *(Continued)*

Original Hierarchy	Changed Hierarchy
	readable | no-world-readable>; flag *flag* <*flag-modifier*>; } } } }

Enhanced Layer 2 CLI Configuration Statement and Command Changes for Security Devices

Starting in Junos OS Release 15.1X49-D10 and Junos OS Release 17.3R1, some Layer 2 CLI configuration statements are enhanced, and some commands are changed. Table 18 on page 39 and Table 19 on page 41 provide lists of existing commands that have been moved to new hierarchies or changed on SRX Series Firewalls as part of this CLI enhancement effort. The tables are provided as a high-level reference only. For detailed information about these commands, see CLI Explorer.

Table 18: Enhanced Layer 2 Configuration Statement Changes

Original Hierarchy	Changed Hierarchy	Hierarchy Level	Change Description
bridge-domains *bridge-domain--name* { ... } }	**vlans *vlans-name* {** ... } }	[edit]	Hierarchy renamed.
bridge-domains *bridge-domain--name* { **vlan-id-list [*vlan-id*]** ; }	vlans *vlans-name* { **vlan members [*vlan-id*]** ; }	[edit vlans *vlans-name*]	Statement renamed.

Table 18: Enhanced Layer 2 Configuration Statement Changes *(Continued)*

Original Hierarchy	Changed Hierarchy	Hierarchy Level	Change Description
bridge-options { interface *interface-name* { encapsulation-type; ignore-encapsulation-mismatch; pseudowire-status-tlv; static-mac *mac-address* { vlan-id *vlan-id*; } } mac-table-aging-time *seconds*; mac-table-size { *number*; packet-action drop; } }	**switch-options** { interface *interface-name* { encapsulation-type; ignore-encapsulation-mismatch; pseudowire-status-tlv; static-mac *mac-address* { vlan-id *vlan-id*; } } mac-table-aging-time *seconds*; mac-table-size { *number*; packet-action drop; } }	[edit vlans *vlans-name*]	Statement renamed.
bridge { block-non-ip-all; bpdu-vlan-flooding; bypass-non-ip-unicast; no-packet-flooding { no-trace-route; } }	**ethernet-switching** { block-non-ip-all; bpdu-vlan-flooding; bypass-non-ip-unicast; no-packet-flooding { no-trace-route; } }	[edit security flow]	Statement renamed.
family { **bridge** { bridge-domain-type (svlan| bvlan); ...	family { **ethernet-switching** { ...	[edit interfaces *interface-name*] unit *unit-number*	Hierarchy renamed.

Table 18: Enhanced Layer 2 Configuration Statement Changes *(Continued)*

Original Hierarchy	Changed Hierarchy	Hierarchy Level	Change Description
... **routing-interface** irb.0; ...	... **l3-interface** irb.0; ...	[edit vlans *vlans-name*]	Statement renamed.

Table 19: Enhanced Layer 2 Operational Command Changes

Original Operational Command	Modified Operational Command
clear bridge mac-table	clear ethernet-switching table
clear bridge mac-table persistent-learning	clear ethernet-switching table persistent-learning
show bridge domain	show vlans
show bridge mac-table	show ethernet-switching table
show l2-learning interface	show ethernet-switching interface

NOTE: There is no fxp0 out-of-band management interface on the SRX300, SRX320, and SRX500HM devices. (Platform support depends on the Junos OS release in your installation.)

SEE ALSO

Understanding Switching Modes on Security Devices | **1060**

Layer 2 Next Generation Mode for ACX Series

The Layer 2 Next Generation mode, also called Enhanced Layer 2 Software (ELS), is supported on ACX5048, ACX5096, and ACX5448 routers for configuring Layer 2 features. The Layer 2 CLI configurations and show commands for ACX5048, ACX5096, ACX5448, ACX710, ACX7100, ACX7024, ACX7332, ACX7348, and ACX7509 routers differ from those for other ACX Series routers (ACX1000, ACX1100, ACX2000, ACX2100, ACX2200, and ACX4000) and MX Series routers.

Table 20 on page 42 shows the differences in CLI hierarchy for configuring Layer 2 features in Layer 2 next generation mode.

Table 20: Differences in CLI Hierarchy for Layer 2 Features in Layer 2 Next Generation Mode

Feature	ACX1000, ACX1100, ACX2000, ACX2100, ACX2200, ACX4000, and MX Series Routers	ACX5048, ACX5096, ACX5448, ACX710, ACX7100, ACX7024, ACX7332, ACX7348, and ACX7509 Routers
Bridge Domain	[edit bridge-domains *bridge-domain-name*]	[edit vlans *vlan-name*]
Family bridge	[edit interfaces *interface-name* unit *unit-number* family bridge]	[edit interfaces *interface-name* unit *unit-number* family ethernet-switching]
Layer 2 options	[edit bridge-domains *bridge-domain-name* bridge-options]	[edit vlans *vlan-name* switch-options]
Ethernet options	[edit interfaces *interface-name* gigether-options]	[edit interfaces *interface-name* ether-options]
Integrated routing and bridging (IRB)	[edit bridge-domains *bridge-domain-name*] routing-interface irb.*unit*;	[edit vlans *vlan-name*] l3-interface irb.*unit*;
Storm control	[edit vlans *vlan-name* forwarding-options flood filter *filter-name*]	[edit forwarding-options storm-control-profiles] [edit interfaces *interface-name* ether-options] storm-control *name*; recovery-timeout *interval*;

Table 20: Differences in CLI Hierarchy for Layer 2 Features in Layer 2 Next Generation Mode *(Continued)*

Feature	ACX1000, ACX1100, ACX2000, ACX2100, ACX2200, ACX4000, and MX Series Routers	ACX5048, ACX5096, ACX5448, ACX710, ACX7100, ACX7024, ACX7332, ACX7348, and ACX7509 Routers
Internet Group Management Protocol (IGMP) snooping	[edit bridge-domains *bridge-domain-name* protocols igmp-snooping]	[edit protocols igmp-snooping vlan *vlan-name*]
Family bridge firewall filter	[edit firewall family bridge]	[edit firewall family ethernet-switching]

Table 21 on page 43 shows the differences in show commands for Layer 2 features in Layer 2 next generation mode.

Table 21: Differences in show Commands for Layer 2 Features in Layer 2 Next Generation Mode

Feature	ACX1000, ACX1100, ACX2000, ACX2100, ACX2200, ACX4000, and MX Series Routers	ACX5048, ACX5096, ACX5448, ACX710, ACX7100, ACX7024, ACX7332, ACX7348, and ACX7509 Routers
VLAN	show bridge-domain	show vlans
MAC table	show bridge mac-table	show ethernet-switching table
MAC table options	show bridge mac-table (MAC address, bridge-domain name, interface, VLAN ID, and instance)	show ethernet-switching table
Switch port listing with VLAN assignments	show l2-learning interface	show ethernet-switching interfaces
Kernel state of flush database	show route forwarding-table family bridge	show route forwarding-table family ethernet-switching

SEE ALSO

- Storm Control on ACX Series Routers Overview
- *Layer 2 Bridge Domains on ACX Series Overview*
- *Guidelines for Configuring Firewall Filters*
- *IGMP Snooping and Bridge Domains*
- *Understanding Ethernet Link Aggregation on ACX Series Routers*

Flexible Ethernet Services Encapsulation to Support the Service Provider and Enterprise Styles of Configuration on ACX7000 Series of Routers

Flexible Ethernet services is a type of *encapsulation* that enables a physical interface to specify Ethernet encapsulations at the logical interface level. Each logical interface can have a different Ethernet encapsulation. Defining multiple per-unit Ethernet encapsulations makes it easier to customize Ethernet-based services to multiple hosts connected to the same physical interface.

An Ethernet interface that is not encapsulated with flexible Ethernet services and is operating in Layer 2 mode is limited to a single logical interface unit (0). Bridging is enabled on the interface by configuring `ethernet-switching` as the interface family on unit 0. The `ethernet-switching` family can be configured only on logical interface unit 0, and no other logical units can be defined on that interface.

Some switching features, however, cannot be configured on logical interface unit 0. Features such as Q-in-Q tunneling require the logical interface to transmit VLAN-tagged frames. To enable a logical interface to receive and forward Ethernet frames tagged with a matching VLAN ID, you must bind the logical interface to that VLAN. These features must be configured on a logical interface unit other than 0, because you cannot bind a VLAN ID to unit 0.

When you encapsulate an interface by using flexible Ethernet services, you can configure a logical interface unit other than 0 with `family ethernet-switching`. You can also configure other logical interfaces on that same interface with different types of Ethernet encapsulations. This enables logical interfaces that are bound to a VLAN ID to coexist with logical interfaces configured with `family ethernet-switching`.

The `flexible-ethernet-services` statement allows configuration of both service-provider-style logical interfaces and enterprise-style logical interfaces.

For example, if you configure PVLAN on the same physical interface on which you are configuring Q-in-Q tunneling, you can use flexible ethernet services to support the enterprise style of configuration for PVLAN, using `family ethernet-switching`, along with `vlan-bridge` encapsulation for Q-in-Q tunneling.

BEST PRACTICE: We recommend you configure the following statements using groups when configuring devices that function as hardware VTEPs:

- set interfaces *interface-name* flexible-vlan-tagging
- set interfaces *interface-name* encapsulation extended-vlan-bridge
- set interfaces *interface-name* native-vlan-id *vlan-id*

Service Provider Style of Configuration

To configure the interface to support the service provider style of configuration:

1. Enable flexible Ethernet services encapsulation on the interface.

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

2. Enable the interface to transmit packets with 802.1Q VLAN single-tagged and dual-tagged frames:

```
[edit interfaces interface-name]
user@host# set flexible-vlan-tagging
```

3. Configure a logical interface (unit) on the interface:

```
[edit interfaces interface-name]
user@host# set unit unit-number
```

NOTE: Do not use logical interface unit 0. You must later bind a VLAN tag ID to the unit you specify in this step, and you cannot bind a VLAN tag ID to unit 0. It is a best practice to match the unit number to the VLAN ID to which the interface is bound.

4. Encapsulate the logical interface for service provider style bridging configuration—for example, use `vlan-bridge` encapsulation on an interface to be used for Q-in-Q tunneling. (If you were configuring the

interface only for Q-in-Q tunneling, you would use `encapsulation extended-vlan-bridge` on the *physical* interface.)

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

5. Bind the logical interface from the preceding step to a VLAN ID:

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

6. Configure another logical interface. (If you were configuring just PVLAN, we would recommend that you configure a single logical interface for all PVLAN domains on an interface.)

```
[edit interfaces interface-name]
user@host# set unit unit-number
```

Enterprise Style of Configuration

To configure the interface to support the enterprise style of configuration:

1. Enable flexible Ethernet services encapsulation on the interface.

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

2. Enable the interface to transmit packets with 802.1Q VLAN single-tagged and dual-tagged frames:

```
[edit interfaces interface-name]
user@host# set flexible-vlan-tagging
```

3. Configure a logical interface (unit) on the interface:

```
[edit interfaces interface-name]
user@host# set unit unit-number
```

4. Enable the logical interface in the preceding step for enterprise style bridging configuration:

```
[edit interfaces interface-name]
user@host# set unit unit-number family ethernet-switching
```

5. Assign VLAN membership to the logical interface:

```
[edit interfaces interface-name]
user@host# set unit unit-number family ethernet-switching vlan members vlan-id
```

6. Configure the interface as a trunk interface to transmit frames with 802.1Q VLAN tags:

```
[edit interfaces interface-name]
user@host# set unit unit-number family ethernet-switching interface-mode trunk
```

Change History Table

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

Release	Description
15.1X49-D40	Starting in Junos OS Release 15.1X49-D40, use the `set protocols l2-learning global-mode(transparent-bridge | switching)` command to switch between the Layer 2 transparent bridge mode and Ethernet switching mode.
15.1X49-D10	Starting in Junos OS Release 15.1X49-D10 and Junos OS Release 17.3R1, some Layer 2 CLI configuration statements are enhanced, and some commands are changed.

2

CHAPTER

Flexible Ethernet Services Encapsulation

IN THIS CHAPTER

Flexible Ethernet Services Encapsulation

IN THIS SECTION

Flexible Ethernet services is a type of encapsulation that enables a physical interface to support different types of Ethernet encapsulations at the logical interface level. You can configure the Flexible Ethernet services encapsulation to support the service provider and the enterprise-style configuration. The below topics discuss the overview of flexible Ethernet services encapsulation and its configuration details.

Understanding Flexible Ethernet Services Encapsulation on Switches

IN THIS SECTION

Junos OS supports two different styles of configuration for switch interfaces: the service provider style and the enterprise style. The service provider style requires more configuration but provides greater flexibility. The enterprise style is easier to configure but offers less functionality. Each configuration style requires a different Ethernet encapsulation type. You can configure a physical interface to support both styles of configuration using flexible Ethernet services.

NOTE: On EX4300, QFX5100 (running Junos OS 16.1R5 or earlier), and QFX5200, the service provider style and enterprise style interface configurations are handled differently within Junos OS. If the service provider style and enterprise style interface configurations are mixed, the egress VLAN translation within the hardware can be incorrectly programmed leading to forwarding issues across the configured ports. Use the service provider style configuration in a Q-in-Q scenario. For all other scenarios, use the enterprise style configuration.

Flexible Ethernet services is a type of encapsulation that enables a physical interface to support different types of Ethernet encapsulations at the logical interface level. Defining multiple per-unit Ethernet encapsulations makes it easier to customize Ethernet-based services to multiple hosts connected to the same physical interface.

Service Provider Style

The service provider style of configuration allows for customization of Ethernet-based services at the logical interface level. Service providers typically have multiple customers connected to the same physical interface. Using the service provider style, you can configure multiple logical interfaces on the physical interface, and associate each unit with a different VLAN. This provides the flexibility to configure different services for each customer, but also requires more configuration, because each feature must be explicitly configured on the logical interface.

When configuring a physical interface to support only the service provider style, the physical interface must be encapsulated with the `extended-vlan-bridge` option to support bridging features. VLAN tagging must also be configured on the physical interface so that it can operate in trunk mode and transmit Ethernet frames with VLAN tags for multiple VLANs. Each logical interface is bound to a unique VLAN ID.

Enterprise Style

The enterprise style of configuration is designed to provide basic bridging functionality for consumers of Ethernet-based services. The isolation of services for different customers on a single port is not required, because each port is typically connected to a host or is providing a trunk to another switch.

With the enterprise style of configuration, logical interfaces are placed into Layer 2 mode by specifying `ethernet-switching` as the interface family. Without using flexible Ethernet services, `ethernet-switching` can only be configured on a single logical unit, unit 0. You cannot bind a VLAN ID to unit 0, because these interfaces operate either in trunk mode, which supports traffic with various VLAN tags, or in access mode, which supports untagged traffic.

Flexible Ethernet Services

The flexible Ethernet services encapsulation type enables a physical interface to support both styles of configuration. To support the service provider style, flexible Ethernet services allows for encapsulations to be configured at the logical interface level instead of the physical interface. To support the enterprise style, flexible Ethernet services allows the `ethernet-switching` family to be configured on any logical interface unit number instead of only unit 0.

For example, the configuration below shows three logical interfaces configured on a physical interface, xe-0/0/51, that is encapsulated for flexible Ethernet services. Unit 100 and unit 200 are configured in the service provider style and unit 300 is configured in the enterprise style. The encapsulation type of `vlan-bridge` is used to enable bridging on unit 100 and unit 200, and `family ethernet-switching` enables bridging on unit 300.

```
interfaces {
    xe-0/0/51 {
        flexible-vlan-tagging;
        encapsulation flexible-ethernet-services;
        unit 100 {
            encapsulation vlan-bridge;
            vlan-id 100;
        }
        unit 200 {
            encapsulation vlan-bridge;
            vlan-id 200;
        }
        unit 300 {
            family ethernet-switching {
                interface-mode trunk;
                vlan members 300;
            }
        }
    }
}
```

Following are the guidelines to follow when you configure the flexible Ethernet services encapsulation:

- On the QFX10000 line of switches, configuring either `vlan-tagging` or `flexible-vlan-tagging` with `family ethernet-switching` on the same interface is not supported.
- Only on the QFX10000 and EX9200 line of switches, you can enable `vlan-ccc` encapsulation when `flexible-ethernet-services` is already enabled.

- On QFX5100 switches, you can combine encapsulations on the same physical interface for `vlan-bridge` and `family ethernet switching`. Starting with Junos OS Release 16.1R6, you can also combine encapsulations on the same physical interface for `family inet` and `family ethernet-switching`.
- It is not required that the unit number and VLAN ID match, but it is considered a best practice.

Configuring Flexible Ethernet Services Encapsulation to Support the Service Provider and Enterprise Styles of Configuration

Flexible Ethernet services is a type of *encapsulation* that enables a physical interface to specify Ethernet encapsulations at the logical interface level. Each logical interface can have a different Ethernet encapsulation. Defining multiple per-unit Ethernet encapsulations makes it easier to customize Ethernet-based services to multiple hosts connected to the same physical interface.

An Ethernet interface that is not encapsulated with flexible Ethernet services and is operating in Layer 2 mode is limited to a single logical interface unit (0). Bridging is enabled on the interface by configuring `ethernet-switching` as the interface family on unit 0. The `ethernet-switching` family can be configured only on logical interface unit 0, and no other logical units can be defined on that interface.

Some switching features, however, cannot be configured on logical interface unit 0. Features such as Q-in-Q tunneling require the logical interface to transmit VLAN-tagged frames. To enable a logical interface to receive and forward Ethernet frames tagged with a matching VLAN ID, you must bind the logical interface to that VLAN. These features must be configured on a logical interface unit other than 0, because you cannot bind a VLAN ID to unit 0.

When you encapsulate an interface by using flexible Ethernet services, you can configure a logical interface unit other than 0 with `family ethernet-switching`. You can also configure other logical interfaces on that same interface with different types of Ethernet encapsulations. This enables logical interfaces that are bound to a VLAN ID to coexist with logical interfaces configured with `family ethernet-switching`.

For example, if you configure PVLAN on the same physical interface on which you are configuring Q-in-Q tunneling, you can use flexible ethernet services to support the enterprise style of configuration for PVLAN, using `family ethernet-switching`, along with `vlan-bridge` encapsulation for Q-in-Q tunneling.

BEST PRACTICE: We recommend you configure the following statements using groups when configuring devices that function as hardware VTEPs:

- set interfaces *interface-name* flexible-vlan-tagging
- set interfaces *interface-name* encapsulation extended-vlan-bridge

- set interfaces *interface-name* native-vlan-id *vlan-id*

To configure the interface to support both the service provider and enterprise styles of configuration:

1. Enable flexible Ethernet services encapsulation on the interface. The `flexible-ethernet-services` statement allows configuration of both service-provider-style logical interfaces and enterprise-style logical interfaces:

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

2. Enable the interface to transmit packets with 802.1Q VLAN single-tagged and dual-tagged frames:

```
[edit interfaces interface-name]
user@switch# set flexible-vlan-tagging
```

3. Configure a logical interface (unit) on the interface:

```
[edit interfaces interface-name]
user@switch# set unit unit-number
```

NOTE: Do not use logical interface unit 0. You must later bind a VLAN tag ID to the unit you specify in this step, and you cannot bind a VLAN tag ID to unit 0. It is a best practice to match the unit number to the VLAN ID to which the interface is bound.

4. Encapsulate the logical interface for service provider style bridging configuration—for example, use `vlan-bridge` encapsulation on an interface to be used for Q-in-Q tunneling. (If you were configuring the interface only for Q-in-Q tunneling, you would use `encapsulation extended-vlan-bridge` on the *physical* interface.)

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

5. Bind the logical interface from the preceding step to a VLAN ID:

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

6. Configure another logical interface. (If you were configuring just PVLAN, we would recommend that you configure a single logical interface for all PVLAN domains on an interface.)

```
[edit interfaces interface-name]
user@switch# set unit unit-number
```

7. Enable the logical interface in the preceding step for enterprise style bridging configuration:

```
[edit interfaces interface-name]
user@switch# set unit unit-number family ethernet-switching
```

8. Assign VLAN membership to the logical interface:

```
[edit interfaces interface-name]
user@switch# set unit unit-number family ethernet-switching vlan members vlan-id
```

9. Configure the interface as a trunk interface to transmit frames with 802.1Q VLAN tags:

```
[edit interfaces interface-name]
user@switch# set unit unit-number family ethernet-switching interface-mode trunk
```

NOTE: For EX4300 device, the service provider style configuration (encapsulation extended-vlan-bridge) is recommended only for QinQ scenarios. For other scenarios, use the enterprise style configuration.

SEE ALSO

Configuring Q-in-Q Tunneling on QFX Series, NFX Series, and EX4600 Switches with ELS Support

Creating a Private VLAN on a Single Switch with ELS Support (CLI Procedure)

Configure Flexible Ethernet Services Encapsulation to Include Layer 2 Interface Support with Other Encapsulations

SUMMARY

Flexible Ethernet services is a type of encapsulation that enables a physical interface to specify Ethernet encapsulations at the logical interface level. Perform the following steps to configure flexible Ethernet services to support a Layer 2 bridging interface while simultaneously supporting other encapsulation options on the same physical interface.

NOTE: On the QFX10000 line of Switches running Junos OS releases earlier than Release 21.2R1, you cannot configure `vlan-bridging` and any other encapsulations on an interface that has `flexible-ethernet-services` enabled.

Configure a physical or aggregated Ethernet interface to simultaneously support a VLAN based circuit cross-connect (CCC) connection, Layer 3 IP routing, and Layer 2 bridging:

1. Enable flexible Ethernet services encapsulation on the interface.

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

2. Configure the interface to support 802.1Q VLAN single-tagged and dual-tagged frames.

```
[edit interfaces interface-name]
user@switch# set vlan-tagging
```

3. Define a logical interface to support Ethernet VLAN encapsulation for CCC:

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

4. Bind the L2 CCC logical interface from the preceding step to a VLAN ID. This step is needed for all logical interfaces because the physical interface is set for VLAN tagged traffic.

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

5. Configure a second logical interface as an L3 routed IP interface.

```
[edit interfaces interface-name]
user@switch# set unit unit-number family inet address ip-address/mask
```

6. Bind the L3 logical interface from the preceding step to a VLAN ID:

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

7. Configure a third logical interface to support VLAN based bridging by specifying `vlan-bridge` encapsulation on the logical unit.

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

8. Bind the logical interface from the preceding step to a VLAN ID.

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

9. Define a bridge domain and add the L2 logical interface.

```
[edit]
user@switch# set bridge-domains bridge-domain-name vlan-id vlan-id
```

```
[edit]
user@switch# set bridge-domains bridge-domain-name interface interface-id
```

Verify your configuration using the `show interfaces` *interface-name* command in the configuration mode.

```
user@switch> show interfaces xe-0/0/0
vlan-tagging;
encapsulation flexible-ethernet-services;
unit 1 {
    encapsulation vlan-ccc;
    vlan-id 103;
   }
unit 2 {
    vlan-id 102;
    family inet {
        address 10.0.0.1/30;
    }
unit 3 {
    encapsulation vlan-bridge;
    vlan-id 101;
   }

}
```

SEE ALSO

encapsulation (interfaces)

encapsulation (Logical Interface)

vlan-tagging

Configure Flexible Ethernet Services Encapsulation to Support Multiple Logical Interfaces on the Same Physical Interface Mapped to the Same Bridge Domain

SUMMARY

Flexible Ethernet services is a type of encapsulation that enables a physical interface to specify Ethernet encapsulations at the logical interface level. Perform the following steps to configure multiple logical interfaces on the same physical interface mapped to the same bridge domain.

NOTE: The QFX10002-60C switches do not support this feature.

Configure a physical or aggregated Ethernet interface to simultaneously support multiple logical interfaces using the same bridge domain. You cannot configure an ESI interface as one of the logical interfaces over a physical interface when both are part of the same VLAN. ESI interfaces have a limitation.

NOTE: The combination of enterprise-style and service provider-style interfaces on the same physical interface is not supported when there are multiple service provider style logical interfaces attached to the same bridge domain.

1. Configure the interface to support 802.1Q VLAN single-tagged and dual-tagged frames.

```
[edit interfaces interface-name]
user@switch# set vlan-tagging
```

2. Enable flexible Ethernet services encapsulation on the interface.

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

3. Configure a logical interface to support VLAN based bridging by specifying `vlan-bridge` encapsulation on the logical unit.

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

4. Bind the logical interface from the preceding step to a VLAN ID. This step is needed for all logical interfaces because the physical interface is set for VLAN tagged traffic.

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

5. Configure another logical interface to support VLAN based bridging by specifying `vlan-bridge` encapsulation on the logical unit.

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

6. Bind the logical interface from the preceding step to a VLAN ID.

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

7. Configure a bridge domain by specifying the VLAN name and assigning a VLAN ID:

```
[edit]
user@switch# set vlans vlan-name vlan-id vlan-id
```

8. Bind the first logical interface to the bridge domain:

```
[edit]
user@switch# set vlans vlan-name interfaces interface-name.unit
```

9. Bind the second logical interface to the bridge domain.

```
[edit]
user@switch# set vlans vlan-name  interfaces interface-name.unit
```

Verify your configuration using the `show interfaces interface-name` and `show vlans` command in the configuration mode.

```
user@switch> show interfaces xe-0/0/2:2
flexible-vlan-tagging;
encapsulation flexible-ethernet-services;
unit 1 {
    encapsulation vlan-bridge;
    vlan-id 1;
}
unit 2 {
    encapsulation vlan-bridge;
```

```
    vlan-id 2;
}
```

```
user@switch> show vlans
v100 {
    vlan-id 100;
    interface xe-0/0/2:2.1;
    interface xe-0/0/2:2.2;
}
```

SEE ALSO

- *encapsulation (interfaces)*
- *encapsulation (Logical Interface)*
- *vlan-tagging*
- *vlans*

Change History Table

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

Release	Description
16.1R3	Starting with Junos OS Release 16.1R6, you can also combine encapsulations on the same physical interface for `family inet` and `family ethernet-switching`.

3
CHAPTER

Configuring MAC Addresses

IN THIS CHAPTER

MAC Addresses

IN THIS SECTION

Introduction to the Media Access Control (MAC) Layer 2 Sublayer

This topic provides an introduction to the MAC sublayer of the data link layer (Layer 2).

In Layer 2 of a network, the Media Access Control (MAC) sublayer provides addressing and channel access control mechanisms that enable several terminals or network nodes to communicate in a network.

The MAC sublayer acts as an interface between the logical link control (LLC) Ethernet sublayer and Layer 1 (the physical layer). The MAC sublayer emulates a full-duplex logical communication channel in a multipoint network. This channel may provide unicast, multicast, or broadcast communication service. The MAC sublayer uses MAC protocols to prevent collisions.

In Layer 2, multiple devices on the same physical link can uniquely identify one another at the data link layer, by using the MAC addresses that are assigned to all ports on a switch. A MAC algorithm accepts as input a secret key and an arbitrary-length message to be authenticated, and outputs a MAC address.

A MAC address is a 12-digit hexadecimal number (48 bits in long). MAC addresses are usually written in one of these formats:

- MM:MM:MM:SS:SS:SS
- MM-MM-MM-SS-SS-SS

The first half of a MAC address contains the ID number of the adapter manufacturer. These IDs are regulated by an Internet standards body. The second half of a MAC address represents the serial number assigned to the adapter by the manufacturer.

Contrast MAC addressing, which works at Layer 2, with IP addressing, which runs at Layer 3 (networking and routing). One way to remember the difference is that the MAC addresses apply to a physical or virtual node, whereas IP addresses apply to the software implementation of that node. MAC addresses are typically fixed on a per-node basis, whereas IP addresses change when the node moves from one part of the network to another.

IP networks maintain a mapping between the IP and MAC addresses of a node using the Address Resolution Protocol (ARP) table. DHCP also typically uses MAC addresses when assigning IP addresses to nodes.

SEE ALSO

Overview of Layer 2 Networking | 2

Understanding MAC Learning | 75

Understanding MAC Address Assignment on an EX Series Switch

This topic describes MAC address assignment for interfaces on standalone Juniper Networks EX Series Ethernet Switches. For information regarding MAC address assignments in a *Virtual Chassis*, see *Understanding MAC Address Assignment on a Virtual Chassis*.

MAC addresses are used to identify network devices at Layer 2. Because all Layer 2 traffic decisions are based on an interface's MAC address, understanding MAC address assignment is important to understanding how network traffic is forwarded and received by the switch. For additional information on how a network uses MAC addresses to forward and receive traffic, see "Understanding Bridging and VLANs on Switches" on page 140.

A MAC address comprises six groups of two hexadecimal digits, with each group separated from the next group by a colon—for instance, aa:bb:cc:dd:ee:00. The first five groups of hexadecimal digits are derived from the switch and are the same for all interfaces on the switch.

The assignment of a unique MAC address to each network interface helps ensure that functions that require MAC address differentiation—such as redundant trunk groups (RTGs), Link Aggregation Control Protocol (LACP), and general monitoring functions—can properly function.

On switches that use line cards, this MAC addressing scheme differentiates the Layer 2 interfaces on different line cards in the switch.

For EX Series switches, the first five groups of hexadecimal digits are determined when the switch is manufactured. The switch then assigns a unique MAC address to each interface by assigning a unique identifier as the last group of hexadecimal digits. The assignment depends on how the interface is configured. The switch uses a different pattern to distinguish between an interface that is configured as any of a *routed VLAN interface* (*RVI*), a virtual management Ethernet (VME) interface, or an aggregated Ethernet interface or is not configured as any of an RVI, a VME, or as an aggregated Ethernet interface.

For aggregated Ethernet interfaces, the MAC address assignment remains constant regardless of whether the configuration of the interface is Layer 2 or Layer 3.

NOTE: In Junos OS Release 11.3 and later releases through Release 12.1, the MAC address assignment for aggregated Ethernet interfaces changes if the interface is changed from Layer 2 to Layer 3 or the reverse. Starting with Junos Release 12.2, the MAC address assignment for aggregated Ethernet interfaces remains constant regardless of whether the interface is Layer 2 or Layer 3.

NOTE: Prior to Junos OS Release 11.3, MAC addresses for Layer 2 interfaces could be shared between interfaces and RVIs on different line cards in the same switch. However, if you upgrade from Junos OS Release 11.2 or earlier to Junos OS Release 11.3 or later on a switch that supports line cards, the MAC addresses of these interfaces will change.

MAC addresses are assigned to interfaces automatically—no user configuration is possible or required. You can view MAC addresses assigned to interfaces using the `show interfaces` command.

SEE ALSO

Interfaces Overview for Switches

Configuring MAC Move Parameters

When a MAC address appears on a different physical interface or within a different unit of the same physical interface and this behavior occurs frequently, it is considered a MAC move. You can configure the router to report a MAC address move based on the following parameters: the number of times a MAC address move occurs, a specified period of time over which the MAC address move occurs, and specified number of times a MAC address move occurs in one second. You can only configure the `global-mac-move` statement at the global hierarchy level.

To globally disable the MAC move action feature, include the `disable-action` statement at the `[edit protocols l2-learning global-mac-move]`. This disables the MAC move action feature, while MAC move detection exists.

To configure the time duration after which the port will be unblocked, include the `reopen-time` statement at the `[edit protocols l2-learning global-mac-move]`. The default reopen timer is 180 second.

To configure MAC address move reporting if the MAC address moves at least a specified number of times in one second, include the `threshold-time` statement at the `[edit protocols l2-learning global-mac-move]` hierarchy level. The default threshold time is 1 second.

To configure reporting of a MAC address move if the MAC address moves for a specified period of time, include the `notification-time` statement at the `[edit protocols l2-learning global-mac-move]` hierarchy level. The default notification timer is 1 second.

To configure reporting of a MAC address move if the MAC address moves a specified number of times, include the `threshold-count` statement at the `[edit protocols l2-learning global-mac-move]` hierarchy level. The default threshold count is 50 moves.

Use the `show l2-learning mac-move-buffer` command to view the actions as a result of MAC address move feature.

Use the `show l2-learning mac-move-buffer active` command to view the set of IFLs blocked as a result of MAC move action.

Use the `exclusive-mac` command exclude a MAC address from the MAC move limit algorithm, preventing a MAC address from being tracked.

Use the `clear l2-learning mac-move-buffer active` command to unblock the IFBDs that were blocked by MAC move action feature. This allows the user to keep the `reopen-time` configured to a large value, but when the looping error is fixed, user can manually release the blocking.

The following example sets the notification time for MAC moves to 1 second, the threshold time to 1 second, reopen-time to 180 seconds and the threshold count to 50 moves.

```
[edit protocols l2-learning]
global-mac-move {
    notification-time 1;
    reopen-time 180;
    threshold-count 50;
    threshold-time 1;
}
```

Configuring MAC Limiting (ELS)

IN THIS SECTION

This topic describes the different ways of configuring a limitation on MAC addresses in packets that are received and forwarded by the device.

NOTE: The tasks presented in this section uses Junos OS for EX Series switches, QFX3500 and QFX3600 switches, and PTX Series routers that support the Enhanced Layer 2 Software (ELS) configuration style. See Using the Enhanced Layer 2 Software CLI for more information about ELS configurations.

- For information on configuring an interface to automatically recover from a shutdown caused by MAC limiting, see *Configuring Autorecovery for Port Security Events*. If you do not configure the device for autorecovery from the disabled condition, you can bring up the disabled interfaces by running the `clear ethernet-switching recovery-timeout` command.

The different ways of setting a MAC limit are described in the following sections:

Platform-Specific Configuring Global MAC Limit Behavior

Table 22: Platform-Specific Behavior

Platform	Difference
Junos Evolved ACX7000 series	ACX7000 currently does not support packet-action Default MAC numbers are not applicable for ACX7000

Limiting the Number of MAC Addresses Learned by an Interface

NOTE: On PTX Series routers, you can limit the number of MAC addresses learned by an interface only.

To secure a port, you can set the maximum number of MAC addresses that can be learned by an interface.

- Set the MAC limit on an interface, and specify an action that the device takes after the specified limit is exceeded.

  If you want to set the MAC limit on an interface that is part of the default routing instance:

  ```
  [edit switch-options]
  user@switch# set interface interface-name interface-mac-limit limit packet-action action
  ```

  If you want to set the MAC limit on an interface that is part of a routing instance:

  ```
  [edit routing-instances]
  user@switch# set routing-instance-name switch-options interface interface-name interface-mac-
  limit limit
  ```

  If you want to set the MAC limit on all interfaces that are part of the default routing instance:

  ```
  [edit switch-options]
  user@switch# set interface-mac-limit limit
  ```

  If you want to set the MAC limit on all interfaces that are part of a routing instance:

  ```
  [edit routing-instances]
  user@switch# set routing-instance-name switch-options interface-mac-limit limit
  ```

  After you set a new MAC limit for the interface, the system clears existing entries in the MAC address forwarding table associated with the interface.

Limiting the Number of MAC Addresses Learned by a VLAN

IN THIS SECTION

To limit the number of MAC addresses learned by a VLAN, perform the following steps:

Platform-Specific Configuring MAC Limit Behavior

Table 23: Platform-Specific Behavior

Platform	Difference
Junos Evolved ACX7000 series	ACX7000 currently does not support packet-action Default MAC numbers are not applicable for ACX7000

Set the maximum number of MAC addresses that can be learned by a VLAN, and specify an action that the device takes after the specified limit is exceeded:

```
[edit vlans]
user@switch# set vlan-name switch-options mac-table-size limit packet-action action
```

Limiting the Number of MAC Addresses Learned by an Interface in a VLAN

To limit the number of MAC addresses learned by an interface in a VLAN, perform the following steps:

1. Set the maximum number of MAC addresses that can be learned by an interface in a VLAN, and specify an action that the device takes after the specified limit is exceeded:

   ```
   [edit vlans]
   user@switch# set vlan-name switch-options interface-mac-limit limit packet-action action
   ```

2. Set the maximum number of MAC addresses that can be learned by one *or* all interfaces in the VLAN, and specify an action that the device takes after the specified limit is exceeded:

NOTE: If you specify a MAC limit and packet action for all interfaces in the VLAN *and* a specific interface in the VLAN, the MAC limit and packet action specified at the specific interface level takes precedence. Also, at the VLAN interface level, only the `drop` and `drop-and-log` options are supported.

```
[edit vlans]
user@switch# set vlan-name switch-options interface interface-name interface-mac-limit limit
packet-action action
```

```
[edit vlans]
user@switch# set vlan-name switch-options interface-mac-limit limit packet-action action
```

After you set new MAC limits for a VLAN by using the `mac-table-size` statement or for interfaces associated with a VLAN by using the `interface-mac-limit` statement, the system clears the corresponding existing entries in the MAC address forwarding table.

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

Adding a Static MAC Address Entry to the Ethernet Switching Table on a Switch with ELS Support

IN THIS SECTION

NOTE: This task uses Junos OS for EX Series switches and Junos OS for QFX3500 and QFX3600 switches with support for the Enhanced Layer 2 Software (ELS) configuration style. If your switch runs software that does not support ELS, see "Adding a Static MAC Address Entry to the Ethernet Switching Table" on page 71. For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15.

The Ethernet switching table, also known as the forwarding table, specifies the known locations of VLAN nodes and the addresses of devices within those nodes. There are two ways to populate the Ethernet switching table on a switch. The easiest method is to let the switch update the table with MAC addresses.

The second way to populate the Ethernet switching table is to manually insert addresses into the table. You can do this to reduce flooding and speed up the switch's automatic learning process.

Platform-Specific Behavior

Table 24: Platform-Specific Behavior

Platform	Difference
Junos Evolved ACX7000 series	Static MAC entries can be configured only on logical interfaces, not on bridge domains directly. MAC learning can be selectively disabled per VLAN or interface. MAC accounting is supported per VLAN. MAC aging timers and MAC move detection are configurable. Unified Forwarding Table (UFT) allows memory partitioning between MAC, host, and LPM entries for optimized scale. MAC learning behavior is influenced by flooding and split horizon logic, which is implemented late in the forwarding pipeline, so flood traffic is still counted in interface statistics.

Before configuring a static MAC address, be sure that you have:

- Set up the VLAN. See Configuring VLANs for EX Series Switches with ELS Support (CLI Procedure).

To configure an interface to have a static MAC address:

```
[edit vlans vlan-name switch-options interface interface-name]
user@switch# set static-mac mac-address
```

Adding a Static MAC Address Entry to the Ethernet Switching Table

NOTE: This task uses Junos OS for EX Series switches and Junos OS for QFX3500 and QFX3600 switches that does not support the Enhanced Layer 2 Software (ELS) configuration style. If your switch runs software that supports ELS, see "Adding a Static MAC Address Entry to the Ethernet Switching Table on a Switch with ELS Support" on page 69. For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15.

The Ethernet switching table, also known as the forwarding table, specifies the known locations of VLAN nodes. There are two ways to populate the Ethernet switching table on a switch. The easiest method is to let the switch update the table with MAC addresses.

The second way to populate the Ethernet switching table is to manually insert a VLAN node location into the table. You can do this to reduce flooding and speed up the switch's automatic learning process. To further optimize the switching process, indicate the next hop (next interface) packets will use after leaving the node.

Before configuring a static MAC address, be sure that you have:

- Set up the VLAN. See *Configuring VLANs for EX Series Switches* or"Configuring VLANs on Switches" on page 152.

To add a MAC address to the Ethernet switching table:

1. Specify the MAC address to add to the table:

```
[edit ethernet-switching-options]
set static vlan  vlan-name mac mac-address
```

2. Indicate the next hop MAC address for packets sent to the indicated MAC address:

```
[edit ethernet-switching-options]
set static vlan  vlan-name mac mac-address next-hop interface
```

Example: Configuring the Default Learning for Unknown MAC Addresses

IN THIS SECTION

This example shows how to configure the device to use only ARP requests to learn the outgoing interfaces for unknown destination MAC addresses.

Requirements

Before you begin, determine the MAC addresses and associated interfaces of the forwarding table. See "Layer 2 Learning and Forwarding for VLANs Overview" on page 102.

Overview

In this example, you configure the device to use only ARP queries without traceroute requests.

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

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

```
set security flow ethernet-switching no-packet-flooding no-trace-route
```

Step-by-Step Procedure

To configure the device to use only ARP requests to learn unknown destination MAC addresses:

1. Enable the device.

```
[edit]
user@host# set security flow ethernet-switching no-packet-flooding no-trace-route
```

2. If you are done configuring the device, commit the configuration.

```
[edit]
user@host# commit
```

Verification

To verify the configuration is working properly, enter the `show security flow` command.

4

CHAPTER

Configuring MAC Learning

IN THIS CHAPTER

MAC Learning

IN THIS SECTION

Understanding MAC Learning

MAC learning is the process of obtaining the MAC addresses of all the nodes on a network.

When a node is first connected to an Ethernet LAN or VLAN, it has no information about the other nodes on the network. As data is sent through the network, data packets include a data frame listing their source and destination MAC addresses. The data frame is forwarded to a target port, which is connected to the second device. The MAC address is learned locally at the target port, which facilitates communications for frames that later enter the target port and contain addresses previously learned from a received frame.

By default, MAC learning is enabled on the QFX and NFX Series.

Disabling MAC Learning on Devices with ELS Support

By default, MAC learning is globally enabled on all nodes. This topic describes how to disable MAC learning, as well as how to reenable and verify that MAC learning has been enabled or disabled.

NOTE: This task supports the Enhanced Layer 2 Software (ELS) configuration style. For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15 If your switch

runs software that does not support ELS, see "Disabling MAC Learning on QFX Switches" on page 76.

Disabling dynamic MAC learning prevents a node from learning source and destination MAC addresses.

- To disable MAC learning:

```
[edit vlans vlan-name switch-options interface interface-name]
user@switch# set no-mac-learning
```

- To enable MAC learning:

```
[edit vlans vlan-name switch-options interface interface-name]
user@switch# delete no-mac-learning
user@switch# deactivate no-mac-learning
```

- To verify the status of MAC learning, view the Ethernet MAC learning statistics in operational mode.

```
user@switch> show ethernet-switching table
Ethernet-switching table: 2 entries, 1 learned
  VLAN              MAC address       Type         Age Interfaces
  default           *                 Flood          - All-members
  default           00:1f:12:39:90:80 Learn         29 xe-/0/0.0
```

Disabling MAC Learning on QFX Switches

By default, MAC learning is globally enabled on all nodes in a device. This topic describes how to disable MAC learning, as well as how to reenable and verify that MAC learning has been enabled or disabled.

Disabling dynamic MAC learning on the device prevents a node from learning source and destination MAC addresses.

NOTE: This task uses Junos OS for QFX3500 and QFX3600 switches and does not support the Enhanced Layer 2 Software (ELS) configuration style. If your switch runs software that supports ELS, see "Disabling MAC Learning on Devices with ELS Support" on page 75.

- To disable MAC learning on the QFX Series:

```
[edit ethernet-switching-options interfaces interface]
user@switch# set no-mac-learning
```

- To enable MAC learning on the QFX Series:

```
[edit ethernet-switching-options interfaces interface]
user@switch# delete no-mac-learning
user@switch# deactivate no-mac-learning
```

- To verify the status of MAC learning on the QFX Series, view the Ethernet MAC learning statistics in operational mode.

```
user@switch> show ethernet-switching table
Ethernet-switching table: 2 entries, 1 learned
  VLAN              MAC address       Type         Age Interfaces
  default           *                 Flood          - All-members
  default           00:1f:12:39:90:80 Learn         29 xe-/0/0.0
```

Disabling MAC Learning in a VLAN on a QFX Switch

By default, MAC learning is enabled on a VLAN. This topic describes how to disable MAC learning in a VLAN, as well as how to reenable and verify that MAC learning has been enabled or disabled.

Disabling dynamic MAC learning in a VLAN on a QFX Series product prevents a node from learning source and destination MAC addresses.

- To disable MAC learning in a VLAN:

```
[edit vlans vlan-name]
user@switch# set no-mac-learning
```

- To reenable MAC learning in a VLAN, use either of the following two commands:

```
[edit vlans vlan-name]
user@switch# delete no-mac-learning
user@switch# deactivate no-mac-learning
```

- To verify the status of MAC learning on the QFX series:

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

Disabling MAC Learning for a VLAN or Logical Interface

IN THIS SECTION

You can disable MAC learning for all logical interfaces in a specified VLAN, or for a specific logical interface in a VLAN. Disabling dynamic MAC learning prevents the specified interfaces from learning source MAC addresses.

To disable MAC learning for all logical interfaces in a VLAN in a virtual switch, include the `no-mac-learning` statement at the `[edit vlans` *`vlan-name`* `switch-options]` hierarchy level:

```
[edit]
vlans {
    vlan-name {
        domain-type bridge;
        interface interface-name;
        switch-options {
            no-mac-learning;
        }
    }
}
```

To disable MAC learning for a specific logical interface in a VLAN, include the `no-mac-learning` statement at the `[edit vlans vlan-name switch-options interface interface-name]` hierarchy level.

```
[edit]
vlans {
    vlan-name {
        domain-type bridge;
        interface interface-name;
        switch-options {
            interface interface-name {
                no-mac-learning;
            }
        }
    }
}
```

NOTE: When you disable MAC learning, source MAC addresses are not dynamically learned, and any packets sent to these source addresses are flooded into the VLAN.

NOTE: When you gather interfaces into a VLAN, the `no-mac-learn-enable` statement at the `[edit interfaces interface-name ether-options ethernet-switch-profile]` hierarchy level is not supported. You must use the `no-mac-learning` statement at the `[edit vlans vlan-name switch-options interface interface-name]` hierarchy level to disable MAC learning on an interface in a VLAN.

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

Platform-Specific Enable/Disable dynamic MAC Learning Behavior

Table 25: Platform-Specific Behavior

Platform	Difference
Junos Evolved ACX7000 series	MAC learning is enabled by default per VLAN and per logical interface. MAC learning can be selectively disabled globally, per VLAN, or per interface. Static MAC entries are supported only on logical interfaces. MAC aging timers and MAC move detection are configurable. Flood traffic is still counted in interface statistics due to late split horizon implementation. Unified Forwarding Table (UFT) allows memory partitioning for MAC, host, and LPM entries.
Junos Evolved ACX7000 series	Disabling dynamic MAC learning globally prevents all entities (like VLANs, Routing Instances, Logical interfaces etc) from learning source MAC addresses. • ACX Hierarchy for disabling global learning,**set protocols l2-learning global-no-mac-learning** • ACX Hierarchy for disabling global learning,**set protocols l2-learning global-no-mac-learning**

SEE ALSO

global-no-mac-learning

Disabling MAC Learning for a Set of VLANs

You can disable MAC learning for a set of VLANs. Disabling dynamic MAC learning prevents the Layer 2 trunk port associated with the set of VLANs from learning source and destination MAC addresses.

When you disable MAC learning, source MAC addresses are not dynamically learned, and any packets sent to these source addresses are flooded into the switch.

To disable MAC learning for a set of VLANs, include the `no-mac-learning` statement at the `[edit switch-options]` hierarchy level:

```
[edit switch-options]
no-mac-learning;
```

5
CHAPTER

Configuring MAC Accounting

IN THIS CHAPTER

MAC Accounting

IN THIS SECTION

Enabling MAC Accounting on a Device

By default, MAC accounting is disabled on the device. You can enable packet accounting either for a device as a whole or for a specific VLAN. After you enable packet accounting, the Junos OS maintains packet counters for each MAC address learned.

To enable MAC accounting, include the `global-mac-statistics` statement at the `[edit protocols l2-learning]` hierarchy level:

```
[edit protocols l2-learning]
global-mac-statistics;
```

Enabling MAC Accounting for a VLAN

By default, MAC accounting is disabled. You can enable packet counting for a VLAN. When you enable packet accounting, the Junos OS maintains packet counters for each MAC address learned on the interfaces in the VLAN.

To enable MAC accounting for a VLAN, include the `mac-statistics` statement at the `[edit vlans` *`vlan-name`* `switch-options]` hierarchy level:

```
[edit vlans vlan-name switch-options]
mac-statistics;
```

Enabling MAC Accounting for a Set of VLANs

By default, MAC accounting is disabled. You can enable packet counting for a set of VLANs. After you enable packet accounting, the Junos OS maintains packet counters for each MAC address learned on the trunk port associated with the set of VLANs.

To enable MAC accounting for a set of VLANs, include the `mac-statistics` statement at the `[edit switch-options]` hierarchy level:

```
[edit switch-options]
mac-statistics;
```

Verifying That MAC Accounting Is Working

IN THIS SECTION

Purpose

Verify that MAC accounting is enabled and the system is counting packets and collecting statistics.

Action

1. Verify that MAC accounting is enabled.

```
user@switch> show ethernet-switching table
MAC flags (S - static MAC, D - dynamic MAC, L - locally learned
        SE - statistics enabled, NM - non configured MAC, R - remote PE MAC)

Routing instance : default-switch
    Vlan                MAC                 MAC         Age    Logical
```

```
    name                address             flags              interface
    VLAN101             88:e0:f3:bb:07:f0   D,SE           -   ae20.0


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

Routing instance : default-switch
    Vlan                MAC                 MAC         Age    Logical
    name                address             flags              interface
    VLAN102             88:e0:f3:bb:07:f0   D,SE           -   ae20.0


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

Routing instance : default-switch
    Vlan                MAC                 MAC         Age    Logical
    name                address             flags              interface
    VLAN103             88:e0:f3:bb:07:f0   D,SE           -   ae20.0
[...output truncated...]
```

2. Display MAC accounting statistics for all VLANs associated with an interface.

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

 Local interface: ae20.0, Index: 1039
   Broadcast packets:                  115
   Broadcast bytes  :                 6900
   Multicast packets:               395113
   Multicast bytes  :             61622869
   Flooded packets  :                    0
   Flooded bytes    :                    0
   Unicast packets  :                 1419
   Unicast bytes    :               117924
   Current MAC count:                    4 (Limit 8192)
[...output truncated...]
```

3. Display MAC accounting statistics for each address in the MAC address table.

```
user@switch> show ethernet-switching table extensive
MAC address: 88:e0:f3:bb:07:f0
  Routing instance: default-switch
VLAN ID: 101
   Learning interface: ae20.0
   Layer 2 flags: in_hash,in_ifd,in_ifl,in_vlan,in_rtt,acct,kernel,in_ifbd
   Epoch: 6                          Sequence number: 13
   Learning mask: 0x00000020
MAC address used as destination:
Packet count:                     0  Byte count:                     0
MAC address used as source:
Packet count:                     9  Byte count:                  1116

MAC address: 88:e0:f3:bb:07:f0
  Routing instance: default-switch
VLAN ID: 102
   Learning interface: ae20.0
   Layer 2 flags: in_hash,in_ifd,in_ifl,in_vlan,in_rtt,acct,kernel,in_ifbd
   Epoch: 6                          Sequence number: 13
   Learning mask: 0x00000020
MAC address used as destination:
Packet count:                     0  Byte count:                     0
MAC address used as source:
Packet count:                     9  Byte count:                  1116

MAC address: 88:e0:f3:bb:07:f0
  Routing instance: default-switch
VLAN ID: 103
   Learning interface: ae20/0
   Layer 2 flags: in_hash,in_ifd,in_ifl,in_vlan,in_rtt,acct,kernel,in_ifbd
   Epoch: 6                          Sequence number: 13
   Learning mask: 0x00000020
MAC address used as destination:
Packet count:                     0  Byte count:                     0
MAC address used as source:
Packet count:                     9  Byte count:                  1116
[...output truncated...]
```

Meaning

In the output for `show ethernet-switching table`, the MAC flag `SE` indicates that MAC accounting is enabled for VLANs 101, 102, and 103, which are all associated with the `default-switch` routing instance.

The output for `show ethernet-switching statistics` displays packet statistics and the current number of MAC addresses learned by the VLANs associated with aggregated Ethernet interface `ae20.0`.

The output for `show ethernet-switching table extensive` shows information for each address in the MAC address table. In particular, it displays the number of packets sent to and received by an interface, which is identified by a MAC address.

The output from the three commands demonstrates that MAC accounting is working properly. That is, MAC accounting is enabled on VLANs 101, 102, and 103, and as a result, you can view statistics for each of these VLANs, aggregated Ethernet interface `ae20.0`, and each MAC address.

CHAPTER 6

Configuring MAC Notification

IN THIS CHAPTER

MAC Notification

IN THIS SECTION

Understanding MAC Notification on EX Series Switches

Juniper Networks EX Series Switches track clients on a network by storing Media Access Control (MAC) addresses in the Ethernet switching table on the switch. When switches learn or unlearn a MAC address, SNMP notifications can be sent to the network management system at regular intervals to record the addition or removal of the MAC address. This process is known as MAC notification.

The MAC Notification MIB controls MAC notification for the network management system. For general information on the MAC Notification MIB, see the *Junos OS Network Management Configuration Guide*.

The MAC notification interval defines how often these SNMP notifications are sent to the network management system. The MAC notification interval works by tracking all of the MAC address additions or removals on the switch over a period of time and then sending all of the tracked MAC address additions or removals to the network management server at the end of the interval. For instance, if the MAC notification interval is set to 10, all of the MAC address addition and removal SNMP notifications are sent to the network management system every 10 seconds.

Enabling MAC notification allows users to monitor the addition and removal of MAC addresses from the Ethernet switching table remotely using a network management system. The advantage of setting a high MAC notification interval is that the amount of network traffic is reduced because updates are sent less frequently. The advantage of setting a low MAC notification interval is that the network management system is better synchronized with the switch.

MAC notification is disabled by default. When MAC notification is enabled, the default MAC notification interval is 30 seconds.

Configuring MAC Notification on Switches with ELS Support

IN THIS SECTION

NOTE: This task uses the Enhanced Layer 2 Software (ELS) configuration style. If your switch runs software that does not support ELS, see "Configuring Non-ELS MAC Notification " on page 92 or "Configuring Non-ELS MAC Notification " on page 92. For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15.

When a switch learns or unlearns a MAC address, SNMP notifications can be sent to the network management system at regular intervals to record the addition or removal of the MAC address. This process is known as MAC notification.

The MAC notification interval defines how often Simple Network Management Protocol (SNMP) notifications logging the addition or removal of MAC addresses on the switch are sent to the network management system.

MAC notification is disabled by default. When MAC notification is enabled, the default MAC notification interval is 30 seconds.

To enable or disable MAC notification, or to set the MAC notification interval, perform these tasks:

Enabling MAC Notification

MAC notification is disabled by default. You need to perform this procedure to enable MAC notification.

To enable MAC notification on the switch with the default MAC notification interval of 30 seconds:

```
[edit switch-options]
user@switch# set mac-notification
```

To enable MAC notification on the switch with any other MAC notification interval (here, the MAC notification interval is set to 60 seconds):

```
[edit switch-options]
user@switch# set mac-notification notification-interval 60
```

Disabling MAC Notification

MAC notification is disabled by default. Perform this procedure only if MAC notification was previously enabled on your switch.

To disable MAC notification on the switch:

```
[edit switch-options]
user@switch# delete mac-notification
```

To disable MAC notification on a specific interface (here, the interface is ge-0/0/3):

```
[edit switch-options]
user@switch# set interface ge-0/0/3 no-mac-notification
```

Setting the MAC Notification Interval

The default MAC notification interval is 30 seconds. The procedure to change the MAC notification interval to a different interval is identical to the procedure to enable MAC notification on the switch with a nondefault value for the MAC notification interval.

To set the MAC notification interval on the switch (here, the MAC notification interval is set to 5 seconds):

```
[edit switch-options]
user@switch# set mac-notification notification-interval 5
```

Configuring Non-ELS MAC Notification

IN THIS SECTION

NOTE: This task uses Junos OS for EX Series switches that do not support Enhanced Layer 2 Software (ELS) configuration style. If your switch runs software that supports ELS, see "Configuring MAC Notification on Switches with ELS Support" on page 90. For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15.

When a switch learns or unlearns a MAC address, SNMP notifications can be sent to the network management system at regular intervals to record the addition or removal of the MAC address. This process is known as MAC notification.

The MAC notification interval defines how often Simple Network Management Protocol (SNMP) notifications logging the addition or removal of MAC addresses on the switch are sent to the network management system.

MAC notification is disabled by default. When MAC notification is enabled, the default MAC notification interval is 30 seconds.

To enable or disable MAC notification, or to set the MAC notification interval, perform these tasks:

Enabling MAC Notification

MAC notification is disabled by default. You need to perform this procedure to enable MAC notification.

To enable MAC notification on the switch with the default MAC notification interval of 30 seconds:

```
[edit ethernet-switching-options]
user@switch# set mac-notification
```

To enable MAC notification on the switch with any other MAC notification interval (here, the MAC notification interval is set to 60 seconds):

```
[edit ethernet-switching-options]
user@switch# set mac-notification notification-interval 60
```

Disabling MAC Notification

MAC Notification is disabled by default. Perform this procedure only if MAC notification was previously enabled on your switch.

To disable MAC notification on the switch:

```
[edit ethernet-switching-options]
user@switch# delete mac-notification
```

Setting the MAC Notification Interval

The default MAC notification interval is 30 seconds. The procedure to change the MAC notification interval to a different interval is identical to the procedure to enable MAC notification on the switch with a nondefault value for the MAC notification interval.

To set the MAC notification interval on the switch (here, the MAC notification interval is set to 5 seconds):

```
[edit ethernet-switching-options]
user@switch# set mac-notification notification-interval 5
```

Verifying That MAC Notification Is Working Properly

IN THIS SECTION

Purpose

Verify that MAC notification is enabled or disabled, and that the MAC notification interval is set to the specified value.

Action

To verify that MAC notification is enabled or disabled on a QFX Series switch or an EX4600, and also to verify the MAC notification interval setting:

```
user@switch> show ethernet-switching mac-notification
Notification Status: Enabled
Notification Interval: 60
Notifications Sent          : 0
Notifications Table Maxsize : 256
```

The output in the **Notification Status** field shows that MAC notification is enabled. The output in the **Notification Status** field would display **Disabled** if MAC notification was disabled.

The **Notification Interval** field output shows that the MAC notification interval is set to 60 seconds.

To verify that MAC notification is enabled on an EX Series switch while also verifying the MAC notification interval setting:

```
user@switch> show ethernet-switching mac-notification
Notification Status: Enabled
Notification Interval: 30
```

The output in the **Notification Status** field shows that MAC notification is enabled. The output in the **Notification Status** field would display **Disabled** if MAC notification was disabled.

The **Notification Interval** field output shows that the MAC notification interval is set to 30 seconds.

CHAPTER 7

Configuring MAC Table Aging

IN THIS CHAPTER

MAC Table Aging

IN THIS SECTION

Understanding MAC Table Aging

IN THIS SECTION

Juniper Networks EX Series Ethernet Switches store MAC addresses in the Ethernet switching table, also called the *MAC table*. When the aging time for a MAC address in the table expires, the address is removed.

If your switch runs Juniper Networks Junos operating system (Junos OS) for EX Series switches with support for the Enhanced Layer 2 Software (ELS) configuration style, you can configure the MAC table aging time on all VLANs on the switch. If your switch runs Junos OS that does not support ELS, you can configure the MAC table aging time on all VLANs on the switch or on specified VLANs, as well as configure aging time to be unlimited, either on all VLANs or on specified VLANs, so that MAC addresses never age out of the table.

To learn MAC addresses, the switch reads all packets that it detects on the LAN or on the local VLAN, looking for MAC addresses of sending nodes. It places these addresses into its Ethernet switching table, along with two other pieces of information—the interface on which the traffic was received and the time when the address was learned.

When the switch receives traffic on an interface, it searches the Ethernet switching table for the MAC address of the destination. If the MAC address is not found, the traffic is flooded out all of the other interfaces associated with the VLAN. For example, if traffic is received on an interface that is associated with VLAN v-10 and there is no entry in the Ethernet switching table for VLAN v-10 (the Ethernet

switching table is organized by VLAN), then the traffic is flooded to all access and trunk interfaces that are members of VLAN v-10.

Flooding allows the switch to learn about destinations that are not yet in its Ethernet switching table. If a particular destination MAC address is not in the Ethernet switching table, the switch floods the traffic to all interfaces except the interface on which it was received. When the destination node receives the flooded traffic, it sends an acknowledgment packet back to the switch, allowing the switch to learn the MAC address of the node and to add the address to its Ethernet switching table.

The switch uses a mechanism called aging to keep the Ethernet switching table current. For each MAC address in the Ethernet switching table, the switch records a timestamp of when the information about the network node was learned. Each time the switch detects traffic from a MAC address that is in its Ethernet switching table, it updates the timestamp of that MAC address. A timer on the switch periodically checks the timestamp, and if the MAC address of a node is older than the value set, the switch removes that MAC address from the Ethernet switching table. This aging process ensures that the switch tracks only active MAC addresses on the network and that it is able to flush out from the Ethernet switching table MAC addresses that are no longer available.

You configure how long MAC addresses remain in the Ethernet switching table by:

- (On switches that run Junos OS with support for the ELS configuration style) Using the `global-mac-table-aging-time` statement in the `[edit protocols l2-learning]` hierarchy.
- (On switches that run Junos OS that does not support ELS) Using the `mac-table-aging-time` statement in either the **[edit ethernet-switching-options]** or the **[edit vlans]** hierarchy, depending on whether you want to configure it for the entire switch or only for specific VLANs.

For example, in a topology with EX switches that run Junos OS that does not support ELS, if you have a printer VLAN, you might choose to configure the aging time for that VLAN to be considerably longer than for other VLANs so that MAC addresses of printers on this VLAN age out less frequently. Because the MAC addresses remain in the table, even if a printer has been idle for some time before traffic arrives for it, the switch still finds the MAC address and does not need to flood the traffic to all other interfaces.

Similarly, in a data center environment where the list of servers connected to the switch is fairly stable, you might choose to increase MAC address aging time, or even set it to unlimited, to increase the efficiency of the utilization of network bandwidth by reducing flooding.

Platform-Specific Configuring MAC Aging Behavior

Table 26: Platform-Specific Behavior

Platform	Difference
Junos Evolved ACX7000 series	MAC aging configuration can only be done globally for ACX7000 series.

SEE ALSO

Access Control and Authentication on Switching Devices

Configuring MAC Table Aging on Switches

MAC table aging ensures that a switch tracks only active nodes on the network and that it is able to flush out network nodes that are no longer available.

To manage MAC entries more efficiently, you can configure an entry's aging time, which is the maximum time that an entry can remain in the MAC address table before it is deleted because it has reached its maximum age.

The following example uses Junos OS for Junos OS for QFX3500 and QFX3600 switches with no support for the Enhanced Layer 2 Software (ELS) configuration style. Use the `set-mac-table-aging-time` command to configure how long entries remain in the Ethernet switching table before expiring. Here the VLAN is **employee-vlan**:

```
[edit vlans employee-vlan]
user@switch# set mac-table-aging-time 200
```

The following example uses Junos OS for QFX Series switches with support for the Enhanced Layer 2 Software (ELS) configuration style. Use the `global-mac-table-aging-time` command to configure how long entries remain in the Ethernet switching table before expiring, as follows:

```
[edit protocols l2-learning]
user@switch# set global-mac-table-aging-time 200
```

NOTE: This command applies to all VLANs configured for the switch. You cannot configure separate MAC table aging times for specific VLANs.

The following example uses Junos OS for EX Series switches with support for the Enhanced Layer 2 Software (ELS) configuration style.

The Ethernet switching table (or MAC table) aging process ensures that the EX Series switch tracks only active MAC addresses on the network and is able to flush out MAC addresses that are no longer used.

You can configure the MAC table aging time, the maximum time that an entry can remain in the Ethernet Switching table before it *ages out*, on all VLANs on the switch. This setting can influence efficiency of network resource use by affecting the amount of traffic that is flooded to all interfaces because when traffic is received for MAC addresses no longer in the Ethernet switching table, the switch floods the traffic to all interfaces.

```
[edit]
user@switch# set protocols l2-learning global-mac-table-aging-time seconds
```

The following example uses Junos OS for EX Series switches that do not support the Enhanced Layer 2 Software (ELS) configuration style.

The Ethernet switching table (or MAC table) aging process ensures that the EX Series switch tracks only active MAC addresses on the network and is able to flush out MAC addresses that are no longer used.

You can configure the MAC table aging time, the maximum time that an entry can remain in the Ethernet Switching table before it “ages out,” either on all VLANs on the switch or on particular VLANs. This setting can influence efficiency of network resource use by affecting the amount of traffic that is flooded to all interfaces because when traffic is received for MAC addresses no longer in the Ethernet switching table, the switch floods the traffic to all interfaces.

To configure the MAC table aging time on all VLANs on the switch:

```
[edit]
user@switch# set ethernet-switching-options mac-table-aging-time seconds
```

To configure the MAC table aging time on a VLAN:

```
[edit]
user@switch# set vlans vlan-name mac-table-aging-time seconds
```

NOTE: You can set the MAC table aging time to unlimited. If you specify the value as **unlimited**, entries are never removed from the table. Generally, use this setting only if the switch or the VLAN has a fairly static number of end devices; otherwise the table will eventually fill up. You can use this setting to minimize traffic loss and flooding that might occur when traffic arrives for MAC addresses that have been removed from the table.

8
CHAPTER

Configuring Learning and Forwarding

IN THIS CHAPTER

Layer 2 Forwarding Tables

IN THIS SECTION

Layer 2 Learning and Forwarding for VLANs Overview

IN THIS SECTION

Understanding Layer 2 Forwarding Tables on Switches, Routers and NFX Series Devices

You can configure Layer 2 MAC address and VLAN learning and forwarding properties in support of Layer 2 bridging. Unicast media access control (MAC) addresses are learned to avoid flooding the packets to all the ports in a VLAN. A source MAC entry is created in its source and destination MAC tables for each MAC address learned from packets received on ports that belong to the VLAN.

When you configure a VLAN, Layer 2 address learning is enabled by default. The VLAN learns unicast media access control (MAC) addresses to avoid flooding the packets to all the ports in the VLAN. Each

VLAN creates a source MAC entry in its source and destination MAC tables for each source MAC address learned from packets received on the ports that belong to the VLAN.

NOTE: Traffic is not flooded back onto the interface on which it was received. However, because this "split horizon" occurs at a late stage, the packet statistics displayed by commands such as `show interfaces queue` will include flood traffic.

You can optionally disable MAC learning either for the entire device or for a specific VLAN or logical interface. You can also configure the following Layer 2 learning and forwarding properties:

- Timeout interval for MAC entries
- Static MAC entries for logical interfaces only
- Limit to the number of MAC addresses learned from a specific logical interface or from all the logical interfaces in a VLAN
- Size of the MAC address table for the VLAN
- MAC accounting for a VLAN

Understanding Layer 2 Forwarding Tables on Security Devices

The SRX Series Firewall maintains forwarding tables that contain MAC addresses and associated interfaces for each Layer 2 VLAN. When a packet arrives with a new source MAC address in its frame header, the device adds the MAC address to its forwarding table and tracks the interface at which the packet arrived. The table also contains the corresponding interface through which the device can forward traffic for a particular MAC address.

If the destination MAC address of a packet is unknown to the device (that is, the destination MAC address in the packet does not have an entry in the forwarding table), the device duplicates the packet and floods it on all interfaces in the VLAN other than the interface on which the packet arrived. This is known as *packet flooding* and is the default behavior for the device to determine the outgoing interface for an unknown destination MAC address. Packet flooding is performed at two levels: packets are flooded to different zones as permitted by configured Layer 2 security policies, and packets are also flooded to different interfaces with the same VLAN identifier within the same zone. The device learns the forwarding interface for the MAC address when a reply with that MAC address arrives at one of its interfaces.

You can specify that the SRX Series Firewall use ARP queries and traceroute requests (which are ICMP echo requests with the time-to-live values set to 1) instead of packet flooding to locate an unknown destination MAC address. This method is considered more secure than packet flooding because the device floods ARP queries and traceroute packets—not the initial packet—on all interfaces. When ARP or traceroute flooding is used, the original packet is dropped. The device broadcasts an ARP or ICMP query

to all other devices on the same subnetwork, requesting the device at the specified destination IP address to send back a reply. Only the device with the specified IP address replies, which provides the requestor with the MAC address of the responder.

ARP allows the device to discover the destination MAC address for a unicast packet if the destination IP address is in the same subnetwork as the ingress IP address. (The ingress IP address refers to the IP address of the last device to send the packet to the device. The device might be the source that sent the packet or a router forwarding the packet.) Traceroute allows the device to discover the destination MAC address even if the destination IP address belongs to a device in a subnetwork beyond that of the ingress IP address.

When you enable ARP queries to locate an unknown destination MAC address, traceroute requests are also enabled. You can also optionally specify that traceroute requests not be used; however, the device can then discover destination MAC addresses for unicast packets only if the destination IP address is in the same subnetwork as the ingress IP address.

Whether you enable ARP queries and traceroute requests or ARP-only queries to locate unknown destination MAC addresses, the SRX Series Firewall performs the following series of actions:

1. The device notes the destination MAC address in the initial packet. The device adds the source MAC address and its corresponding interface to its forwarding table, if they are not already there.
2. The device drops the initial packet.
3. The device generates an ARP query packet and optionally a traceroute packet and floods those packets out all interfaces except the interface on which the initial packet arrived.

   ARP packets are sent out with the following field values:

   - Source IP address set to the IP address of the IRB
   - Destination IP address set to the destination IP address of the original packet
   - Source MAC address set to the MAC address of the IRB
   - Destination MAC address set to the broadcast MAC address (all `0xf`)

   Traceroute (ICMP echo request or ping) packets are sent out with the following field values:

   - Source IP address set to the IP address of the original packet
   - Destination IP address set to the destination IP address of the original packet
   - Source MAC address set to the source MAC address of the original packet
   - Destination MAC address set to the destination MAC address of the original packet
   - Time-to-live (TTL) set to `1`

4. Combining the destination MAC address from the initial packet with the interface leading to that MAC address, the device adds a new entry to its forwarding table.
5. The device forwards all subsequent packets it receives for the destination MAC address out the correct interface to the destination.

Platform-Specific Layer 2 Forwarding Tables Behavior

Table 27: Platform-Specific Behavior

Platform	Difference
Junos Evolved ACX7000 series	MAC learning is enabled by default per VLAN; each VLAN maintains its own MAC forwarding table. Split horizon behavior is implemented late in the forwarding pipeline, so flood traffic is still counted in interface statistics. MAC learning can be disabled globally, per VLAN, or per logical interface. MAC table size and MAC learning limits can be configured per VLAN or per logical interface. Static MAC entries are supported only on logical interfaces. MAC accounting is available per VLAN. Unified Forwarding Table (UFT) profiles allow partitioning of shared memory (MDB) for MAC, L3 host, and LPM entries, optimizing for L2-heavy or L3-heavy deployments.

Layer 2 Learning and Forwarding for VLANs Acting as a Switch for a Layer 2 Trunk Port

Layer 2 learning is enabled by default. A set of VLANs, configured to function as a switch with a Layer 2 trunk port, learns unicast media access control (MAC) addresses to avoid flooding packets to the trunk port.

NOTE: Traffic is not flooded back onto the interface on which it was received. However, because this "split horizon" occurs at a late stage, the packet statistics displayed by commands such as `show interfaces queue` will include flood traffic.

You can optionally disable Layer 2 learning for the entire set of VLANs as well as modify the following Layer 2 learning and forwarding properties:

- Limit the number of MAC addresses learned from the Layer 2 trunk port associated with the set of VLANs
- Modify the size of the MAC address table for the set of VLANs
- Enable MAC accounting for the set of VLANs

Understanding the Unified Forwarding Table

IN THIS SECTION

Benefits of Unified Forwarding Tables

Traditionally, forwarding tables have been statically defined and have supported only a fixed number of entries for each type of address. The unified forwarding table (UFT) provides the following benefits:

- Enables you to allocate forwarding table resources to optimize the memory available for different address types based on the needs of your network.
- Enables you to allocate a higher percentage of memory for one type of address or another.

Using the Unified Forwarding Table to Optimize Address Storage

On EX4400, EX4600, EX4650, QFX5100, QFX5110, QFX5120, and QFX5200 switches, you can control the allocation of forwarding table memory available to store the following:

- MAC addresses—In a Layer 2 environment, the switch learns new MAC addresses and stores them in a MAC address table.
- Layer 3 host entries—In a Layer 2 and Layer 3 environment, the switch learns which IP addresses are mapped to which MAC addresses; these key-value pairs are stored in the Layer 3 host table.
- Longest prefix match (LPM) table entries—In a Layer 3 environment, the switch has a routing table and the most specific route has an entry in the forwarding table to associate a prefix or netmask to a next hop. Note, however, that all IPv4 /32 prefixes and IPv6 /128 prefixes are stored in the Layer 3 host table.

UFT essentially combines the three distinct forwarding tables to create one table with flexible resource allocation. You can select one of five forwarding table profiles that best meets your network needs. Each profile is configured with different maximum values for each type of address. For example, for a switch that handles a great deal of Layer 2 traffic, such as a virtualized network with many servers and virtualized machines, you would likely choose a profile that allocates a higher percentage of memory to MAC addresses. For a switch that operates in the core of a network, participates in an IP fabric, you probably want to maximize the number of routing table entries it can store. In this case, you would choose a profile that allocates a higher percentage of memory to longest match prefixes. The QFX5200 switch supports a custom profile that allows you to partition the four available shared memory banks with a total of 128,000 entries among MAC addresses, Layer 3 host addresses, and LPM prefixes.

NOTE: We introduced support for QFX5200 switches in Junos OS Release 15.1x53-D30. The QFX5200 switch is not supported on Junos OS Release 16.1R1.

Understanding the Allocation of MAC Addresses and Host Addresses

All five profiles are supported, each of which allocates different amounts of memory for Layer 2 or Layer 3 entries, enabling you choose one that best suits the needs of your network. The QFX5200 and QFX5210 switches, however, supports different maximum values for each profile from the other switches. For more information about the custom profile, see "Configuring the Unified Forwarding Table on Switches" on page 124.

NOTE: The default profile is `l2-profile-three`, which allocates equal space for MAC Addresses and Layer 3 host addresses. On EX4400, EX4600, QFX5100, QFX5110, and QFX5200 switches, the space is equal to 16,000 IPv4 entries for the LPM table, and on QFX5210 switches, the space is equal to 32,000 IPv4 entries for the LPM table. For the `lpm-profile` the LPM table size is equal to 256,000 IPv4 entries.

NOTE: Starting with Junos OS Release 18.1R1 on the QFX5210-64C switch, for all these profiles except for the `lpm-profile`, the longest prefix match (LPM) table size is equal to 32,000 IPv4 entries.

NOTE: Starting with Junos OS Release 18.3R1, on the EX4650 and QFX5120 switches, for all these profiles except for the `lpm-profile`, the longest prefix match (LPM) table size is equal to 144,000 IPv4 entries.

NOTE: On EX4400, EX4600, EX4650, QFX5100, QFX5110, QFX5120, QFX5200, and QFX5210-64C switches, IPv4 and IPv6 host routes with ECMP next hops are stored in the host table.

BEST PRACTICE: If the host or LPM table stores the maximum number of entries for any given type of entry, the entire shared table is full and is unable to accommodate *any* entries of any other type. Different entry types occupy different amounts of memory. For example, an IPv6 unicast address occupies twice as much memory as an IPv4 unicast address, and an IPv6 multicast address occupies four times as much memory as an IPv4 unicast address.

Table 28 on page 109 lists the profiles you can choose and the associated maximum values for the MAC address and host table entries on EX4400 switches.

Table 28: Unified Forwarding Table Profiles on EX4400 Switches

Profile Name	MAC Table	Host Table (unicast and multicast addresses)					
	MAC Addresses	IPv4 unicast	IPv6 unicast	IPv4 (*, G)	IPv4 (S, G)	IPv6 (*, G)	IPv6 (S, G)
`l2-profile-one`	112K	16K	8K	8K	8K	4K	4K
`l2-profile-two`	96K	32K	16K	16K	16K	8K	8K
`l2-profile-three` (default)	80K	48K	24K	24K	24K	12K	12K
`l3-profile`	48K	80K	40K	40K	40K	20K	20K
`lpm-profile`	16K	16K	8K	8K	8K	4K	4K

Table 29 on page 109 lists the profiles you can choose and the associated maximum values for the MAC address and host table entries on EX4600 and QFX5100 switches.

Table 29: Unified Forwarding Table Profiles on EX4600 and QFX5100 Switches

Profile Name	MAC Table	Host Table (unicast and multicast addresses)					
	MAC Addresses	IPv4 unicast	IPv6 unicast	IPv4 (*, G)	IPv4 (S, G)	IPv6 (*, G)	IPv6 (S, G)
`l2-profile-one`	288K	16K	8K	8K	8K	4K	4K
`l2-profile-two`	224K	80K	40K	40K	40K	20K	20K

Table 29: Unified Forwarding Table Profiles on EX4600 and QFX5100 Switches *(Continued)*

Profile Name	MAC Table	Host Table (unicast and multicast addresses)					
`l2-profile-three` (default)	160K	144K	72K	72K	72K	36K	36K
`l3-profile`	96K	208K	104K	104K	104K	52K	52K
`lpm-profile`	32K	16K	8K	8K	8K	4K	4K
`lpm-profile`with `unicast-in-lpm` option	32K	(stored in LPM table)	(stored in LPM table)	8K	8K	4K	4K

Table 30 on page 110 lists the profiles you can choose and the associated maximum values for the MAC address and host table entries on QFX5110 switches.

Table 30: Unified Forwarding Table Profiles on QFX5110 Switches

Profile Name	MAC Table	Host Table (unicast and multicast addresses)					
	MAC Addresses	IPv4 unicast	IPv6 unicast	IPv4 (*, G)	IPv4 (S, G)	IPv6 (*, G)	IPv6 (S, G)
`l2-profile-one`	288K	16K	8K	8K	8K	4K	4K
`l2-profile-two`	224K	80K	40K	40K	40K	20K	20K
`l2-profile-three` (default)	160K	144K	72K	72K	72K	36K	36K
`l3-profile`	96K	208K	104K	104K	104K	52K	52K

Table 31 on page 111 lists the LPM table size variations for the QFX5110 switch depending on the prefix entries.

Table 31: LPM Table Size Variations on QFX5110 Switches

Profile Name	Prefix Entries		
num-65-127-prefix	IPv4 LPM<= /32	IPv6 LPM <= /64	IPv6 LPM > /64
0	16K	8K	0K
1	12K	6K	1K
2	8K	4K	2K
3	4K	2K	3K
4	0K	0K	4K

Table 32 on page 111 lists the profiles you can choose and the associated maximum values for the MAC address and host table entries on QFX5200-32C switches.

Table 32: Unified Forwarding Table Profiles on QFX5200-32C Switches

Profile Name	MAC Table	Host Table (unicast and multicast addresses)						
	MAC Addresses	IPv4 unicast	IPv6 unicast	IPv4 (*, G)	IPv4 (S, G)	IPv6 (*, G)	IPv6 (S, G)	Exact-Match
l2-profile-one	136K	8K	4K	4K	4K	2K	2K	0
l2-profile-two	104K	40K	20K	20K	20K	10K	10K	0
l2-profile-three (default)	72K	72K	36K	36K	36K	18K	18K	0
l3-profile	40K	104K	52K	52K	52K	26K	26K	0

Table 32: Unified Forwarding Table Profiles on QFX5200-32C Switches *(Continued)*

Profile Name	MAC Table	Host Table (unicast and multicast addresses)						
lpm-profile	8K	8K	4K	4K	4K	2K	2K	0

Table 33 on page 112 lists the profiles you can choose and the associated maximum values for the MAC address and host table entries on QFX5200-48Y switches.

Table 33: Unified Forwarding Table Profiles on QFX5200-48Y Switches

Profile Name	MAC Table	Host Table (unicast and multicast addresses)					
	MAC Addresses	IPv4 unicast	IPv6 unicast	IPv4 (*, G)	IPv4 (S, G)	IPv6 (*, G)	IPv6 (S, G)
l2-profile-one	136K	8K	4K	4K	4K	2K	2K
l2-profile-two	104K	40K	20K	20K	20K	10K	10K
l2-profile-three (default)	72K	72K	36K	36K	36K	18K	18K
l3-profile	40K	104K	52K	52K	52K	26K	26K
lpm-profile	8K	8K	4K	4K	4K	2K	2K

Table 34 on page 112 lists the LPM table size variations for the QFX5200-48Y switch depending on the prefix entries.

Table 34: LPM Table Size Variations on QFX5200-48Y Switches

Profile Name	Prefix Entries		
num-65-127-prefix	IPv4 LPM<= /32	IPv6 LPM <= /64	IPv6 LPM > /64
0	16K	8K	0K

Table 34: LPM Table Size Variations on QFX5200-48Y Switches *(Continued)*

Profile Name	Prefix Entries		
1	12K	6K	1K
2	8K	4K	2K
3	40K	2K	3K
4	0K	0K	4K

Table 35 on page 113 lists the profiles you can choose and the associated maximum values for the MAC address and host table entries on QFX5210-64C switches.

Table 35: Unified Forwarding Table Profiles on QFX5210-64C Switches

Profile Name	MAC Table	Host Table (unicast and multicast addresses)						
	MAC Addresses	IPv4 unicast	IPv6 unicast	IPv4 (*, G)	IPv4 (S, G)	IPv6 (*, G)	IPv6 (S, G)	Exact Match
l2-profile-one	264K	8K	4K	4K	4K	2K	2K	0K
l2-profile-two	200K	72K	36K	36K	36K	18K	18K	0K
l2-profile-three (default)	136K	136K	72K	72K	72K	36K	36K	0K
l3-profile	72K	200K	100K	100K	100K	50K	50K	0K

Table 36 on page 114 lists the profiles you can choose and the associated maximum values for the MAC address and host table entries on EX4650 and QFX5120 switches.

Table 36: Unified Forwarding Table Profiles on EX4650 and QFX5120 Switches

Profile Name	MAC Table	Host Table (unicast and multicast addresses)					
	MAC Addresses	IPv4 unicast	IPv6 unicast	IPv4 (*, G)	IPv4 (S, G)	IPv6 (*, G)	IPv6 (S, G)
`l2-profile-one`	288K	16K	8K	8K	8K	4K	4K
`l2-profile-two`	224K	80K	40K	40K	40K	20K	20K
`l2-profile-three` (default)	160K	144K	72K	72K	72K	36K	36K
`l3-profile`	96K	208K	104K	104K	104K	52K	52K
`lpm-profile`	32K	16K	8K	8K	8K	4K	4K

Table 37 on page 114 lists the LPM table size variations for the QFX5210-64C switch depending on the prefix entries.

Table 37: LPM Table Size Variations on QFX5210-64C Switches

Profile Name	Prefix Entries		
num-65-127-prefix	IPv4 LPM < /32	IPv6 LPM <= /64	IPv6 LPM > /64
`0`	32K	16K	0K
`1`	28K	14K	1K
`2`	24K	12K	2K
`3`	20K	10K	3K

Table 37: LPM Table Size Variations on QFX5210-64C Switches *(Continued)*

Profile Name	Prefix Entries		
4	0K	0K	4K

Table 38 on page 115 lists the Layer 3 Defip table size variations for the EX4650 and QFX5120 switches depending on the changing IPv6/128 prefix entries.

Table 38: LPM Table Size Variations on EX4650 and QFX5120 Switches

Profile Name	Prefix Entries		
num-65-127-prefix	**IPv4 LPM<= /32**	**IPv6 LPM <= /64**	**IPv6 LPM > /64**
0	32K	16K	0K
2	24K	12K	2K
4	16K	8K	4K
6	8K	4K	6K
8	0K	0K	8K

Unified Forwarding Table Profiles on QFX5130 and QFX5700 Switches for Junos OS Evolved Releases

You can configure a forwarding-profile for the unified forwarding table on QFX5130 and QFX5700 switches using the `forwarding-profile` configuration statement at the [edit system packet-forwarding-options] hierarchy level for Junos OS Evolved.

NOTE: Only the `host-profile` unified `forwarding profile` is recommended with EVPN-VXLAN configurations.

For VXLAN, each Layer 2-MAC uses 2X entries width in the L2 table. Hence the VXLAN `L2 mac` scale will be half of the `L2-mac` scale. For higher `L2 mac` scale the `host-profile` is recommended.

```
user@switch# set system packet-forwarding-options forwarding-profile ?
Possible completions:
+ apply-groups         Groups from which to inherit configuration data
+ apply-groups-except  Don't inherit configuration data from these groups
  default-profile      Refer 'show pfe uft-profile-info' for profile info; restarts PFE
  host-acl-profile     Refer 'show pfe uft-profile-info' for profile info; restarts PFE
  host-profile         Refer 'show pfe uft-profile-info' for profile info; restarts PFE
  lpm-profile          Refer 'show pfe uft-profile-info' for profile info; restarts PFE
```

You can view the scale per profile using the `show pfe uft-profile-info` command.

```
user@switch> show pfe uft-profile-info
SENT: Ukern command: show evo-pfemand uft profile-info
================================================================================
                   PFE UFT Profiles
================================================================================
                default-profile  lpm-profile  host-profile  host-acl-profile
================================================================================
IPV4-host       32K              32K          160K          160K
IPV4-lpm        720K             1.24M        72K           65K
IPV6-host       16K              16K          80K           80K
IPV6-lpm        550K             868K         50K           22K
L2-mac          32K              32K          160K          160K
FP-compression  18K              0            0             18K
ARP-overlay     32K              64K          32K           32K
ARP-underlay    32K              0            32K           32K
L3-mcast v4     16K              16K          32K           32K
L3-mcast v6     8K               8K           16K           16K
Tunnels         Supported        No support   Supported     Supported
================================================================================
```

Table 39: Unified Forwarding Table Profiles on QFX5130 and QFX5700 Switches

Profile Applications Feature	Default profile	LPM-profile	Host-profile	Host-ACL-profile
Layer 2-MAC	32K	32K	160K	160K
Layer 3 Host Unicast -IPv4	32K	32K	160K	160K
Layer 3 Host Unicast -IPv6	16K	16K	80K	80K
IPv4 LPM	720K	1.24M	72K	65K
IPv6 LPM <= /64	550K	868K	50K	22K
IPv6 LPM > /64	335K	495K	22K	12K
FP-compression	18K	0	0	18K
ARP & NDP	32K	61K	32K	32K
VRF	upto 8K	upto 12K	upto 8K	upto 4K
Layer 3 Multicast IPv4	8K	8K	16K	16K
Layer 3 Multicast IPv6	4K	4K	8K	8K
Tunnels (VXLAN and GRE)	Supported	Not Supported	Supported	Supported

NOTE:

1. When the host capacity is exceeded, the host unicast routes (IPv4 and IPv6) roll-over to the LPM table.
2. LPM profile does not support tunnels (vxlan, gre etc) due to which an overlay next-hop scale increases to 64K resulting in increasing ARP/NDP scale to 61K.

Understanding Ternary Content Addressable Memory (TCAM) and Longest Prefix Match Entries

You can further customize non-LPM profiles by configuring the space available for ternary content addressable memory (TCAM) to allocate more memory for longest prefix match entries. You can change the number of entries allocated to these IPv6 addresses, essentially allocating more or less space for LPM IPv4 entries with any prefix length or IPv6 entries with prefix lengths of 64 of shorter. For more information about how to change the default parameters of the TCAM memory space for LPM entries, see "Configuring the Unified Forwarding Table on Switches" on page 124.

NOTE: The option to adjust TCAM space is not supported on the longest prefix match (LPM) or custom profiles. However, for the LPM profile, you can configure TCAM space not to allocate any memory for IPv6 entries with prefix lengths of 65 or longer, thereby allocating that memory space only for IPv4 routes or IP routes with prefix lengths equal to or less than 64 or a combination of the two types of prefixes.

NOTE: Starting with Junos OS Release 18.1R1 on QFX5210 switches, you can configure TCAM space to allocate a maximum of 8,000 IPv6 entries with prefix lengths of 65 or longer. The default value is 2,000 entries. Starting with Junos OS Release 13.2X51-D15, you can configure TCAM space to allocate a maximum of 4,000 IPv6 entries with prefix lengths of 65 or longer. The default value is 1,000 entries. Previous to Junos OS Release 13.2X51-D15, you could allocate only a maximum of 2,048 entries for IPv6 the IPv6 prefixes with lengths in the range /65 to /127 range. The default value was 16 entries for these types of IPv6 prefixes.

On Junos OS Releases 13.2x51-D10 and 13.2x52D10, the procedure to change the default value of 16 entries differs from later releases, where the maximum and default values are higher. For more information about that procedure, see "Configuring the Unified Forwarding Table on Switches" on page 124

Host Table Example for Profile with Heavy Layer 2 Traffic

Table 40 on page 119 lists various valid combinations that the host table can store if you use the `l2-profile-one` profile on EX4600 and QFX5100 switches. This profile allocates the percentage of memory to

Layer 2 addresses. Note that the default values might be different on other switches. Each row in the table represents a case in which the host table is full and cannot accommodate any more entries.

Table 40: Example Host Table Combinations Using l2-profile-one on EX4600 and QFX5100 Switches

IPv4 unicast	IPv6 unicast	IPv4 multicast (*, G)	IPv4 multicast (S, G)	IPv6 multicast (*, G)	IPv6 multicast (S, G)
16K	0	0	0	0	0
12K	2K	0	0	0	0
12K	0	2K	2K	0	0
8K	4K	0	0	0	0
4K	2K	2K	2K	0	0
0	4K	0	0	1K	1K

Example: Configuring a Unified Forwarding Table Custom Profile

IN THIS SECTION

Traditionally, forwarding tables have been statically defined and have supported only a fixed number of entries for each type of address. The Unified Forwarding Table (UFT) feature enables you to optimize how forwarding-table memory is allocated to best suit the needs of your network. This example shows how to configure a Unified Forwarding Table profile that enables you to partition four shared hash

memory banks among three different types of forwarding-table entries: MAC addresses, Layer 3 host addresses, and longest prefix match (LPM).

The UFT feature also supports five profiles that each allocate a specific maximum amount of memory for each type of forwarding table entry. Some profiles allocate more memory to Layer 2 entries, while other profiles allocate more memory to Layer 3 or LPM entries. The maximum values for each type of entry are fixed in these profiles. With the custom profile, you can designate one or more shared memory banks to store a specific type of forwarding-table entry. You can configure as few as one or as many as four memory banks in a custom profile. The custom profile thus provides even more flexibility in enabling you to allocate forwarding-table memory for specific types of entries.

Requirements

This example uses the following hardware and software components:

- One QFX5200 switch
- Junos OS Release 15.1x53-D30 or later.

Before you configure a custom profile, be sure you have:

- Configured interfaces

Overview

The Unified Forwarding Table custom profile enables you to allocate forwarding-table entries among four banks of shared hash tables with a total memory equal to 128,000 unicast IPv4 addresses, or 32,000 entries for each bank. Specifically, you can allocate one or more of these shared banks to store a specific type of forwarding-table entry. The custom profile does not affect the dedicated hash tables. Those tables remain fixed with 8,000 entries allocated to Layer 2 addresses, the equivalent of 8,000 entries allocated to IPv4 addresses, and the equivalent of 16,000 entries allocated to longest prefix match (LPM) addresses.

In this example, you allocate two memory banks to Layer 3 host addresses, and two memory banks to LPM entries. This means that no shared hash table memory is allocated for Layer 2 addresses. Only the dedicated hash table memory is allocated for Layer 2 addresses in this scenario.

Configuration

IN THIS SECTION

To configure a custom profile for the Unified Forwarding Table feature on a QFX5200 switch that allocates two shared memory banks for Layer 3 host address and two shared memory banks for LPM entries, perform these tasks:

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the `[edit]` hierarchy level, and then enter `commit` from configuration mode. A commit check is performed to ensure that you have allocated forwarding-table space for no more than four memory banks.

CAUTION: When you configure and commit a profile, the Packet Forwarding Engine restarts and all the data interfaces on the switch go down and come back up.

```
user@switch# set chassis forwarding-options custom-profile
user@switch# set chassis forwarding-options custom-profile l2-entries num-banks 0
user@switch# set chassis forwarding-options custom-profile l3-entries num-banks 2
user@switch# set chassis forwarding-options custom-profile lpm-entries num-banks 2
```

Configuring the Custom Profile

Step-by-Step Procedure

To create the custom profile:

1. Specify the `custom-profile` option.

```
[edit chassis forwarding-options]
user@switch# set custom-profile
```

Configuring the Allocation of Shared Memory Banks

Step-by-Step Procedure

To allocate memory for specific types of entries for the shared memory banks:

1. Specify to allocate no shared bank memory for Layer 2 entries.

```
[edit chassis forwarding-options custom-profile]
user@switch# set l2-entries num-banks 0
```

2. Specify to allocate two shared memory banks (or the equivalent of 64,000 IPv4 entries) for Layer 3 host entries.

```
[edit chassis forwarding-options custom-profile]
user@switch# set l3-entries num-banks 2
```

3. Specify to allocate two shared memory banks (or the equivalent of 64,000 IPv4 entries) for LPM entries.

```
[edit chassis forwarding-options custom-profile]
user@switch# set lpm-entries numer-banks 2
```

Results

From configuration mode, confirm your configuration by entering the show chassis forwarding-options command. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@switch# show chassis forwarding-profile
custom-profile {
    l2-entries {
        num-banks 0;
    }
    l3-entries {
        num-banks 2;
    }
    lpm-entries {
        num-banks 2
```

```
    }
}
```

If you are done configuring the switch, enter `commit` from configuration mode

CAUTION: The Packet Forwarding Engine will restart and all the data interfaces on the switch will go down and come back up.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Checking the Parameters of the Custom Profile

Purpose

Verify that the custom profile is enabled.

Action

```
user@switch> show chassis forwarding-options
UFT Configuration:
custom-profile
Configured custom scale:
Entry type                Total scale(K)
L2(mac)                   8
L3 (unicast & multicast)        72
Exact Match                0
Longest Prefix Match (lpm)        80
num-65-127-prefix = 1K
-------------Bank details for various types of entries------------
Entry type                  Dedicated Bank Size(K)      Shared Bank Size(K)
L2 (mac)                    8                         32 * num shared banks
```

```
L3 (unicast & multicast          8                          32 * num shared banks
Exact match                   0                          16 * num shared banks
Longest Prefix match(lpm)          16                       32 * num shared banks
```

Meaning

The output shows that the custom profile is enabled as configured with two shared memory banks designated for Layer 3 host entries; two shared memory banks designated for LPM entries; and no shared memory allocated for Layer 2 entries.

The total scale(K) field shows the total allocation of memory, that is, the amount allocated through the shared memory banks plus the amount allocated through the dedicated hash tables. The amount allocated through the dedicated hash tables is fixed and cannot be changed. Therefore, Layer 2 entries have 8K of memory allocated only through the dedicated hash table. Layer 3 host entries have 64K of memory allocated through two shared memory banks plus 8K through the dedicated hash table, for a total of 72K of memory. LPM entries have 64K of memory allocated through two shared memory banks plus 16K through the dedicated hash table, for a total of 80K of memory.

Configuring the Unified Forwarding Table on Switches

IN THIS SECTION

Traditionally, forwarding tables have been statically defined and have supported only a fixed number of entries for each type of address stored in the tables. The Unified Forwarding Table feature lets you optimize how your switch allocates forwarding-table memory for different types of addresses. You can choose one of five unified forwarding table profiles. Each profile allocates a different maximum amount of memory for Layer 2, Layer 3 host, and longest prefix match (LPM) entries. In addition to selecting a profile, you can also select how much additional memory to allocate for LPM entries.

Two profiles allocate higher percentages of memory to Layer 2 addresses. A third profile allocates a higher percentage of memory to Layer 3 host address, while a fourth profile allocates a higher percentage of memory to LPM entries. There is a default profile configured that allocates an equal amount of memory to Layer 2 and Layer 3 host addresses with the remainder allocated to LPM entries. For a switch in a virtualized network that handles a great deal of Layer 2 traffic, you would choose a

profile that allocates a higher percentage of memory to Layer 2 addresses. For a switch that operates in the core of the network, you would choose a profile that allocates a higher percentage of memory to LPM entries.

On QFX5200 and QFX5210-64C switches only, you can also configure a custom profile that allows you to partition shared memory banks among the different types of forwarding table entries. On QFX5200 switches, these shared memory banks have a total memory equal to 128,000 IPv4 unicast addresses. On QFX5210 switches, these shared memory banks have a total memory equal to 256,000 IPv4 unicast addresses. For more information about configuring the custom profile, see "Example: Configuring a Unified Forwarding Table Custom Profile" on page 119.

Configuring a Unified Forwarding Table Profile

To configure a unified forwarding table profile:

Specify a forwarding-table profile.

```
[edit chassis forwarding-options]
user@switch# set profile-name
```

For example, to specify the profile that allocates the highest percentage of memory to Layer 2 traffic:

```
[edit chassis forwarding-options]
user@switch# set l2-profile-one
```

CAUTION: When you configure and commit a profile, in most cases the Packet Forwarding Engine automatically restarts and all the data interfaces on the switch go down and come back up (the management interfaces are unaffected).

Starting with Junos OS Releases 14.1X53-D40, 15.1R5, and 16.1R3, for a Virtual Chassis or Virtual Chassis Fabric (VCF) comprised of EX4600 or QFX5100 switches, the Packet Forwarding Engine in member switches does not automatically restart upon configuring and committing a unified forwarding table profile change. This behavior avoids Virtual Chassis or VCF instability after the change propagates to member switches and multiple Packet Forwarding Engines automatically restart at the same time. Instead, a message is displayed at the CLI prompt and logged to the switch's system log to notify you that the profile change does not take effect until the next time you reboot the Virtual Chassis or VCF. We recommend that you plan to make profile changes only when you can perform a Virtual Chassis or VCF system reboot immediately after committing the configuration update. Otherwise, the Virtual Chassis or VCF could become inconsistent if one or more

members have a problem and restart with the new configuration before a planned system reboot activates the change on all members.

NOTE: You can configure only one profile for the entire switch.

NOTE: The `l2-profile-three` is configured by default.

NOTE: If the host table stores the maximum number of entries for any given type, the entire table is full and is unable to accommodate *any* entries of any other type. Keep in mind that an IPv6 unicast address occupies twice as much memory as an IPv4 unicast address, and an IPv6 multicast address occupies four times as much memory as an IPv4 unicast address.

Configuring the Memory Allocation for Longest Prefix Match Entries

IN THIS SECTION

In addition to choosing a profile, you can further optimize memory allocation for longest prefix match (LPM) entries by configuring how many IPv6 prefixes to store with lengths from /65 through /127. The switch uses LPM entries during address lookup to match addresses to the most-specific (longest) applicable prefix. Prefixes of this type are stored in the space for ternary content addressable memory (TCAM). Changing the default parameters makes this space available for LPM entries. Increasing the amount of memory available for these IPv6 prefixes reduces by the same amount how much memory is available to store IPv4 unicast prefixes and IPv6 prefixes with lengths equal to or less than 64.

The procedures for configuring the LPM table are different, depending on which version of Junos OS you are using. In the initial releases that UFT is supported, Junos OS Releases 13.2X51-D10 and 13.2X52-10, you can only increase the amount of memory allocated to IPv6 prefixes with lengths from /65 through /127 for any profile, except for `lpm-profile`. Starting with Junos OS Release 13.2X51-D15, you can also allocate either less or no memory for IPv6 prefixes with lengths in the range /65 through /127, depending on which profile is configured. For the `lpm-profie`, however, the only change you can make to the default parameters is to allocate no memory for these types of prefixes.

Configuring the LPM Table With Junos OS Releases 13.2X51-D10 and 13.2X52-D10

In Junos OS Releases 13.2x51-D10 and 13.2X52-D10, by default, the switch allocates memory for 16 IPv6 with prefixes with lengths in the range /65 through /127. You can configure the switch to allocate more memory for IPv6 prefixes with lengths in the range /65 through /127.

To allocate more memory for IPv6 prefixes in the range /65 through /127:

1. Choose a forwarding table profile.

```
[edit chassis forwarding-options]
user@swtitch# set profile-name
```

For example, to specify the profile that allocates the highest percentage of memory to Layer 2 traffic:

```
[edit chassis forwarding-options]
user@swtitch# set l2-profile-one
```

2. Select how much memory to allocate for IPv6 prefixes in the range /65 through 127.

```
[edit chassis forwarding-options profile-name]
user@swtitch# set num-65-127-prefix number
```

For example, to specify to allocate memory for 32 IPv6 prefixes in the range /65 through 127:

```
[edit chassis forwarding-options l2-profile-one]
user@switch# set num-65-127-prefix 2
```

NOTE: When you configure and commit the `num-65-127-prefix number` statement, all the data interfaces on the switch restart. The management interfaces are unaffected.

The `num-65-127-prefix number` statement is not supported on the `lpm-profile`.

Configuring the LPM Table With Junos OS Release 13.2x51-D15 and Later

IN THIS SECTION

Configuring Layer 2 and Layer 3 Profiles With Junos OS Release 13.2x51-D15 or Later

Starting in Junos OS Release 13.2X51-D15, you can configure the switch to allocate forwarding table memory for as many as 4,000 IPv6 prefixes with lengths in the range /65 through /127 for any profile other than the `lpm-profile` or `custom-profile`. You can also specify to allocate no memory for these IPv6 entries. The default is 1,000 entries for IPv6 prefixes with lengths in the range /65 through /127. Previously, the maximum you could configure was for 2,048 entries for IPv6 prefixes with lengths in the range /65 through /127. The minimum number of entries was previously 16, which was the default.

To specify how much forwarding table memory to allocate for IPv6 prefixes with length in the range /65 through /127:

1. Choose a forwarding table profile.

```
[edit chassis forwarding-options]
user@swtitch# set profile-name
```

   For example, to specify the profile that allocates the highest percentage of memory to Layer 2 traffic:

```
[edit chassis forwarding-options]
user@swtitch# set l2-profile-one
```

2. Select how much memory to allocate for IPv6 prefixes in the range /65 through 127.

```
[edit chassis forwarding-options profile-name]
user@swtitch# set num-65-127-prefix number
```

For example, to specify to allocate memory for 2,000 IPv6 prefixes in the range /65 through 127:

```
[edit chassis forwarding-options l2-profile-one]
user@switch# set num-65-127-prefix 2
```

Starting with Junos OS Release 13.2X51-D15, you can use the `num-65-127-prefix` statement to allocate entries. Table 15 shows the numbers of entries that you can allocate. Each row represents a case in which the table is full and cannot accommodate any more entries.

Table 41: LPM Table Combinations for L2 and L3 profiles With Junos OS 13.2X51-D15 and Later

num-65-127-prefix Value	IPv4 Entries	IPv6 Entries (Prefix <= 64)	IPv6 Entries (Prefix >= 65)
0	16K	8K	0K
1 (default)	12K	6K	1K
2	8K	4K	2K
3	4K	2K	3K
4	0K	0K	4K

CAUTION: When you configure and commit a profile change with the `num-65-127-prefix` *number* statement, the Packet Forwarding Engine automatically restarts and all the data interfaces on the switch go down and come back up (the management interfaces are unaffected).

However, starting with Junos OS Releases 14.1X53-D40, 15.1R5, and 16.1R3, Packet Forwarding Engines on switches in a Virtual Chassis or Virtual Chassis Fabric (VCF) do not automatically restart upon configuring a unified forwarding table profile change. This behavior avoids Virtual Chassis or VCF instability after the change propagates to member switches and multiple Packet Forwarding Engines automatically restart at the same time. Instead, a message is displayed at the CLI prompt and logged to the switch's system log to notify you that the profile change does not take effect until the next time you reboot the Virtual Chassis or VCF. We recommend that you plan to make profile changes only when you can perform a Virtual Chassis or VCF system reboot immediately

after committing the configuration update. Otherwise, the Virtual Chassis or VCF could become inconsistent if one or more members have a problem and restart with the new configuration before a planned system reboot activates the change on all members.

Configuring the lpm-profile With Junos OS Release 13.2x51-D15 and Later

Starting with Junos OS Release 13.2X51-D15 you can configure the `lpm-profile` profile not to allocate any memory for IPv6 entries with prefix lengths from /65 through /127. These are the default maximum values allocated for LPM memory for the `lpm-profile` by address type:

- 128K of IPv4 prefixes
- 16K of IPv6 prefixes (all lengths)

NOTE: The memory allocated for each address type represents the maximum default value for all LPM memory.

To configure the `lpm-profile` not to allocate forwarding-table memory for IPv6 entries with prefixes from /65 through /127, thus allocating more memory for IPv4:

Specify to disable forwarding-table memory for IPv6 prefixes with lengths in the range /65 through /127.

```
[edit chassis forwarding-options lpm-profile]
user@switch# set prefix-65-127-disable
```

For example, on the QFX5100 and EX4600 switches only, if you use the `prefix-65-127-disable` option, each of the following combinations are valid:

- 100K IPv4 and 28K IPv6 /64 or shorter prefixes.
- 64K IPv4 and 64K IPv6 /64 or shorter prefixes.
- 128K IPv4 and 0K IPv6 /64 or shorter prefixes.
- 0K IPv4 and 128K IPv6 /64 or shorter prefixes.

NOTE: On the QFX5200 switches, when you configure the `prefix-65-127-disable` statement, the maximum number of IPv6 entries with prefixes equal to or shorter than 64 is 98,000.

Configuring the lpm-profile With Junos OS Release 14.1x53-D30 and Later

Starting in Junos OS Release 15.1X53-D30, you can configure the `lpm-profile` profile to store unicast IPv4 and IPv6 host addresses in the LPM table , thereby freeing memory in the host table. Unicast IPv4 and IPv6 addresses are stored in the LPM table instead of the host table, as shown in Table 16 for QFX5100 and EX4600 switches. (Platform support depends on the Junos OS release in your installation.) You can use this option in conjunction with the option to allocate no memory in the LPM table for IPv6 entries with prefix lengths in the range /65 through /127. Together, these options maximize the amount of memory available for IPv4 unicast entries and IPv6 entries with prefix lengths equal to or less than 64.

Table 42: lpm-profile with unicast-in-lpm Option for QFX5100 and EX4600 Switches

prefix-65-127-disable	MAC Table	Host Table (multicast addresses)						LPM Table unicast addresses)		
	MAC	IPv4 unicast	IPv6 unicast	IPv4 (*, G)	IPv4 (S, G)	IPv6 (*, G)	IPv6 (S, G)	IPv4 unicast	IPv6 unicast (</65)	IPv6 unicast (>/64)
No	32K	0	0	8K	8K	4K	4K	128K	16K	16K
Yes	32K	0	0	8K	8K	4K	4K	128K	128K	0

Starting with Junos Release 18.1R1, you cannot set configure a prefix for the `num-65-127-prefix` statement on non-LPM profiles. You can only enable or disable the `prefix-65-127-disable` statement for the `lpm-profile`.

Table 17 lists the situations in which the `prefix-65-127-disable` statement should be enabled or disabled.

Table 43: LPM Table Size Variations on QFX5200-48Y Switches

Profile Name	Prefix Entries		
num-65-127-prefix	IPv4 <= /32	IPv6 <= /64	IPv6 > /64
Enabled	> 128K (minimum guaranteed)	98K	0K
Disabled	128K	16K	16K

On QFX5120 and EX4600 switches, you cannot set configure a prefix for the num-65-127-prefix statement on non-LPM profiles. You can only enable or disable the prefix-65-127-disable statement for the lpm-profile

Table 18 lists the situations in which the prefix-65-127-disable statement should be enabled or disabled.

Table 44: LPM Table Size Variations on QFX5120 and EX4650 Switches

Profile Name	Prefix Entries		
prefix-65-127-disable	**IPv4 <= /32**	**IPv6 <= /64**	**IPv6 > /64**
Enabled	351K (360,000 approximate)	168K (172,000 approximate)	0K
Disabled	168K (172,000 approximate)	64K (65,524 approximate)	64K (65,524 approximate)

Note that all entries in each table share the same memory space. If a table stores the maximum number of entries for any given type, the entire shared table is full and is unable to accommodate any entries of any other type. For example, if you use the the unicast-in-lpm option and there are 128K IPv4 unicast addresses stored in the LPM table, the entire LPM table is full and no IPv6 addresses can be stored. Similarly, if you use the unicast-in-lpm option but do not use the prefix-65-127-disable option, and 16K IPv6 addresses with prefixes shorter than /65 are stored, the entire LPM table is full and no additional addresses (IPv4 or IPv6) can be stored.

To configure the lpm-profile to store unicast IPv4 entries and IPv6 entries with prefix lengths equal to or less than 64 in the LPM table:

1. Specify the option to store these entries in the LPM table.

```
[edit chassis forwarding-options lpm-profile]
user@switch# set unicast-in-lpm
```

2. (Optional) Specify to allocate no memory for in the LPM table for IPv6 prefixes with length in the range /65 through /127:

```
[edit chassis forwarding-options lpm-profile]
user@switch# set prefix-65-127-disable
```

Configuring Non-LPM Profiles on QFX5120 and EX4650 Switches

For non-LPM profiles, each profile provides the option of reserving a portion of the 16K L3-defip table to store IPv6 Prefixes > 64. Because these are 128-bit prefixes, you can have maximum of 8k IPv6/128 entries in the l3-defip table.

1. Choose a forwarding table profile.

```
[edit chassis forwarding-options]
user@swtitch# set profile-name
```

For example, to specify the profile that allocates the highest percentage of memory to Layer 3 traffic:

```
[edit chassis forwarding-options]
user@swtitch# set l3-profile
```

2. Select how much memory to allocate for IPv6 prefixes in the range /65 through 127.

```
[edit chassis forwarding-options profile-name]
user@swtitch# set num-65-127-prefix number
```

For example, to specify to allocate memory for 2,000 IPv6 prefixes in the range /65 through 127:

You can choose between 0 and 4, 1 being the default.

```
[edit chassis forwarding-options l3-profile]
user@switch# set num-65-127-prefix 1
```

Understand and Configure the Unified Forwarding Table

IN THIS SECTION

Use the Unified Forwarding Table to Optimize Address Storage

ACX5048 and ACX5096 routers support the use of a unified forwarding table to optimize address storage. This feature gives you the flexibility to configure your router to match the needs of your particular network environment. You can control the allocation of forwarding table memory available to store the following entries:

- MAC addresses
- Layer 3 host entries
- Longest prefix match (LPM) table entries

You can use five predefined profiles (**l2-profile-one, l2-profile-two, l2-profile-three, l3-profile, lpm-profile**) to allocate the table memory space differently for each of these entries. The sizes of the Layer 2 MAC address table, Layer 3 host entry table, and Layer 3 LPM table are decided based on the selected profile. You can configure and select the profiles that best suits your network environment needs.

Table 19 illustrates the predefined profiles in the unified forwarding table and the respective table sizes.

Table 45: Unified Forwarding Table Profiles

Profile	Layer 2 MAC Address Table	Layer 3 Host Table	Layer 3 LPM Table
l2-profile-one	288 K	16 K	16 K
l2-profile-two	224 K	80 K	16 K
l2-profile-three (default)	160 K	144 K	16 K
l3-profile	96 K	208 K	16 K
lpm-profile	32 K	16 K	128 K

IPv4 unicast, IPv6 unicast, IPv4 multicast, and IPv6 multicast route addresses share the Layer 3 host entry table. If the host table stores the maximum number of entries for any given type, the entire table is full and is unable to accommodate any entries of any other type. The IPv4 multicast and IPv6 unicast addresses occupy double the space as that occupied by IPv4 unicast entries, and IPv6 multicast addresses occupy four times the space of the IPv4 unicast addresses. Table 20 shows the Layer 3 host table size for each profile.

Table 46: Layer 3 Host Table

Profile	Layer 3 Host Table			
	IPv4 Unicast	IPv4 Multicast	IPv6 Unicast	IPv6 Multicast
l2-profile-one	16 K	8 K	8 K	4 K
l2-profile-two	80 K	40 K	40 K	20 K
l2-profile-three (default)	144 K	72 K	72 K	36 K
l3-profile	208 K	104 K	104 K	52 K
lpm-profile	16 K	8 K	8 K	4 K

The Layer 3 LPM table is shared between IPv4 route prefixes and IPv6 route prefixes. Table 21 illustrates the size of the table for different profiles of the IPv4 and IPv4 addresses in the Layer 3 LPM table. When unicast reverse-path forwarding (unicast RPF) is enabled, the table size reduces to half.

Table 47: Layer 3 LPM Table

Profile	Layer 3 LPM Table		
	IPv4 Unicast	IPv6 Unicast (Prefix <= /64)	IPv6 Unicast (Prefix > /64)
l2-profile-one	16 K	8 K	4 K
l2-profile-two	16 K	8 K	4 K
l2-profile-three (default)	16 K	8 K	4 K
l3-profile	16 K	8 K	4 K
lpm-profile	128 K	40 K	8 K

By default, there is no space allocated for IPv6 prefix address longer than /64 in the LPM table. Therefore, prefix address longer than /64 are not allowed in the table by default. The entire table is available for IPv4 addresses and for IPv6 addresses that have prefixes shorter than /64. You can provide space in the table for addresses with prefixes longer than /64 by using CLI configuration. The number of entries reserved for these prefixes is configured in multiples of 16.

Configure the Unified Forwarding Table to Optimize Address Storage Using Profiles

You can use five predefined profiles (**l2-profile-one, l2-profile-two, l2-profile-three, l3-profile, lpm-profile**) to allocate the table memory space. The sizes of the Layer 2 MAC address table, Layer 3 host entry table, and Layer 3 LPM table are decided based on the selected profile. You can configure and select the profiles that best suits your network environment needs.

1. To configure the profile that you want, enter the following statement:

```
[edit]
user@host# set chassis forwarding-options profile-name profile-name
```

2. Commit the profile.

```
[edit]
user@host# commit
```

NOTE: When you configure and commit a profile, the Packet Forwarding Engine (PFE) process restarts and all the data interfaces on the router go down and come back up.

The settings for `l2-profile-three` are configured by default. That is, if you do not configure the `forwarding-options chassis` *`profile-name`* statement, the `l2-profile-three` profile settings are configured.

Configuring Forwarding Mode on Switches

By default, switches forward packets using store-and-forward mode. You can configure all the interfaces to use the cut-through mode instead, and thereby reduce the time of packet switching.

'

To enable cut-through switching mode, enter the following statement:

```
[edit forwarding-options]
user@switch# set cut-through
```

SEE ALSO

> *cut-through*

Disabling Layer 2 Learning and Forwarding

Disabling dynamic MAC learning on an MX Series router or an EX Series switch prevents all the logical interfaces on the router or switch from learning source and destination MAC addresses.

To disable MAC learning for an MX Series router or an EX Series switch, include the `global-no-mac-learning` statement at the `[edit protocols l2-learning]` hierarchy level:

```
[edit protocols l2-learning]
global-no-mac-learning;
```

For information about how to configure a virtual switch, see *Configuring a Layer 2 Virtual Switch* .

SEE ALSO

> *Understanding Layer 2 Learning and Forwarding*
>
> *Configuring the MAC Table Timeout Interval*
>
> *Enabling MAC Accounting*
>
> *Limiting the Number of MAC Addresses Learned from Each Logical Interface*

Change History Table

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

Release	Description
18.1R1	Starting with Junos OS Release 18.1R1 on the QFX5210-64C switch, for all these profiles, except for the `lpm-profile`, the longest prefix match (LPM) table size is equal to 32,000 IPv4 entries.
18.1R1	Starting with Junos OS Release 18.3R1 on the QFX5120 and EX4650 switches, for all these profiles except for the `lpm-profile`, the longest prefix match (LPM) table size is equal to 32,000 IPv4 entries.
18.1R1	Starting with Junos OS Release 18.1R1 on QFX5210 switches, you can configure TCAM space to allocate a maximum of 8,000 IPv6 entries with prefix lengths of 65 or longer. The default value is 2,000 entries.
18.1R1	Starting with Junos Release 18.1R1, you cannot set configure a prefix for the `num-65-127-prefix` statement on non-LPM profiles. You can only enable or disable the `prefix-65-127-disable` statement for the `lpm-profile`.
14.1X53-D40	Starting with Junos OS Releases 14.1X53-D40, 15.1R5, and 16.1R3, for a Virtual Chassis or Virtual Chassis Fabric (VCF) comprised of EX4600 or QFX5100 switches, the Packet Forwarding Engine in member switches does not automatically restart upon configuring and committing a unified forwarding table profile change.
13.2X51-D15	Starting with Junos OS Release 13.2X51-D15, you can configure TCAM space to allocate a maximum of 4,000 IPv6 entries with prefix lengths of 65 or longer. The default value is 1,000 entries.
13.2X51-D15	Starting with Junos OS Release 13.2X51-D15, you can also allocate either less or no memory for IPv6 prefixes with lengths in the range /65 through /127, depending on which profile is configured.
13.2X51-D15	Starting in Junos OS Release 13.2X51-D15, you can configure the switch to allocate forwarding table memory for as many as 4,000 IPv6 prefixes with lengths in the range /65 through /127 for any profile other than the `lpm-profile` or `custom-profile`.
13.2X51-D15	Starting with Junos OS Release 13.2X51-D15, you can use the `num-65-127-prefix` statement to allocate entries.

9

CHAPTER

Configuring Bridging and VLANs

IN THIS CHAPTER

Bridging and VLANs

IN THIS SECTION

Understanding Bridging and VLANs on Switches

IN THIS SECTION

Network switches use Layer 2 bridging protocols to discover the topology of their LAN and to forward traffic toward destinations on the LAN. This topic explains the following concepts regarding bridging and VLANs:

NOTE: For Ethernet, Fast Ethernet, Tri-Rate Ethernet copper, Gigabit Ethernet, 10-Gigabit Ethernet, and aggregated Ethernet interfaces supporting VPLS, the Junos OS supports a subset of the IEEE 802.1Q standard for channelizing an Ethernet interface into multiple logical interfaces, allowing many hosts to be connected to the same Gigabit Ethernet switch, but preventing them from being in the same routing or bridging domain.

Benefits of Using VLANs

In addition to reducing traffic and thereby speeding up the network, VLANs have the following advantages:

- VLANs provide segmentation services traditionally provided by routers in LAN configurations, thereby reducing hardware equipment costs.
- Packets coupled to a VLAN can be reliably identified and sorted into different domains. You can contain broadcasts within parts of the network, thereby freeing up network resources. For example, when a DHCP server is plugged into a switch and starts broadcasting its presence, you can prevent some hosts from accessing it by using VLANs to split up the network.
- For security issues, VLANs provide granular control of the network because each VLAN is identified by a single IP subnetwork. All packets passing in and out of a VLAN are consistently tagged with the VLAN ID of that VLAN, thereby providing easy identification, because a VLAN ID on a packet cannot be altered.
- VLANs react quickly to host relocation—this is also due to the persistent VLAN tag on packets.

- On an Ethernet LAN, all network nodes must be physically connected to the same network. In VLANs, the physical location of nodes is not important—you can group network devices in any way that makes sense for your organization, such as by department or business function, types of network nodes, or physical location.

History of VLANs

Ethernet LANs were originally designed for small, simple networks that primarily carried text. However, over time, the type of data carried by LANs grew to include voice, graphics, and video. This more complex data, when combined with the ever-increasing speed of transmission, eventually became too much of a load for the original Ethernet LAN design. Multiple packet collisions were significantly slowing down the larger LANs.

The IEEE 802.1D-2004 standard helped evolve Ethernet LANs to cope with the higher data and transmission requirements by defining the concept of *transparent bridging* (generally called simply *bridging*). Bridging divides a single physical LAN (now called a single *broadcast domain*) into two or more virtual LANs, or VLANs. Each *VLAN* is a collection of some of the LAN nodes grouped together to form individual broadcast domains.

When VLANs are grouped logically by function or organization, a significant percentage of data traffic stays within the VLAN. This relieves the load on the LAN because all traffic no longer has to be forwarded to all nodes on the LAN. A VLAN first transmits packets within the VLAN, thereby reducing the number of packets transmitted on the entire LAN. Because packets whose origin and destination are in the same VLAN are forwarded only within the local VLAN, packets that are not destined for the local VLAN are the only ones forwarded to other broadcast domains. This way, bridging and VLANs limit the amount of traffic flowing across the entire LAN by reducing the possible number of collisions and packet retransmissions within VLANs and on the LAN as a whole.

How Bridging of VLAN Traffic Works

Because the objective of the IEEE 802.1D-2004 standard was to reduce traffic and therefore reduce potential transmission collisions for Ethernet, a system was implemented to reuse information. Instead of having a switch go through a location process every time a frame is sent to a node, the transparent bridging protocol allows a switch to record the location of known nodes. When packets are sent to nodes, those destination node locations are stored in address-lookup tables called *Ethernet switching tables*. Before sending a packet, a switch using bridging first consults the switching tables to see if that node has already been located. If the location of a node is known, the frame is sent directly to that node.

Transparent bridging uses five mechanisms to create and maintain Ethernet switching tables on the switch:

- Learning
- Forwarding

- Flooding
- Filtering
- Aging

The key bridging mechanism used by LANs and VLANs is *learning*. When a switch is first connected to an Ethernet LAN or VLAN, it has no information about other nodes on the network. As packets are sent, the switch learns the embedded MAC addresses of the sending nodes and stores them in the Ethernet switching table, along with two other pieces of information—the interface (or port) on which the traffic was received on the destination node and the time the address was learned.

Learning allows switches to then do *forwarding*. By consulting the Ethernet switching table to see whether the table already contains the frame's destination MAC address, switches save time and resources when forwarding packets to the known MAC addresses. If the Ethernet switching table does not contain an entry for an address, the switch uses flooding to learn that address.

Flooding finds a particular destination MAC address without using the Ethernet switching table. When traffic originates on the switch and the Ethernet switching table does not yet contain the destination MAC address, the switch first floods the traffic to all other interfaces within the VLAN. When the destination node receives the flooded traffic, it can send an acknowledgment packet back to the switch, allowing it to learn the MAC address of the node and add the address to its Ethernet switching table.

Filtering, the fourth bridging mechanism, is how broadcast traffic is limited to the local VLAN whenever possible. As the number of entries in the Ethernet switching table grows, the switch pieces together an increasingly complete picture of the VLAN and the larger LAN—it learns which nodes are in the local VLAN and which are on other network segments. The switch uses this information to filter traffic. Specifically, for traffic whose source and destination MAC addresses are in the local VLAN, filtering prevents the switch from forwarding this traffic to other network segments.

To keep entries in the Ethernet switching table current, the switch uses a fifth bridging mechanism, *aging*. Aging is the reason that the Ethernet switching table entries include timestamps. Each time the switch detects traffic from a MAC address, it updates the timestamp. A timer on the switch periodically checks the timestamp, and if it is older than a user-configured value, the switch removes the node's MAC address from the Ethernet switching table. This aging process eventually flushes unavailable network nodes out of the Ethernet switching table.

Packets Are Either Tagged or Untagged

When an Ethernet LAN is divided into VLANs, each VLAN is identified by a unique 802.1Q ID. The number of available VLANs and VLAN IDs are listed below:

- On a switch running ELS software, except EX4100 switch, you can configure 4094 VLANs using VLAN IDs 1 through 4094. EX4100 switches support 1022 VLANs. VLAN IDs 0 and 4095 are reserved by Junos OS and cannot be assigned.

- On a switch running non-ELS software, you can configure 4094 VLANs using VLAN IDs 1-4094.

Ethernet packets include a tag protocol identifier (TPID) EtherType field, which identifies the protocol being transported. When a device within a VLAN generates a packet, this field includes a value of 0x8100, which indicates that the packet is a VLAN-tagged packet. The packet also has a VLAN ID field that includes the unique 802.1Q ID, which identifies the VLAN to which the packet belongs.

Junos OS switches support the TPID value 0x9100 for Q-in-Q on switches. In addition to the TPID EtherType value of 0x8100, EX Series switches that do not support the Enhanced Layer 2 Software (ELS) configuration style also support values of 0x88a8 (Provider Bridging and Shortest Path Bridging) and 0x9100 (Q-inQ).

For a simple network that has only a single VLAN, all packets include a default 802.1Q tag, which is the only VLAN membership that does not mark the packet as tagged. These packets are untagged packets.

NOTE: Q-in-Q tunnelling is not supported on NFX150 devices.

Switch Interface Modes—Access, Trunk, or Tagged Access

Ports, or interfaces, on a switch operate in one of three modes:

- Access mode
- Trunk mode
- Tagged-access mode

Access Mode

An interface in access mode connects a switch to a single network device, such as a desktop computer, an IP telephone, a printer, a file server, or a security camera. Access interfaces accept only untagged packets.

By default, when you boot a switch that runs Junos OS that supports ELS and uses the factory default configuration, or when you boot the switch and do not explicitly configure a port mode, all interfaces on the switch are in access mode and accept only untagged packets from the `default` VLAN. You can optionally configure another VLAN and use that VLAN instead of the `default` VLAN.

On an EX Series switch that does not support ELS, the `default` VLAN is not created. Therefore, on such switches, you must explicitly configure at least one VLAN, even if your network is simple and you want only one broadcast domain to exist. You must also explicitly configure the interface in access mode. After you assign the interface to a VLAN, the interface functions in access mode.

For switches that run either type of software, you can also configure a trunk port or interface to accept untagged packets from a user-configured VLAN. For details about native VLAN, see "Trunk Mode and Native VLAN" on page 145.

Trunk Mode

Trunk mode interfaces are generally used to connect switches to one another. Traffic sent between switches can then consist of packets from multiple VLANs, with those packets multiplexed so that they can be sent over the same physical connection. Trunk interfaces usually accept only tagged packets and use the VLAN ID tag to determine both the packets' VLAN origin and VLAN destination.

On an EX Series switch that runs software that does not support ELS, an untagged packet is not recognized on a trunk port unless you configure additional settings on that port.

On a switch that runs Junos OS that supports ELS, a trunk port recognizes untagged control packets for protocols such as the Link Aggregation Control Protocol (LACP) and the Link Layer Discovery Protocol (LLDP). However, the trunk port does not recognize untagged data packets unless you configure additional settings on that port.

NOTE: LACP is not supported on NFX150 devices.

In the rare case where you want untagged packets to be recognized by a trunk port on switches that run either type of software, you must configure the single VLAN on a trunk port as a *native VLAN*. For more information about native VLANs, see "Trunk Mode and Native VLAN" on page 145.

Trunk Mode and Native VLAN

On an EX Series switch that runs Junos OS that does not support ELS, a trunk port does not recognize packets that do not include VLAN tags, which are also known an untagged packets. On a switch that runs Junos OS that supports ELS, a trunk port recognizes untagged control packets, but it does not recognize untagged data packets. With native VLAN configured, untagged packets that a trunk port normally does not recognize are sent over the trunk interface. In a situation where packets pass from a device, such as an IP phone or printer, to a switch in access mode, and you want those packets sent from the switch over a trunk port, use native VLAN mode. Create a native VLAN by configuring a VLAN ID for it, and specify that the trunk port is a member of the native VLAN.

The switch's trunk port will then treat those packets differently than the other tagged packets. For example, if a trunk port has three VLANs, 10, 20, and 30, assigned to it with VLAN 10 being the native VLAN, packets on VLAN 10 that leave the trunk port on the other end have no 802.1Q header (tag).

To configure the native VLAN identifier, see "Layer 2 Networking" on page 2.

You can have the switch add tags for the untagged packets and remove tags from the tagged packets. For example, if you have two interfaces, and an untagged packet passes into the first interface with a

user-configured native VLAN ID, it passes out of the second interface with the same VLAN ID as the tagged packet. A tagged packet that passes into the second interface passes out of the first interface as an untagged packet.

Tagged-Access Mode

Starting in Junos OS Release 15.2, only SRX firewalls and EX Series switches that run Junos OS and in the non-ELS configuration style support tagged-access mode.

Maximum VLANs and VLAN Members Per Switch

Starting in Junos OS Release 17.3 on QFX10000 switches, the number of vmembers has increased to 256k for integrated routing and bridging interfaces and aggregated Ethernet interfaces.

Starting in Junos OS Release 22.3R2-S2, the number of active VLANs allowed for EX4100-48t is less than 1000.

The number of VLANs supported per switch varies for each switch. Use the configuration-mode command `set vlans` *`vlan-name`* `vlan-id ?` to determine the maximum number of VLANs allowed on a switch. You cannot exceed this VLAN limit because you have to assign a specific ID number when you create a VLAN—you could overwrite one of the numbers, but you cannot exceed the limit.

You can, however, exceed the recommended VLAN member maximum for a switch.

On an EX Series switch that runs Junos OS that does not support the ELS configuration style, the maximum number of VLAN members allowed on the switch is eight times the maximum number of VLANs that the switch supports (vmember limit = vlan max * 8). If the configuration of the switch exceeds the recommended VLAN member maximum, a warning message appears when you commit the configuration. If you commit the configuration despite the warning, the commit succeeds, but there is a risk of the Ethernet switching process (eswd) failing as a result of memory allocation failure.

On most switches running Junos OS that supports ELS, the maximum number of VLAN members allowed on the switch is 24 times the maximum number of VLANs that the switch supports (vmember limit = vlan max * 24). If the configuration of the switch exceeds the recommended VLAN member maximum, a warning message appears in the system log (syslog).

The maximum number of VLAN members allowed on these switches are as follows:

- EX2300—8 times the maximum number of VLANs that the switch supports (vmember limit = *vlan max* * 8)
- EX3400—16 times the maximum number of VLANs that the switch supports (vmember limit = *vlan max* * 16)

- EX4100—64 times the maximum number of VLANs that the switch supports (vmember limit = *vlan max* * 1)
- EX4300—24 times the maximum number of VLANs that the switch supports (vmember limit = *vlan max* * 24)
- EX4400—64 times the maximum number of VLANs that the switch supports (vmember limit = *vlan max* * 1)

A QFabric system supports up to 131,008 VLAN members (vmembers) on a single network node group, server node group, or redundant server node group. The number of vmembers is calculated by multiplying the maximum number of VLANs by 32.

For example, to calculate how many interfaces are required to support 4,000 VLANs, divide the maximum number of vmembers (128,000) by the number of configured VLANs (4,000). In this case, 32 interfaces are required.

On network Node groups and server Node groups, you can configure link aggregation groups (LAGs) across multiple interfaces. Each LAG and VLAN combination is considered a vmember.

NOTE: LAG is not supported on NFX150 devices.

A Virtual Chassis Fabric supports up to 512,000 vmembers. The number of vmembers is based on the number of VLANs, and the number of interfaces configured in each VLAN.

NOTE: You cannot configure a default VLAN on NFX150 devices.

Assigning Traffic to VLANs

You can assign traffic on any switch to a particular VLAN by referencing either the interface port of the traffic or the MAC addresses of devices sending traffic.

NOTE: Two logical interfaces that are configured on the same physical interface cannot be mapped to the same VLAN.

Assign VLAN Traffic According to the Interface Port Source

This method is most commonly used to assign traffic to VLANs. In this case, you specify that all traffic received on a particular switch interface is assigned to a specific VLAN. You configure this VLAN assignment when you configure the switch, by using either the VLAN number (called a VLAN ID) or by using the VLAN name, which the switch then translates into a numeric VLAN ID. This method is referred

to ascreating a VLAN membership,because it is the most commonly used method, and is used for switches running Junos OS that supports ELS. For EX Series switches running Junos OS that do not support ELS, you can configure the switch only by using the VLAN number.

Assign VLAN Traffic According to the Source MAC Address

In this case, all traffic destined for a specific MAC address is forwarded to a specific egress interface (next hop) on the switch. MAC-based VLANs are either static (via configuration) or dynamic (via 802.1X authentication).

To configure a static MAC-based VLAN on a switch that supports ELS, see "Adding a Static MAC Address Entry to the Ethernet Switching Table" on page 71.To configure a static MAC-based VLAN on a switch that does not support ELS, see "Adding a Static MAC Address Entry to the Ethernet Switching Table" on page 71.

For information about using 802.1X authentication to authenticate end devices and allow access to dynamic VLANs configured on a RADIUS server, see *Understanding Dynamic VLAN Assignment Using RADIUS Attributes*. You can optionally implement this feature to offload the manual assignment of VLAN traffic to automated RADIUS server databases.

Forwarding VLAN Traffic

To pass traffic within a VLAN, the switch uses Layer 2 forwarding protocols, including IEEE 802.1Q spanning-tree protocols.

To pass traffic between two VLANs, the switch uses standard Layer 3 routing protocols, such as static routing, OSPF, and RIP. The same interfaces that support Layer 2 bridging protocols also support Layer 3 routing protocols, providing multilayer switching.

To pass traffic from a single device on an access port to a switch and then pass those packets on a trunk port, use the native mode configuration previously discussed under "Trunk Mode" on page 145.

VLANs Communicate with Integrated Routing and Bridging Interfaces

Traditionally, switches sent traffic to hosts that were part of the same broadcast domain (VLAN) but routers were needed to route traffic from one broadcast domain to another. Also, only routers performed other Layer 3 functions such as traffic engineering.

Switches perform inter-VLAN routing functions using an integrated routing and bridging (IRB) interface named irb. These interfaces detect both MAC addresses and IP addresses and route data to Layer 3 interfaces, thereby frequently eliminating the need to have both a switch and a router.

To configure the IRB interface, see "Layer 2 Networking" on page 2.

VPLS Ports

You can configure VPLS ports in a virtual switch instead of a dedicated routing instance of type **vpls** so that the logical interfaces of the Layer 2 VLANs in the virtual switch can handle VPLS routing instance traffic. Packets received on a Layer 2 trunk interface are forwarded within a VLAN that has the same VLAN identifier.

SEE ALSO

> *Understanding FCoE*
>
> *Interfaces Overview for Switches*
>
> Understanding Multiple VLAN Registration Protocol (MVRP) | **829**
>
> Understanding Integrated Routing and Bridging | **765**

Enabling a VLAN

A VLAN must include a set of logical interfaces that participate in Layer 2 learning and forwarding. You can optionally configure a VLAN identifier and a Layer 3 interface for the VLAN to also support Layer 3 IP routing. To configure a VLAN, see "Layer 2 Networking" on page 2.

To enable a VLAN, include the following statements:

```
[edit]
vlans {
    vlan-name {
        interface interface-name;
        l3-interface interface-name;
        vlan-id (none | all | number);
        vlan-id-list [ vlan-id-numbers ];
        vlan-tags outer number inner number);
    }
}
```

You cannot use the slash (/) character in VLAN names. If you do, the configuration does not commit and an error is generated.

For the `vlan-id` statement, you can specify either a valid VLAN identifier or the **none** or **all** options.

To include one or more logical interfaces in the VLAN, specify an *interface-name* for an Ethernet interface you configured at the `[edit interfaces]` hierarchy level.

NOTE: A maximum of 4096 active logical interfaces are supported for a VLAN or on each mesh group in a virtual private LAN service (VPLS) instance configured for Layer 2 bridging.

By default, each VLAN maintains a Layer 2 forwarding database that contains media access control (MAC) addresses learned from packets received on the ports that belong to the VLAN. You can modify Layer 2 forwarding properties, for example, disabling MAC learning for the entire system or a VLAN, adding static MAC addresses for specific logical interfaces, and limiting the number of MAC addresses learned by the entire system, the VLAN, or a logical interface.

You can also configure spanning tree protocols to prevent forwarding loops.

Configuring VLANs on Switches with Enhanced Layer 2 Support

Switches use VLANs to make logical groupings of network nodes with their own broadcast domains. You can use VLANs to limit the traffic flowing across the entire LAN and reduce collisions and packet retransmissions. To configure layer 2 interfaces, see "Layer 2 Networking" on page 2.

NOTE: These steps are also applicable to ACX routers, SRX firewalls, and NFX series devices.

NOTE: For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15.

NOTE: Starting with Junos OS Release 17.1R3, on QFX10000 switches, you cannot configure an interface with both `family ethernet-switching` and `flexible-vlan-tagging`. This configuration is not supported, and a warning will be issued if you try to commit this configuration.

NOTE: Two logical interfaces that are configured on the same physical interface cannot be mapped to the same VLAN or have an overlapping VLAN ID.

For each endpoint on the VLAN, configure the following VLAN parameters on the corresponding interface:

1. Specify the description of the VLAN:

```
[edit interfaces interface-name unit 0]
user@switch# set description vlan-description
```

2. Specify the unique name of the VLAN:

> **NOTE**: Switches that run Junos OS that does not support the ELS configuration style do not support a default VLAN. Therefore, on such switches, you must explicitly configure at least one VLAN, even if your network is simple and you want only one broadcast domain to exist.

> **NOTE**: On QFX5100 switches running Junos OS Release 14.1X53-D46 or earlier, when you configure an interface under a VLAN but do not specify the name of the VLAN, the system will not issue a commit error.

```
[edit interfaces interface-name unit 0]
user@switch# set family ethernet-switching vlan members vlan-name
```

3. Create the subnet for the VLAN:

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

> **NOTE**: The **family inet** option is not supported on NFX150 devices.

4. Configure the VLAN tag ID or VLAN ID list for the VLAN:

```
[edit vlans]
user@switch# set vlan-name vlan-id vlan-id-number
```

or

```
[edit vlans]
user@switch# set  vlan-name  vlan-id-list [vlan-ids | vlan-id--vlan-id-]
```

5. Specify a VLAN firewall filter to be applied to incoming or outgoing packets:

```
[edit vlans]
user@switch# set vlan-name filter (input | output) filter-name
```

SEE ALSO

Example: Setting Up Basic Bridging and a VLAN on Switches | 171

Configuring IRB Interfaces on Switches | 772

Configuring VLANs on QFX Series Switches without Enhanced Layer 2 Support

Switches use VLANs to make logical groupings of network nodes with their own broadcast domains. You can use VLANs to limit the traffic flowing across the entire LAN and reduce collisions and packet retransmissions.

For each endpoint on the VLAN, configure the following VLAN parameters on the corresponding interface:

1. Specify the description of the VLAN:

```
[edit interfaces interface-name unit 0]
user@switch# set description vlan-description
```

2. Specify the unique name of the VLAN:

NOTE: In a QFabric system, do not configure “default” as the name of a VLAN. Though the QFabric system will allow you to configure and commit a VLAN with the name “default” in the current software with no commit errors, it will not work. Junos OS 12.2 and onwards will not allow you to commit a VLAN with the name “default.”

```
[edit interfaces interface-name unit 0]
user@switch# set family ethernet-switching vlan members vlan-name
```

3. Create the subnet for the VLAN:

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

4. Configure the VLAN tag ID or VLAN ID range for the VLAN:

```
[edit vlans]
user@switch# set vlan-name vlan-id vlan-id-number
```

or

```
[edit vlans]
user@switch# set vlan-name vlan-range vlan-id-low-vlan-id-high
```

5. Specify a VLAN firewall filter to be applied to incoming or outgoing packets:

```
[edit vlans]
user@switch# set vlan-name filter (input | output) filter-name
```

SEE ALSO

Example: Setting Up Basic Bridging and a VLAN on Switches | 171

Configuring IRB Interfaces on Switches | 772

Creating a Series of Tagged VLANs | 347

Example: Configuring VLANs on Security Devices

IN THIS SECTION

This example shows you how to configure a VLAN.

Requirements

Before you begin:

- Determine which interfaces to use and verify that they are in switching mode. See Understanding VLANs.
- Determine what ports to use on the device and how to segment your network. See "Ethernet Ports Switching Overview for Security Devices" on page 1061.

Overview

In this example, you create a new VLAN and then configure its attributes. You can configure one or more VLANs to perform Layer 2 switching. The Layer 2 switching functions include integrated routing and bridging (IRB) for support for Layer 2 switching and Layer 3 IP routing on the same interface. SRX Series Firewalls can function as Layer 2 switches, each with multiple switching or broadcast domains that participate in the same Layer 2 network.

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

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

```
set interfaces ge-0/0/1 unit 0 family ethernet-switching interface-mode access
set interfaces ge-0/0/1 unit 0 family ethernet-switching vlan members v10
set vlans v10 vlan-id 10
set vlans v10 l3-interface irb.10
set interfaces irb unit 10 family inet address 198.51.100.0/24
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure a VLAN:

1. Configure a Gigabit Ethernet interface or a 10-Gigabit Ethernet interface as a access interface:

   ```
   [edit interfaces]
   user@host# set ge-0/0/1 unit 0 family ethernet-switching interface-mode access
   ```

2. Assign an interface to the VLAN by specifying the logical interface (with the unit statement) and specifying the VLAN name as the member.

   ```
   [edit interfaces]
   user@host# set ge-0/0/1 unit 0 family ethernet-switching vlan members v10
   ```

3. Create the VLAN by setting the unique VLAN name and configuring the VLAN ID.

   ```
   [edit]
   user@host# set vlans v10 vlan-id 10
   ```

4. Bind a Layer 3 interface with the VLAN.

```
[edit]
user@host# set vlans v10 l3-interface irb.10
```

5. Create the subnet for the VLAN's broadcast domain.

```
[edit interfaces]
user@host# set irb unit 10 family inet address 198.51.100.0/24
```

Results

From configuration mode, confirm your configuration by entering the `show vlans` command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
[edit]
user@host# show vlans
v10 {
    vlan-id 10;
    l3-interface irb.10;
}
user@host# show interfaces
ge-0/0/1 {
    unit 0 {
        family ethernet-switching {
            interface-mode access;
            vlan {
                members v10;
            }
        }
    }
    irb {
        unit 10 {
            family inet {
                address 198.51.100.0/24;
            }
        }
```

```
    }
}
```

If you are done configuring the device, enter `commit` from configuration mode.

Verification

IN THIS SECTION

Verifying VLANs

Purpose

Verify that VLANs are configured and assigned to the interfaces.

Action

From operational mode, enter the `show vlans` command.

```
user@host> show vlans

Routing instance        VLAN name             Tag          Interfaces
default-switch          default               1

default-switch          v10                   10
                                                           ge-0/0/1.0
```

Meaning

The output shows the VLAN is configured and assigned to the interface.

Example: Setting Up Basic Bridging and a VLAN for an EX Series Switch with ELS Support

IN THIS SECTION

NOTE: This example uses Junos OS for EX Series switches with support for the Enhanced Layer 2 Software (ELS) configuration style. If your switch runs Junos OS that does not support ELS, see "Example: Setting Up Basic Bridging and a VLAN for an EX Series Switch" on page 195. For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15.

EX Series switches use bridging and virtual LANs (VLANs) to connect network devices in a LAN—desktop computers or laptops, IP telephones, printers, file servers, wireless access points, and others—and to segment the LAN into smaller broadcast domains.

This example describes how to configure basic bridging and a VLAN on an EX Series switch:

Requirements

This example uses the following hardware and software components:

- One EX Series switch
- Junos OS Release 13.2X50-D10 or later for EX Series switches

Before you set up bridging and a VLAN, be sure you have:

- Installed your EX Series switch. See the installation instructions for your switch.
- Performed the initial switch configuration. See *Connecting and Configuring an EX Series Switch (CLI Procedure)*.

Overview and Topology

IN THIS SECTION

EX Series switches connect network devices in an office LAN or a data center LAN to provide sharing of common resources such as printers and file servers and to enable wireless devices to connect to the LAN through wireless access points. Without bridging and VLANs, all devices on the Ethernet LAN are in a single broadcast domain, and all the devices detect all the packets on the LAN. Bridging creates separate broadcast domains on the LAN, creating VLANs, which are independent logical networks that group together related devices into separate network segments. The grouping of devices on a VLAN is independent of where the devices are physically located in the LAN.

To use an EX Series switch to connect network devices on a LAN, you must, at a minimum, explicitly configure at least one VLAN, even if your network is simple and you want only one broadcast domain to exist, as is the case with this example. You must also assign all needed interfaces to the VLAN, after which the interfaces function in access mode. After the VLAN is configured, you can plug access devices —such as desktop or laptop computers, IP telephones, file servers, printers, and wireless access points— into the switch, and they are joined immediately into the VLAN, and the LAN is up and running.

The topology used in this example consists of one EX4300-24P switch, which has a total of 24 ports. All ports support Power over Ethernet (PoE), which means they provide both network connectivity and electric power for the device connecting to the port. To these ports, you can plug in devices requiring PoE, such as Avaya VoIP telephones, wireless access points, and some IP cameras. (Avaya phones have a built-in hub that allows you to connect a desktop PC to the phone, so the desktop and phone in a single office require only one port on the switch.) Table 48 on page 159 details the topology used in this configuration example.

Table 48: Components of the Basic Bridging Configuration Topology

Property	Settings
Switch hardware	EX4300-24P switch, with 24 Gigabit Ethernet ports: in this example, 8 ports are used as PoE ports (ge-0/0/0 through ge-0/0/7) and 16 ports used as non-PoE ports (ge-0/0/8 through ge-0/0/23)
VLAN name	employee-vlan

Table 48: Components of the Basic Bridging Configuration Topology *(Continued)*

Property	Settings
VLAN ID	10
Connection to wireless access point (requires PoE)	ge-0/0/0
Connections to Avaya IP telephone—with integrated hub, to connect phone and desktop PC to a single port (requires PoE)	ge-0/0/1 through ge-0/0/7
Direct connections to desktop PCs and laptops (no PoE required)	ge-0/0/8 through ge-0/0/12
Connections to file servers (no PoE required)	ge-0/0/17 and ge-0/0/18
Connections to integrated printer/fax/copier machines (no PoE required)	ge-0/0/19 through ge-0/0/20
Unused ports (for future expansion)	ge-0/0/13 through ge-0/0/16, and ge-0/0/21 through ge-0/0/23

Topology

Configuration

IN THIS SECTION

To set up basic bridging and a VLAN:

Procedure

CLI Quick Configuration

To quickly configure a VLAN, copy the following commands and paste them into the switch terminal window:

```
[edit]
set vlans employee-vlan vlan-id 10
set interfaces ge-0/0/0 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces ge-0/0/1 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces ge-0/0/2 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces ge-0/0/3 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces ge-0/0/4 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces ge-0/0/5 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces ge-0/0/6 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces ge-0/0/7 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces ge-0/0/8 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces ge-0/0/9 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces ge-0/0/10 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces ge-0/0/11 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces ge-0/0/12 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces ge-0/0/17 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces ge-0/0/18 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces ge-0/0/19 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces ge-0/0/20 unit 0 family ethernet-switching vlan members employee-vlan
```

You must then plug the wireless access point into PoE-enabled port `ge-0/0/0` and the Avaya IP phones into the PoE-enabled ports **ge-0/0/1** through **ge-0/0/7**. Also, plug the PCs, file servers, and printers into ports **ge-0/0/8** through **ge-0/0/12** and **ge-0/0/17** through **ge-0/0/20**.

Step-by-Step Procedure

To set up basic bridging and a VLAN:

1. Create a VLAN named employee-vlan and specify the VLAN ID of 10 for it:

```
[edit vlans]
user@switch# set employee-vlan vlan-id 10
```

2. Assign interfaces ge-0/0/0 through ge-0/0/12, and ge-0/0/17 through ge-0/0/20 to the employee-vlan VLAN:

```
[edit interface]
user@switch# set ge-0/0/0 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set ge-0/0/1 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set ge-0/0/2 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set ge-0/0/3 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set ge-0/0/4 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set ge-0/0/5 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set ge-0/0/6 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set ge-0/0/7 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set ge-0/0/8 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set ge-0/0/9 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set ge-0/0/10 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set ge-0/0/11 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set ge-0/0/12 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set ge-0/0/17 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set ge-0/0/18 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set ge-0/0/19 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set ge-0/0/20 unit 0 family ethernet-switching vlan members employee-vlan
```

3. Connect the wireless access point to switch port ge-0/0/0.
4. Connect the seven Avaya phones to switch ports ge-0/0/1 through ge-0/0/7.
5. Connect the five PCs to ports ge-0/0/8 through ge-0/0/12.
6. Connect the two file servers to ports ge-0/0/17 and ge-0/0/18.
7. Connect the two printers to ports ge-0/0/19 and ge-0/0/20.

Results

Check the results of the configuration:

```
user@switch> show configuration
    ge-0/0/0 {
        unit 0 {
            family ethernet-switching {
                vlan {
                    members employee-vlan;
```

```
            }
        }
    }
}
ge-0/0/1 {
    unit 0 {
        family ethernet-switching {
            vlan {
                members employee-vlan;
            }
        }
    }
}
ge-0/0/2 {
    unit 0 {
        family ethernet-switching {
            vlan {
                members employee-vlan;
            }
        }
    }
}
ge-0/0/3 {
    unit 0 {
        family ethernet-switching {
            vlan {
                members employee-vlan;
            }
        }
    }
}
ge-0/0/4 {
    unit 0 {
        family ethernet-switching {
            vlan {
                members employee-vlan;
            }
        }
    }
}
ge-0/0/5 {
    unit 0 {
        family ethernet-switching {
```

```
            vlan {
                members employee-vlan;
            }
        }
    }
    ge-0/0/6 {
        unit 0 {
            family ethernet-switching {

            vlan {
                members employee-vlan;
            }
        }
    }
    ge-0/0/7 {
        unit 0 {
            family ethernet-switching {
             vlan {
                members employee-vlan;
            }
        }
    }
    ge-0/0/8 {
        unit 0 {
            family ethernet-switching {
                vlan {
                    members employee-vlan;
                }
            }
        }
    }
    ge-0/0/9 {
        unit 0 {
            family ethernet-switching {
                vlan {
                    members employee-vlan;
                }
            }
        }
    }
```

```
    ge-0/0/10 {
        unit 0 {
            family ethernet-switching {
                       vlan {
                    members employee-vlan;
                }
            }
        }
    }
    ge-0/0/11 {
        unit 0 {
            family ethernet-switching {
                       vlan {
                    members employee-vlan;
                }
            }
        }
    }
    ge-0/0/12 {
        unit 0 {
            family ethernet-switching {
                vlan {
                    members employee-vlan;
                }
            }
        }
    }
    ge-0/0/17 {
        unit 0 {
            family ethernet-switching {
                vlan {
                    members employee-vlan;
                }
            }
        }
    }
    ge-0/0/18 {
        unit 0 {
            family ethernet-switching {
                vlan {
                    members employee-vlan;
                }
            }
```

```
        }
    }
    ge-0/0/19 {
        unit 0 {
            family ethernet-switching {
                vlan {
                    members employee-vlan;
                }
            }
        }
    }
    ge-0/0/20 {
        unit 0 {
            family ethernet-switching {
                       vlan {
                    members employee-vlan;
                }
            }
        }
    }
```

Verification

IN THIS SECTION

To verify that switching is operational and that `employee-vlan` has been created, perform these tasks:

Verifying That the VLAN Has Been Created

Purpose

Verify that the VLAN named **employee-vlan** has been created on the switch.

Action

List all VLANs configured on the switch:

```
user@switch> show vlans
Routing instance        VLAN name             Tag         Interfaces
default-switch          employee-vlan            10
                                                            ge-0/0/0.0
                                                            ge-0/0/1.0
                                                            ge-0/0/2.0
                                                            ge-0/0/3.0
                                                            ge-0/0/4.0
                                                            ge-0/0/5.0
                                                            ge-0/0/6.0
                                                            ge-0/0/7.0
                                                            ge-0/0/8.0
                                                            ge-0/0/9.0
                                                            ge-0/0/10.0
                                                            ge-0/0/11.0
                                                            ge-0/0/12.0
                                                            ge-0/0/17.0
                                                            ge-0/0/18.0
                                                            ge-0/0/19.0
                                                            ge-0/0/20.0
...
```

Meaning

The `show vlans` command lists the VLANs configured on the switch. This output shows that the VLAN **employee-vlan** has been created.

Verifying That Interfaces Are Associated with the Proper VLANs

Purpose

Verify that Ethernet switching is enabled on switch interfaces and that all interfaces are included in the VLAN.

Action

List all interfaces on which switching is enabled:

```
user@switch> show ethernet-switching interfaces
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
ge-0/0/0.0                    65535                                 untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
ge-0/0/1.0                    65535                                 untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
ge-0/0/2.0                    65535                                 untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
ge-0/0/3.0                    65535                                 untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
```

```
ge-0/0/4.0                    65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                          LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
ge-0/0/5.0                    65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                          LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
ge-0/0/6.0                    65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                          LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
ge-0/0/7.0                    65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                          LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
ge-0/0/8.0                    65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                          LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
ge-0/0/9.0                    65535                                untagged
             employee-vlan 10
                              65535    Discarding
```

```
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
ge-0/0/10.0                   65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
ge-0/0/11.0                   65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
ge-0/0/12.0                   65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
ge-0/0/17.0                   65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
ge-0/0/18.0                   65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
```

```
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
ge-0/0/19.0                   65535                                 untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
ge-0/0/20.0                   65535                                 untagged
             employee-vlan 10
                              65535    Discarding
...
```

Meaning

The `show ethernet-switching interfaces` command lists all interfaces on which switching is enabled (in the **Logical interface** column), along with the VLANs that are active on the interfaces (in the **VLAN members** column). The output in this example shows all the connected interfaces, ge-0/0/0 through ge-0/0/12 and ge-0/0/17 through ge-0/0/20 and that they are all part of VLAN **employee-vlan**. Notice that the interfaces listed are the logical interfaces, not the physical interfaces. For example, the output shows ge-0/0/0.0 instead of ge-0/0/0. This is because Junos OS creates VLANs on logical interfaces, not directly on physical interfaces.

SEE ALSO

Example: Connecting Access Switches with ELS Support to a Distribution Switch with ELS Support | 224

Example: Setting Up Basic Bridging and a VLAN on Switches

IN THIS SECTION

The QFX Series products use bridging and virtual LANs (VLANs) to connect network devices—storage devices, file servers, and other LAN components—in a LAN and to segment the LAN into smaller bridging domains.

To segment traffic on a LAN into separate broadcast domains, you create separate virtual LANs (VLANs) on a switch. Each VLAN is a collection of network nodes. When you use VLANs, frames whose origin and destination are in the same VLAN are forwarded only within the local VLAN, and only frames not destined for the local VLAN are forwarded to other broadcast domains. VLANs thus limit the amount of traffic flowing across the entire LAN, reducing the possible number of collisions and packet retransmissions within the LAN.

NOTE: You cannot configure more than one logical interface that belongs to the same physical interface in the same bridge domain.

This example describes how to configure basic bridging and VLANs for the QFX Series:

Requirements

This example uses the following software and hardware components:

- Junos OS Release 11.1 or later for the QFX Series
- A configured and provisioned QFX Series product

Overview and Topology

IN THIS SECTION

To use a switch to connect network devices on a LAN, you must at a minimum configure bridging and VLANs. By default, bridging is enabled on all switch interfaces, all interfaces are in access mode, and all interfaces belong to a VLAN called **employee-vlan**, which is automatically configured. When you plug in

access devices—such as desktop computers, file servers, and printers—they are joined immediately into the **employee-vlan** VLAN, and the LAN is up and running.

The topology used in this example consists of a single QFX3500 switch, with a total of 48 10-Gbps Ethernet ports. (For the purposes of this example, the QSFP+ ports Q0-Q3, which are ports xe-0/1/0 through xe-0/1/15, are excluded.) You use the ports to connect devices that have their own power sources. Table 1 details the topology used in this configuration example.

Table 49: Components of the Basic Bridging Configuration Topology

Property	Settings
Switch hardware	QFX3500 switch, with 48 10-Gbps Ethernet ports
VLAN name	**employee-vlan**
VLAN ID	10
Connections to file servers	**xe-0/0/17** and **xe-0/0/18**
Direct connections to desktop PCs and laptops	**xe-0/0/0** through **xe-0/0/16**
Connections to integrated printer/fax/copier machines	**xe-0/0/19** through **xe-0/0/40**
Unused ports	**xe-0/0/41** through **xe-0/0/47**

Topology

Configuration

IN THIS SECTION

- Procedure | **174**

Procedure

CLI Quick Configuration

To quickly configure a VLAN, copy the following commands and paste them into the switch terminal window:

```
[edit]
set vlans employee-vlan vlan-id 10
set interfaces xe-0/0/0 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/1 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/2 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/3 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/4 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/5 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/6 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/7 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/8 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/9 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/10 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/11 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/12 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/13 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/14 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/15 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/16 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/17 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/18 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/19 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/20 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/21 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/22 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/23 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/24 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/25 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/26 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/27 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/28 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/29 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/30 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/31 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/32 unit 0 family ethernet-switching vlan members employee-vlan
```

```
set interfaces xe-0/0/33 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/34 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/35 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/36 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/37 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/38 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/39 unit 0 family ethernet-switching vlan members employee-vlan
set interfaces xe-0/0/40 unit 0 family ethernet-switching vlan members employee-vlan
```

Step-by-Step Procedure

To set up basic bridging and a VLAN:

1. Create a VLAN named employee-vlan and specify the VLAN ID of 10 for it:

```
[edit vlans]
user@switch# set employee-vlan vlan-id 10
```

2. Assign interfaces xe-0/0/0 through xe-0/0/40 to the employee-vlan VLAN:

```
[edit interface]
user@switch# set xe-0/0/0 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/1 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/2 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/3 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/4 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/5 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/6 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/7 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/8 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/9 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/10 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/11 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/12 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/13 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/14 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/15 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/16 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/17 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/18 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/19 unit 0 family ethernet-switching vlan members employee-vlan
```

```
user@switch# set xe-0/0/20 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/21 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/22 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/23 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/24 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/25 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/26 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/27 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/28 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/29 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/30 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/31 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/32 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/33 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/34 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/35 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/36 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/37 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/38 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/39 unit 0 family ethernet-switching vlan members employee-vlan
user@switch# set xe-0/0/40 unit 0 family ethernet-switching vlan members employee-vlan
```

3. Connect the two file servers to ports xe-0/0/17 and xe-0/0/18.
4. Connect the desktop PCs and laptops to ports xe-0/0/0 through xe-0/0/16.
5. Connect the integrated printer/fax/copier machines to ports xe-0/0/19 through xe-0/0/40.

Results

Check the results of the configuration:

```
user@switch> show configuration
    xe-0/0/0 {
        unit 0 {
            family ethernet-switching {
                vlan {
                    members employee-vlan;
                }
            }
        }
    }
```

```
    xe-0/0/1 {
        unit 0 {
            family ethernet-switching {
                vlan {
                    members employee-vlan;
                }
            }
        }
    }
    xe-0/0/2 {
        unit 0 {
            family ethernet-switching {
                vlan {
                    members employee-vlan;
                }
            }
        }
    }
    xe-0/0/3 {
        unit 0 {
            family ethernet-switching {
                vlan {
                    members employee-vlan;
                }
            }
        }
    }
    xe-0/0/4 {
        unit 0 {
            family ethernet-switching {
                vlan {
                    members employee-vlan;
                }
            }
        }
    }
    xe-0/0/5 {
        unit 0 {
            family ethernet-switching {
                vlan {
                    members employee-vlan;
                }
            }
```

```
        }
    }
    xe-0/0/6 {
        unit 0 {
            family ethernet-switching {

                vlan {
                    members employee-vlan;
                }
            }
        }
    }
    xe-0/0/7 {
        unit 0 {
            family ethernet-switching {
                 vlan {
                    members employee-vlan;
                }
            }
        }
    }
    xe-0/0/8 {
        unit 0 {
            family ethernet-switching {
                vlan {
                    members employee-vlan;
                }
            }
        }
    }
    xe-0/0/9 {
        unit 0 {
            family ethernet-switching {
                vlan {
                    members employee-vlan;
                }
            }
        }
    }
    xe-0/0/10 {
        unit 0 {
            family ethernet-switching {
                       vlan {
```

```
                members employee-vlan;
            }
        }
    }
}
xe-0/0/11 {
    unit 0 {
        family ethernet-switching {
                   vlan {
                members employee-vlan;
            }
        }
    }
}
xe-0/0/12 {
    unit 0 {
        family ethernet-switching {
            vlan {
                members employee-vlan;
            }
        }
    }
}
xe-0/0/13 {
    unit 0 {
        family ethernet-switching {
            vlan {
                members employee-vlan;
            }
        }
    }
}
xe-0/0/14 {
    unit 0 {
        family ethernet-switching {
            vlan {
                members employee-vlan;
            }
        }
    }
}
xe-0/0/15 {
    unit 0 {
```

```
            family ethernet-switching {
                vlan {
                    members employee-vlan;
                }
            }
        }
    }
    xe-0/0/16 {
        unit 0 {
            family ethernet-switching {
                       vlan {
                    members employee-vlan;
                }
            }
        }
    }
    xe-0/0/17 {
        unit 0 {
            family ethernet-switching {
                       vlan {
                    members employee-vlan;
                }
            }
        }
    }
    xe-0/0/18 {
        unit 0 {
            family ethernet-switching {
                       vlan {
                    members employee-vlan;
                }
            }
        }
    }
    xe-0/0/19 {
        unit 0 {
            family ethernet-switching {
                       vlan {
                    members employee-vlan;
                }
            }
        }
    }
```

```
    xe-0/0/20 {
        unit 0 {
            family ethernet-switching {
                       vlan {
                    members employee-vlan;
                }
            }
        }
    }
    xe-0/0/21 {
        unit 0 {
            family ethernet-switching {
                       vlan {
                    members employee-vlan;
                }
            }
        }
    }
    xe-0/0/22 {
        unit 0 {
            family ethernet-switching {
                       vlan {
                    members employee-vlan;
                }
            }
        }
    }
    xe-0/0/23 {
        unit 0 {
            family ethernet-switching {
                       vlan {
                    members employee-vlan;
                }
            }
        }
    }
    xe-0/0/24 {
        unit 0 {
            family ethernet-switching {
                       vlan {
                    members employee-vlan;
                }
            }
```

```
        }
    }
    xe-0/0/25 {
        unit 0 {
            family ethernet-switching {
                       vlan {
                    members employee-vlan;
                }
            }
        }
    }
    xe-0/0/26 {
        unit 0 {
            family ethernet-switching {
                       vlan {
                    members employee-vlan;
                }
            }
        }
    }
    xe-0/0/27 {
        unit 0 {
            family ethernet-switching {
                       vlan {
                    members employee-vlan;
                }
            }
        }
    }
    xe-0/0/28 {
        unit 0 {
            family ethernet-switching {
                       vlan {
                    members employee-vlan;
                }
            }
        }
    }
    xe-0/0/29 {
        unit 0 {
            family ethernet-switching {
                       vlan {
                    members employee-vlan;
```

```
            }
        }
    }
}
xe-0/0/30 {
    unit 0 {
        family ethernet-switching {
                   vlan {
                members employee-vlan;
            }
        }
    }
}
xe-0/0/31 {
    unit 0 {
        family ethernet-switching {
                   vlan {
                members employee-vlan;
            }
        }
    }
}
xe-0/0/32 {
    unit 0 {
        family ethernet-switching {
                   vlan {
                members employee-vlan;
            }
        }
    }
}
xe-0/0/33 {
    unit 0 {
        family ethernet-switching {
                   vlan {
                members employee-vlan;
            }
        }
    }
}
xe-0/0/34 {
    unit 0 {
        family ethernet-switching {
```

```
                    vlan {
                members employee-vlan;
            }
        }
    }
}
xe-0/0/35 {
    unit 0 {
        family ethernet-switching {
                    vlan {
                members employee-vlan;
            }
        }
    }
}
xe-0/0/36 {
    unit 0 {
        family ethernet-switching {
                    vlan {
                members employee-vlan;
            }
        }
    }
}
xe-0/0/37 {
    unit 0 {
        family ethernet-switching {
                    vlan {
                members employee-vlan;
            }
        }
    }
}
xe-0/0/38 {
    unit 0 {
        family ethernet-switching {
                    vlan {
                members employee-vlan;
            }
        }
    }
}
xe-0/0/39 {
```

```
        unit 0 {
            family ethernet-switching {
                       vlan {
                    members employee-vlan;
                }
            }
        }
    }
    xe-0/0/40 {
        unit 0 {
            family ethernet-switching {
                       vlan {
                    members employee-vlan;
                }
            }
        }
    }
```

Verification

IN THIS SECTION

To verify that switching is operational and that `employee-vlan` has been created, perform these tasks:

Verifying That the VLAN Has Been Created

Purpose

Verify that the VLAN named **employee-vlan** has been created on the switch.

Action

List all VLANs configured on the switch:

```
user@switch> show vlans
Routing instance        VLAN name             Tag         Interfaces
default-switch          employee-vlan            10
                                                            xe-0/0/0.0
                                                            xe-0/0/1.0
                                                            xe-0/0/2.0
                                                            xe-0/0/3.0
                                                            xe-0/0/4.0
                                                            xe-0/0/5.0
                                                            xe-0/0/6.0
                                                            xe-0/0/7.0
                                                            xe-0/0/8.0
                                                            xe-0/0/9.0
                                                            xe-0/0/10.0
                                                            xe-0/0/11.0
                                                            xe-0/0/12.0
                                                            xe-0/0/13.0
                                                            xe-0/0/14.0
                                                            xe-0/0/15.0
                                                            xe-0/0/16.0
                                                            xe-0/0/17.0
                                                            xe-0/0/18.0
                                                            xe-0/0/19.0
                                                            xe-0/0/20.0
                                                            xe-0/0/21.0
                                                            xe-0/0/22.0
                                                            xe-0/0/23.0
                                                            xe-0/0/24.0
                                                            xe-0/0/25.0
                                                            xe-0/0/26.0
                                                            xe-0/0/27.0
                                                            xe-0/0/28.0
                                                            xe-0/0/29.0
                                                            xe-0/0/30.0
                                                            xe-0/0/31.0
                                                            xe-0/0/32.0
                                                            xe-0/0/33.0
                                                            xe-0/0/34.0
```

```
                                                    xe-0/0/35.0
                                                    xe-0/0/36.0
                                                    xe-0/0/37.0
                                                    xe-0/0/38.0
                                                    xe-0/0/39.0
                                                    xe-0/0/40.0
...
```

Meaning

The `show vlans` command lists the VLANs configured on the switch. This output shows that the VLAN **employee-vlan** has been created.

Verifying That Interfaces Are Associated with the Proper VLANs

Purpose

Verify that Ethernet switching is enabled on switch interfaces and that all interfaces are included in the VLAN.

Action

List all interfaces on which switching is enabled:

```
user@switch> show ethernet-switching interfaces
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/0.0                    65535                                 untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/1.0                    65535                                 untagged
             employee-vlan 10
                              65535    Discarding
```

```
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/2.0                    65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/3.0                    65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/4.0                    65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/5.0                    65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/6.0                    65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
```

```
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/7.0                    65535                                 untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/8.0                    65535                                 untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/9.0                    65535                                 untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/10.0                   65535                                 untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/11.0                   65535                                 untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/12.0                   65535                                 untagged
```

```
             employee-vlan 10
                                65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                          LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/13.0                   65535                                 untagged
             employee-vlan 10
                                65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                          LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/14.0                   65535                                 untagged
             employee-vlan 10
                                65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                          LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/15.0                   65535                                 untagged
             employee-vlan 10
                                65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                          LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/16.0                   65535                                 untagged
             employee-vlan 10
                                65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                          LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/17.0                   65535                                 untagged
             employee-vlan 10
                                65535    Discarding
Routing Instance Name : default-switch
```

```
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/18.0                   65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/19.0                   65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/20.0                   65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/21.0                   65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/22.0                   65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
```

```
interface    members          limit    state       interface flags
xe-0/0/23.0                   65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state       interface flags
xe-0/0/24.0                   65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state       interface flags
xe-0/0/25.0                   65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state       interface flags
xe-0/0/26.0                   65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state       interface flags
xe-0/0/27.0                   65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state       interface flags
xe-0/0/28.0                   65535                                untagged
             employee-vlan 10
```

```
                            65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/29.0                   65535                                 untagged
             employee-vlan 10
                            65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/30.0                   65535                                 untagged
             employee-vlan 10
                            65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/31.0                   65535                                 untagged
             employee-vlan 10
                            65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/32.0                   65535                                 untagged
             employee-vlan 10
                            65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/33.0                   65535                                 untagged
             employee-vlan 10
                            65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
```

```
                        LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/34.0                   65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                        LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/35.0                   65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                        LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/36.0                   65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                        LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/37.0                   65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                        LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/38.0                   65535                                untagged
             employee-vlan 10
                              65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                        LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
```

```
xe-0/0/39.0                  65535                                untagged
             employee-vlan 10
                             65535    Discarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/0/40.0                  65535                                untagged
             employee-vlan 10
                             65535    Discarding
...
```

Meaning

The `show ethernet-switching interfaces` command lists all interfaces on which switching is enabled (in the **Logical interface** column), along with the VLANs that are active on the interfaces (in the **VLAN members** column). The output in this example shows all the connected interfaces, xe-0/0/0 through xe-0/0/40, are all part of VLAN **employee-vlan**. Notice that the interfaces listed are the logical interfaces, not the physical interfaces. For example, the output shows xe-0/0/0.0 instead of xe-0/0/0. This is because Junos OS creates VLANs on logical interfaces, not directly on physical interfaces.

Example: Setting Up Basic Bridging and a VLAN for an EX Series Switch

IN THIS SECTION

NOTE: This example uses Junos OS for EX Series switches that does not support the Enhanced Layer 2 Software (ELS) configuration style. If your switch runs software that

supports ELS, see "Example: Setting Up Basic Bridging and a VLAN for an EX Series Switch with ELS Support " on page 158. For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15

EX Series switches use bridging and virtual LANs (VLANs) to connect network devices in a LAN—desktop computers, IP telephones, printers, file servers, wireless access points, and others—and to segment the LAN into smaller bridging domains. The switch's default configuration provides a quick setup of bridging and a single VLAN.

This example describes how to configure basic bridging and VLANs for an EX Series switch:

Requirements

This example uses the following software and hardware components:

- Junos OS Release 9.0 or later for EX Series switches
- One EX4200 Virtual Chassis switch

Before you set up bridging and a VLAN, be sure you have:

- Performed the initial switch configuration. See *Connecting and Configuring an EX Series Switch (J-Web Procedure)*.

Overview and Topology

IN THIS SECTION

EX Series switches connect network devices in an office LAN or a data center LAN to provide sharing of common resources such as printers and file servers and to enable wireless devices to connect to the LAN through wireless access points. Without bridging and VLANs, all devices on the Ethernet LAN are in a single broadcast domain, and all the devices detect all the packets on the LAN. Bridging creates separate broadcast domains on the LAN, creating VLANs, which are independent logical networks that group together related devices into separate network segments. The grouping of devices on a VLAN is independent of where the devices are physically located in the LAN.

To use an EX Series switch to connect network devices on a LAN, you must, at a minimum, configure bridging and VLANs. If you simply power on the switch and perform the initial switch configuration using the factory-default settings, bridging is enabled on all the switch's interfaces, all interfaces are in access mode, and all interfaces belong to a VLAN called **default**, which is automatically configured.

When you plug access devices—such as desktop computers, Avaya IP telephones, file servers, printers, and wireless access points—into the switch, they are joined immediately into the **default** VLAN and the LAN is up and running.

The topology used in this example consists of one EX4200-24T switch, which has a total of 24 ports. Eight of the ports support Power over Ethernet (PoE), which means they provide both network connectivity and electric power for the device connecting to the port. To these ports, you can plug in devices requiring PoE, such as Avaya VoIP telephones, wireless access points, and some IP cameras. (Avaya phones have a built-in hub that allows you to connect a desktop PC to the phone, so the desktop and phone in a single office require only one port on the switch.) The remaining 16 ports provide only network connectivity. You use them to connect devices that have their own power sources, such as desktop and laptop computers, printers, and servers. Table 50 on page 197 details the topology used in this configuration example.

Table 50: Components of the Basic Bridging Configuration Topology

Property	Settings
Switch hardware	EX4200-24T switch, with 24 Gigabit Ethernet ports: 8 PoE ports (**ge-0/0/0** through **ge-0/0/7**) and 16 non-PoE ports (**ge-0/0/8** through **ge-0/0/23**)
VLAN name	**default**
Connection to wireless access point (requires PoE)	**ge-0/0/0**
Connections to Avaya IP telephone—with integrated hub, to connect phone and desktop PC to a single port (requires PoE)	**ge-0/0/1** through **ge-0/0/7**
Direct connections to desktop PCs (no PoE required)	**ge-0/0/8** through **ge-0/0/12**
Connections to file servers (no PoE required)	**ge-0/0/17** and **ge-0/0/18**
Connections to integrated printer/fax/copier machines (no PoE required)	**ge-0/0/19** through **ge-0/0/20**
Unused ports (for future expansion)	**ge-0/0/13** through **ge-0/0/16**, and **ge-0/0/21** through **ge-0/0/23**

Topology

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

By default, after you perform the initial configuration on the EX4200 switch, switching is enabled on all interfaces, a VLAN named **default** is created, and all interfaces are placed into this VLAN. You do not need to perform any other configuration on the switch to set up bridging and VLANs. To use the switch, simply plug the Avaya IP phones into the PoE-enabled ports **ge-0/0/1** through **ge-0/0/7**, and plug in the PCs, file servers, and printers to the non-PoE ports, **ge-0/0/8** through **ge-0/0/12** and **ge-0/0/17** through **ge-0/0/20**.

Step-by-Step Procedure

To configure bridging and VLANs:

1. Make sure the switch is powered on.
2. Connect the wireless access point to switch port **ge-0/0/0**.
3. Connect the seven Avaya phones to switch ports **ge-0/0/1** through **ge-0/0/7**.
4. Connect the five PCs to ports **ge-0/0/8** through **ge-0/0/12**.
5. Connect the two file servers to ports **ge-0/0/17** and **ge-0/0/18**.
6. Connect the two printers to ports **ge-0/0/19** and **ge-0/0/20**.

Results

Check the results of the configuration:

```
user@switch> show configuration
## Last commit: 2008-03-06 00:11:22 UTC by triumph
version 9.0;
system {
    root-authentication {
        encrypted-password "$1$urmA7AFM$x5SaGEUOdSI3u1K/iITGh1"; ## SECRET-DATA
    }
    syslog {
        user * {
            any emergency;
        }
        file messages {
            any notice;
            authorization info;
        }
        file interactive-commands {
            interactive-commands any;
        }
    }
    commit {
        factory-settings {
            reset-chassis-lcd-menu;
            reset-virtual-chassis-configuration;
        }
    }
}
interfaces {
    ge-0/0/0 {
        unit 0 {
            family ethernet-switching;
        }
    }
    ge-0/0/1 {
        unit 0 {
            family ethernet-switching;
        }
    }
    ge-0/0/2 {
```

```
        unit 0 {
            family ethernet-switching;
        }
    }
    ge-0/0/3 {
        unit 0 {
            family ethernet-switching;
        }
    }
    ge-0/0/4 {
        unit 0 {
            family ethernet-switching;
        }
    }
    ge-0/0/5 {
        unit 0 {
            family ethernet-switching;
        }
    }
    ge-0/0/6 {
        unit 0 {
            family ethernet-switching;
        }
    }
    ge-0/0/7 {
        unit 0 {
            family ethernet-switching;
        }
    }
    ge-0/0/8 {
        unit 0 {
            family ethernet-switching;
        }
    }
    ge-0/0/9 {
        unit 0 {
            family ethernet-switching;
        }
    }
    ge-0/0/10 {
        unit 0 {
            family ethernet-switching;
        }
```

```
    }
    ge-0/0/11 {
        unit 0 {
            family ethernet-switching;
        }
    }
    ge-0/0/12 {
        unit 0 {
            family ethernet-switching;
        }
    }
    ge-0/0/13 {
        unit 0 {
            family ethernet-switching;
        }
    }
    ge-0/0/14 {
        unit 0 {
            family ethernet-switching;
        }
    }
    ge-0/0/15 {
        unit 0 {
            family ethernet-switching;
        }
    }
    ge-0/0/16 {
        unit 0 {
            family ethernet-switching;
        }
    }
    ge-0/0/17 {
        unit 0 {
            family ethernet-switching;
        }
    }
    ge-0/0/18 {
        unit 0 {
            family ethernet-switching;
        }
    }
    ge-0/0/19 {
        unit 0 {
```

```
            family ethernet-switching;
        }
    }
    ge-0/0/20 {
        unit 0 {
            family ethernet-switching;
        }
    }
    ge-0/0/21 {
        unit 0 {
            family ethernet-switching;
        }
    }
    ge-0/0/22 {
        unit 0 {
            family ethernet-switching;
        }
    }
    ge-0/0/23 {
        unit 0 {
            family ethernet-switching;
        }
    }
    ge-0/1/0 {
        unit 0 {
            family ethernet-switching;
        }
    }
    xe-0/1/0 {
        unit 0 {
            family ethernet-switching;
        }
    }
    ge-0/1/1 {
        unit 0 {
            family ethernet-switching;
        }
    }
    xe-0/1/1 {
        unit 0 {
            family ethernet-switching;
        }
    }
```

```
    ge-0/1/2 {
        unit 0 {
            family ethernet-switching;
        }
    }
    ge-0/1/3 {
        unit 0 {
            family ethernet-switching;
        }
    }
}
protocols {
    lldp {
        interface all;
    }
    rstp;
    poe {
        interface all;
    }
}
```

Verification

IN THIS SECTION

To verify that switching is operational and that a VLAN has been created, perform these tasks:

Verifying That the VLAN Has Been Created

Purpose

Verify that the VLAN named **default** has been created on the switch.

Action

List all VLANs configured on the switch:

```
user@switch>                            show
vlans

Name            Tag     Interfaces
default
                        ge-0/0/0.0*, ge-0/0/1.0, ge-0/0/2.0, ge-0/0/3.0,
                        ge-0/0/4.0,  ge-0/0/5.0, ge-0/0/6.0, ge-0/0/7.0,
                        ge-0/0/8.0*, ge-0/0/9.0, ge-0/0/10.0, ge-0/0/11.0*,
                        ge-0/0/12.0, ge-0/0/13.0, ge-0/0/14.0, ge-0/0/15.0,
                        ge-0/0/16.0, ge-0/0/17.0, ge-0/0/18.0, ge-0/0/19.0*,
                        ge-0/0/20.0, ge-0/0/21.0, ge-0/0/22.0, ge-0/0/23.0,
                        ge-0/1/0.0*, ge-0/1/1.0*, ge-0/1/2.0*, ge-0/1/3.0*
mgmt
                        me0.0*
```

Meaning

The `show vlans` command lists the VLANs configured on the switch. This output shows that the VLAN **default** has been created.

Verifying That Interfaces Are Associated with the Proper VLANs

Purpose

Verify that Ethernet switching is enabled on switch interfaces and that all interfaces are included in the VLAN.

Action

List all interfaces on which switching is enabled:

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

Interface   State    VLAN members           Blocking
ge-0/0/0.0  up       default                unblocked
```

```
ge-0/0/1.0  down     default              blocked - blocked by STP/RTG
ge-0/0/2.0  down     default              blocked - blocked by STP/RTG
ge-0/0/3.0  down     default              blocked - blocked by STP/RTG
ge-0/0/4.0  down     default              blocked - blocked by STP/RTG
ge-0/0/5.0  down     default              blocked - blocked by STP/RTG
ge-0/0/6.0  down     default              blocked - blocked by STP/RTG
ge-0/0/7.0  down     default              blocked - blocked by STP/RTG
ge-0/0/8.0  up       default              unblocked
ge-0/0/9.0  down     default              blocked - blocked by STP/RTG
ge-0/0/10.0 down     default              blocked - blocked by STP/RTG
ge-0/0/11.0 up       default              unblocked
ge-0/0/12.0 down     default              blocked - blocked by STP/RTG
ge-0/0/13.0 down     default              blocked - blocked by STP/RTG
ge-0/0/14.0 down     default              blocked - blocked by STP/RTG
ge-0/0/15.0 down     default              blocked - blocked by STP/RTG
ge-0/0/16.0 down     default              blocked - blocked by STP/RTG
ge-0/0/17.0 down     default              blocked - blocked by STP/RTG
ge-0/0/18.0 down     default              blocked - blocked by STP/RTG
ge-0/0/19.0 up       default              unblocked
ge-0/0/20.0 down     default              blocked - blocked by STP/RTG
ge-0/0/21.0 down     default              blocked - blocked by STP/RTG
ge-0/0/22.0 down     default              blocked - blocked by STP/RTG
ge-0/0/23.0 down     default              blocked - blocked by STP/RTG
ge-0/1/0.0  up       default              unblocked
ge-0/1/1.0  up       default              unblocked
ge-0/1/2.0  up       default              unblocked
ge-0/1/3.0  up       default              unblocked
me0.0       up       mgmt                 unblocked
```

Meaning

The `show ethernet-switching interfaces` command lists all interfaces on which switching is enabled (in the **Interfaces** column), along with the VLANs that are active on the interfaces (in the **VLAN members** column). The output in this example shows all the connected interfaces, **ge-0/0/0** through **ge-0/0/12** and **ge-0/0/17** through **ge-0/0/20** and that they are all part of VLAN **default**. Notice that the interfaces listed are the logical interfaces, not the physical interfaces. For example, the output shows **ge-0/0/0.0** instead of **ge-0/0/0**. This is because Junos OS creates VLANs on logical interfaces, not directly on physical interfaces.

Example: Setting Up Bridging with Multiple VLANs

IN THIS SECTION

The QFX Series products use bridging and virtual LANs (VLANs) to connect network devices in a LAN—storage devices, file servers, and other network components—and to segment the LAN into smaller bridging domains.

To segment traffic on a LAN into separate broadcast domains, you create separate virtual LANs (VLANs) on a switch. Each VLAN is a collection of network nodes. When you use VLANs, frames whose origin and destination are in the same VLAN are forwarded only within the local VLAN, and only frames not destined for the local VLAN are forwarded to other broadcast domains. VLANs thus limit the amount of traffic flowing across the entire LAN, reducing the possible number of collisions and packet retransmissions within the LAN.

NOTE: This task uses Junos OS for QFX3500 and QFX3600 switches does not support the Enhanced Layer 2 Software (ELS) configuration style. If your switch runs software that supports ELS, see "Example: Setting Up Bridging with Multiple VLANs on Switches" on page 215.

This example describes how to configure bridging for the QFX Series and how to create two VLANs to segment the LAN:

Requirements

This example uses the following hardware and software components:

- A configured and provisioned QFX3500 switch
- Junos OS Release 11.1 or later for the QFX Series

Overview and Topology

IN THIS SECTION

Switches connect all devices in an office or data center into a single LAN to provide sharing of common resources such as file servers. The default configuration creates a single VLAN, and all traffic on the switch is part of that broadcast domain. Creating separate network segments reduces the span of the broadcast domain and enables you to group related users and network resources without being limited by physical cabling or by the location of a network device in the building or on the LAN.

This example shows a simple configuration to illustrate the basic steps for creating two VLANs on a single switch. One VLAN, called **sales**, is for the sales and marketing group, and a second, called **support**, is for the customer support team. The sales and support groups each have their own dedicated file servers and other resources. For the switch ports to be segmented across the two VLANs, each VLAN must have its own broadcast domain, identified by a unique name and tag (VLAN ID). In addition, each VLAN must be on its own distinct IP subnet.

Topology

The topology used in this example consists of a single QFX3500 switch, with a total of 48 10-Gbps Ethernet ports. (For the purposes of this example, the QSFP+ ports Q0-Q3, which are ports xe-0/1/0 through xe-0/1/15, are excluded.)

Table 51: Components of the Multiple VLAN Topology

Property	Settings
Switch hardware	QFX3500 switch configured with 48 10-Gbps Ethernet ports (**xe-0/0/0** through **xe-0/0/47**)
VLAN names and tag IDs	**sales**, tag **100** **support**, tag **200**

Table 51: Components of the Multiple VLAN Topology *(Continued)*

Property	Settings
VLAN subnets	**sales**: **192.0.2.0/25** (addresses **192.0.2.1** through **192.0.2.126**) **support**: **192.0.2.128/25** (addresses **192.0.2.129** through **192.0.2.254**)
Interfaces in VLAN **sales**	File servers: **xe-0/0/20** and **xe-0/0/21**
Interfaces in VLAN **support**	File servers: **xe-0/0/46** and **xe-0/0/47**
Unused interfaces	**xe-0/0/2** and **xe-0/0/25**

This configuration example creates two IP subnets, one for the sales VLAN and the second for the support VLAN. The switch bridges traffic within a VLAN. For traffic passing between two VLANs, the switch routes the traffic using a Layer 3 routing interface on which you have configured the address of the IP subnet.

To keep the example simple, the configuration steps show only a few devices in each of the VLANs. Use the same configuration procedure to add more LAN devices.

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

To quickly configure Layer 2 switching for the two VLANs (**sales** and **support**) and to quickly configure Layer 3 routing of traffic between the two VLANs, copy the following commands and paste them into the switch terminal window:

```
[edit]
                        set interfaces xe-0/0/0 unit 0 family ethernet-switching vlan members
sales
                        set interfaces xe-0/0/3 unit 0 family ethernet-switching vlan members
sales
                        set interfaces xe-0/0/22 unit 0 family ethernet-switching vlan members
sales
                        set interfaces xe-0/0/20 unit 0 description “Sales file server
port”
                        set interfaces xe-0/0/20 unit 0 family ethernet-switching vlan members
sales
                        set interfaces xe-0/0/24 unit 0 family ethernet-switching vlan members
support
                        set interfaces xe-0/0/26 unit 0 family ethernet-switching vlan members
support
                        set interfaces xe-0/0/44 unit 0 family ethernet-switching vlan members
support
                        set interfaces xe-0/0/46  unit 0 description “Support file server
port”
                        set interfaces xe-0/0/46 unit 0 family ethernet-switching vlan members
support
                        set interfaces irb unit 0 family inet address
192.0.2.0/25
                        set interfaces irb unit 1 family inet address
192.0.2.128/25
                        set vlans sales l3-interface irb.0
                        set vlans sales vlan-id 100
                        set vlans support vlan-id 200
                        set vlans support l3-interface irb.1
```

Step-by-Step Procedure

Configure the switch interfaces and the VLANs to which they belong. By default, all interfaces are in access mode, so you do not have to configure the port mode.

1. Configure the interface for the file server in the **sales** VLAN:

```
[edit interfaces xe-0/0/20 unit 0]
user@switch# set description “Sales file server port”
user@switch# set family ethernet-switching vlan members sales
```

2. Configure the interface for the file server in the **support** VLAN:

```
[edit interfaces xe-0/0/46 unit 0]
user@switch# set description “Support file server port”
user@switch# set family ethernet-switching vlan members support
```

3. Create the subnet for the **sales** broadcast domain:

```
[edit interfaces]
user@switch# set irb unit 0 family inet address 192.0.2.1/25
```

4. Create the subnet for the **support** broadcast domain:

```
[edit interfaces]
user@switch# set irb unit 1 family inet address 192.0.2.129/25
```

5. Configure the VLAN tag IDs for the **sales** and **support** VLANs:

```
[edit vlans]
user@switch# set sales vlan-id 100
user@switch# set support vlan-id 200
```

6. To route traffic between the **sales** and **support** VLANs, define the interfaces that are members of each VLAN and associate a Layer 3 interface:

```
[edit vlans]
user@switch# set sales l3-interface irb.0
user@switch# set support l3-interface irb.1
```

Results

Display the results of the configuration:

```
user@switch> show configuration
interfaces {
    xe-0/0/20 {
        unit 0 {
            description “Sales file server port”;
            family ethernet-switching {
                vlan members sales;
            }
        }
    }
    xe-0/0/46 {
        unit 0 {
            description “Support file server port”;
            family ethernet-switching {
                vlan members support;
            }
        }
    }
    irb {
            unit 0 {
                family inet address 192.0.2.1/25;
            }
            unit 1 {
                family inet address 192.0.2.129/25;
            }
    }
    vlans {
        sales {
            vlan-id 100;
            interface xe-0/0/0.0:
```

```
            interface xe-0/0/3/0;
            interface xe-0/0/20.0;
            interface xe-0/0/22.0;
            l3-interface irb.0;
             }
            support {
                    vlan-id 200;
                    interface xe-0/0/24.0:
                    interface xe-0/0/26.0;
                    interface xe-0/0/44.0;
                    interface xe-0/0/46.0;
                    l3-interface irb.1;
            }
     }
}
```

TIP: To quickly configure the sales and support VLAN interfaces, issue the `load merge terminal` command. Then copy the hierarchy and paste it into the switch terminal window.

Verification

IN THIS SECTION

Verify that the **sales** and **support** VLANs have been created and are operating properly, perform these tasks:

Verifying That the VLANs Have Been Created and Associated with the Correct Interfaces

Purpose

Verify that the **sales** and **support** VLANs have been created on the switch and that all connected interfaces on the switch are members of the correct VLAN.

Action

To list all VLANs configured on the switch, use the `show vlans` command:

```
user@switch> show vlans
Name            Tag     Interfaces
default
                        xe-0/0/1.0, xe-0/0/2.0, xe-0/0/4.0, xe-0/0/5.0,
                        xe-0/0/6.0,  xe-0/0/7.0, xe-0/0/8.0, xe-0/0/9.0,
                        xe-0/0/10.0*, xe-0/0/11.0, xe-0/0/12.0, xe-0/0/13.0*,
                        xe-0/0/14.0, xe-0/0/15.0, xe-0/0/16.0, xe-0/0/17.0,
                        xe-0/0/18.0, xe-0/0/19.0, xe-0/0/21.0, xe-0/0/23.0*,
                        xe-0/0/25.0, xe-0/0/27.0, xe-0/0/28.0, xe-0/0/29.0,
                        xe-0/0/30.0, xe-0/0/31.0, xe-0/0/32.0, xe-0/0/33.0,
                        xe-0/0/34.0, xe-0/0/35.0, xe-0/0/36.0, xe-0/0/37.0,
                        xe-0/0/38.0, xe-0/0/39.0, xe-0/0/40.0, xe-0/0/41.0,
                        xe-0/0/42.0, xe-0/0/43.0, xe-0/0/45.0, xe-0/0/47.0,
                        xe-0/1/0.0*, xe-0/1/1.0*, xe-0/1/2.0*, xe-0/1/3.0*

sales       100
                        xe-0/0/0.0*, xe-0/0/3.0, xe-0/0/20.0, xe-0/0/22.0

support     200
                        xe-0/0/0.24, xe-0/0/26.0, xe-0/0/44.0, xe-0/0/46.0*
mgmt
                        me0.0*
```

Meaning

The `show vlans` command lists all VLANs configured on the switch and which interfaces are members of each VLAN. This command output shows that the **sales** and **support** VLANs have been created. The **sales** VLAN has a tag ID of 100 and is associated with interfaces **xe-0/0/0.0**, **xe-0/0/3.0**, **xe-0/0/20.0**, and **xe-0/0/22.0**. VLAN **support** has a tag ID of 200 and is associated with interfaces **xe-0/0/24.0**, **xe-0/0/26.0**, **xe-0/0/44.0**, and **xe-0/0/46.0**.

Verifying That Traffic Is Being Routed Between the Two VLANs

Purpose

Verify routing between the two VLANs.

Action

List the Layer 3 routes in the switch Address Resolution Protocol (ARP) table:

```
user@switch> show arp
MAC Address       Address         Name                    Flags

00:00:0c:06:2c:0d        192.0.2.3       irb.0                   None
00:13:e2:50:62:e0        192.0.2.11      irb.1                   None
```

Meaning

Sending IP packets on a multiaccess network requires mapping from an IP address to a MAC address (the physical or hardware address). The ARP table displays the mapping between the IP address and MAC address for both **irb.0** (associated with **sales**) and **irb.1** (associated with **support**). These VLANs can route traffic to each other.

Verifying That Traffic Is Being Switched Between the Two VLANs

Purpose

Verify that learned entries are being added to the Ethernet switching table.

Action

List the contents of the Ethernet switching table:

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

Ethernet-switching table: 8 entries, 5 learned
  VLAN              MAC address       Type         Age Interfaces
  default           *                 Flood          - All-members
  default           00:00:05:00:00:01 Learn          - xe-0/0/10.0
  default           00:00:5e:00:01:09 Learn          - xe-0/0/13.0
  default           00:19:e2:50:63:e0 Learn          - xe-0/0/23.0
  sales             *                 Flood          - All-members
  sales             00:00:5e:00:07:09 Learn          - xe-0/0/0.0
  support           *                 Flood          - All-members
```

```
support           00:00:5e:00:01:01 Learn          - xe-0/0/46.0
```

Meaning

The output shows that learned entries for the **sales** and **support** VLANs have been added to the Ethernet switching table, and are associated with interfaces **xe-0/0/0.0** and **xe-0/0/46.0**. Even though the VLANs were associated with more than one interface in the configuration, these interfaces are the only ones that are currently operating.

Example: Setting Up Bridging with Multiple VLANs on Switches

IN THIS SECTION

The QFX Series products use bridging and virtual LANs (VLANs) to connect network devices in a LAN—storage devices, file servers, and other network components—and to segment the LAN into smaller bridging domains.

To segment traffic on a LAN into separate broadcast domains, you create separate virtual LANs (VLANs) on a switch. Each VLAN is a collection of network nodes. When you use VLANs, frames whose origin and destination are in the same VLAN are forwarded only within the local VLAN, and only frames not destined for the local VLAN are forwarded to other broadcast domains. VLANs thus limit the amount of traffic flowing across the entire LAN, reducing the possible number of collisions and packet retransmissions within the LAN.

This example describes how to configure bridging for the QFX Series and how to create two VLANs to segment the LAN:

NOTE: This task supports the Enhanced Layer 2 Software (ELS) configuration style. For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15. If your switch

runs software that does not supports ELS, see "Example: Setting Up Bridging with Multiple VLANs" on page 206.

Requirements

This example uses the following hardware and software components:

- A configured and provisioned QFX3500 switch
- Junos OS Release 13.2X50-D15 or later for the QFX Series

Overview and Topology

IN THIS SECTION

Switches connect all devices in an office or data center into a single LAN to provide sharing of common resources such as file servers. The default configuration creates a single VLAN, and all traffic on the switch is part of that broadcast domain. Creating separate network segments reduces the span of the broadcast domain and enables you to group related users and network resources without being limited by physical cabling or by the location of a network device in the building or on the LAN.

This example shows a simple configuration to illustrate the basic steps for creating two VLANs on a single switch. One VLAN, called **sales**, is for the sales and marketing group, and a second, called **support**, is for the customer support team. The sales and support groups each have their own dedicated file servers and other resources. For the switch ports to be segmented across the two VLANs, each VLAN must have its own broadcast domain, identified by a unique name and tag (VLAN ID). In addition, each VLAN must be on its own distinct IP subnet.

Topology

The topology used in this example consists of a single QFX3500 switch, with a total of 48 10-Gbps Ethernet ports. (For the purposes of this example, the QSFP+ ports Q0-Q3, which are ports xe-0/1/0 through xe-0/1/15, are excluded.)

Table 52: Components of the Multiple VLAN Topology

Property	Settings
Switch hardware	QFX3500 switch configured with 48 10-Gbps Ethernet ports (**xe-0/0/0** through **xe-0/0/47**)
VLAN names and tag IDs	**sales**, tag **100** **support**, tag **200**
VLAN subnets	**sales**: **192.0.2.0/25** (addresses **192.0.2.1** through **192.0.2.126**) **support**: **192.0.2.128/25** (addresses **192.0.2.129** through **192.0.2.254**)
Interfaces in VLAN **sales**	File servers: **xe-0/0/20** and **xe-0/0/21**
Interfaces in VLAN **support**	File servers: **xe-0/0/46** and **xe-0/0/47**
Unused interfaces	**xe-0/0/2** and **xe-0/0/25**

This configuration example creates two IP subnets, one for the sales VLAN and the second for the support VLAN. The switch bridges traffic within a VLAN. For traffic passing between two VLANs, the switch routes the traffic using a Layer 3 routing interface on which you have configured the address of the IP subnet.

To keep the example simple, the configuration steps show only a few devices in each of the VLANs. Use the same configuration procedure to add more LAN devices.

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

To quickly configure Layer 2 switching for the two VLANs (**sales** and **support**) and to quickly configure Layer 3 routing of traffic between the two VLANs, copy the following commands and paste them into the switch terminal window:

```
[edit]
                        set interfaces xe-0/0/0 unit 0 family ethernet-switching vlan members
sales
                        set interfaces xe-0/0/3 unit 0 family ethernet-switching vlan members
sales
                        set interfaces xe-0/0/22 unit 0 family ethernet-switching vlan members
sales
                        set interfaces xe-0/0/20 unit 0 description “Sales file server
port”
                        set interfaces xe-0/0/20 unit 0 family ethernet-switching vlan members
sales
                        set interfaces xe-0/0/24 unit 0 family ethernet-switching vlan members
support
                        set interfaces xe-0/0/26 unit 0 family ethernet-switching vlan members
support
                        set interfaces xe-0/0/44 unit 0 family ethernet-switching vlan members
support
                        set interfaces xe-0/0/46 unit 0 description “Support file server
port”
                        set interfaces xe-0/0/46 unit 0 family ethernet-switching vlan members
support
                        set interfaces irb unit 0 family inet address
192.0.2.0/25
                        set interfaces irb unit 1 family inet address
192.0.2.128/25
                        set vlans sales l3-interface irb.0
                        set vlans sales vlan-id 100
                        set vlans support vlan-id 200
                        set vlans support l3-interface irb.1
```

Step-by-Step Procedure

Configure the switch interfaces and the VLANs to which they belong. By default, all interfaces are in access mode, so you do not have to configure the port mode.

1. Configure the interface for the file server in the **sales** VLAN:

```
[edit interfaces xe-0/0/20 unit 0]
user@switch# set description “Sales file server port”
user@switch# set family ethernet-switching vlan members sales
```

2. Configure the interface for the file server in the **support** VLAN:

```
[edit interfaces xe-0/0/46 unit 0]
user@switch# set description “Support file server port”
user@switch# set family ethernet-switching vlan members support
```

3. Create the subnet for the **sales** broadcast domain:

```
[edit interfaces]
user@switch# set irb unit 0 family inet address 192.0.2.1/25
```

4. Create the subnet for the **support** broadcast domain:

```
[edit interfaces]
user@switch# set irb unit 1 family inet address 192.0.2.129/25
```

5. Configure the VLAN tag IDs for the **sales** and **support** VLANs:

```
[edit vlans]
user@switch# set sales vlan-id 100
user@switch# set support vlan-id 200
```

6. To route traffic between the **sales** and **support** VLANs, define the interfaces that are members of each VLAN and associate a Layer 3 interface:

```
[edit vlans]
user@switch# set sales l3-interface irb.0
user@switch# set support l3-interface irb.1
```

Configuration Results

Display the results of the configuration:

```
user@switch> show configuration
interfaces {
    xe-0/0/20 {
        unit 0 {
            description “Sales file server port”;
            family ethernet-switching {
                vlan members sales;
            }
        }
    }
    xe-0/0/46 {
        unit 0 {
            description “Support file server port”;
            family ethernet-switching {
                vlan members support;
            }
        }
    }
    irb {
        unit 0 {
            family inet address 192.0.2.1/25;
        }
        unit 1 {
            family inet address 192.0.2.129/25;
        }
    }
}
vlans {
        sales {
                vlan-id 100;
```

```
                interface xe-0/0/0.0:
                interface xe-0/0/3/0;
                interface xe-0/0/20.0;
                interface xe-0/0/22.0;
                l3-interface irb.0;
        }
        support {
                vlan-id 200;
                interface xe-0/0/24.0:
                interface xe-0/0/26.0;
                interface xe-0/0/44.0;
                interface xe-0/0/46.0;
                l3-interface irb.1;
        }
}
```

TIP: To quickly configure the sales and support VLAN interfaces, issue the `load merge terminal` command. Then copy the hierarchy and paste it into the switch terminal window.

Verification

IN THIS SECTION

Verify that the **sales** and **support** VLANs have been created and are operating properly, perform these tasks:

Verifying That the VLANs Have Been Created and Associated with the Correct Interfaces

Purpose

Verify that the **sales** and **support** VLANs have been created on the switch and that all connected interfaces on the switch are members of the correct VLAN.

Action

To list all VLANs configured on the switch, use the `show vlans` command:

```
user@switch> show vlans
Name            Tag     Interfaces
default
                        xe-0/0/1.0, xe-0/0/2.0, xe-0/0/4.0, xe-0/0/5.0,
                        xe-0/0/6.0,  xe-0/0/7.0, xe-0/0/8.0, xe-0/0/9.0,
                        xe-0/0/10.0*, xe-0/0/11.0, xe-0/0/12.0, xe-0/0/13.0*,
                        xe-0/0/14.0, xe-0/0/15.0, xe-0/0/16.0, xe-0/0/17.0,
                        xe-0/0/18.0, xe-0/0/19.0, xe-0/0/21.0, xe-0/0/23.0*,
                        xe-0/0/25.0, xe-0/0/27.0, xe-0/0/28.0, xe-0/0/29.0,
                        xe-0/0/30.0, xe-0/0/31.0, xe-0/0/32.0, xe-0/0/33.0,
                        xe-0/0/34.0, xe-0/0/35.0, xe-0/0/36.0, xe-0/0/37.0,
                        xe-0/0/38.0, xe-0/0/39.0, xe-0/0/40.0, xe-0/0/41.0,
                        xe-0/0/42.0, xe-0/0/43.0, xe-0/0/45.0, xe-0/0/47.0,
                        xe-0/1/0.0*, xe-0/1/1.0*, xe-0/1/2.0*, xe-0/1/3.0*

sales           100
                        xe-0/0/0.0*, xe-0/0/3.0, xe-0/0/20.0, xe-0/0/22.0

support         200
                        xe-0/0/0.24, xe-0/0/26.0, xe-0/0/44.0, xe-0/0/46.0*
mgmt
                        me0.0*
```

Meaning

The `show vlans` command lists all VLANs configured on the switch and which interfaces are members of each VLAN. This command output shows that the **sales** and **support** VLANs have been created. The **sales** VLAN has a tag ID of 100 and is associated with interfaces **xe-0/0/0.0**, **xe-0/0/3.0**, **xe-0/0/20.0**, and **xe-0/0/22.0**. VLAN **support** has a tag ID of 200 and is associated with interfaces **xe-0/0/24.0**, **xe-0/0/26.0**, **xe-0/0/44.0**, and **xe-0/0/46.0**.

Verifying That Traffic Is Being Routed Between the Two VLANs

Purpose

Verify routing between the two VLANs.

Action

List the Layer 3 routes in the switch Address Resolution Protocol (ARP) table:

```
user@switch> show arp
MAC Address       Address         Name                    Flags

00:00:0c:06:2c:0d        192.0.2.3       irb.0                   None
00:13:e2:50:62:e0        192.0.2.11      irb.1                   None
```

Meaning

Sending IP packets on a multiaccess network requires mapping from an IP address to a MAC address (the physical or hardware address). The ARP table displays the mapping between the IP address and MAC address for both **irb.0** (associated with **sales**) and **irb.1** (associated with **support**). These VLANs can route traffic to each other.

Verifying That Traffic Is Being Switched Between the Two VLANs

Purpose

Verify that learned entries are being added to the Ethernet switching table.

Action

List the contents of the Ethernet switching table:

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

Ethernet-switching table: 8 entries, 5 learned
  VLAN              MAC address       Type         Age Interfaces
  default           *                 Flood          - All-members
  default           00:00:05:00:00:01 Learn          - xe-0/0/10.0
  default           00:00:5e:00:01:09 Learn          - xe-0/0/13.0
  default           00:19:e2:50:63:e0 Learn          - xe-0/0/23.0
  sales             *                 Flood          - All-members
  sales             00:00:5e:00:07:09 Learn          - xe-0/0/0.0
  support           *                 Flood          - All-members
```

```
support           00:00:5e:00:01:01 Learn          - xe-0/0/46.0
```

Meaning

The output shows that learned entries for the **sales** and **support** VLANs have been added to the Ethernet switching table, and are associated with interfaces **xe-0/0/0.0** and **xe-0/0/46.0**. Even though the VLANs were associated with more than one interface in the configuration, these interfaces are the only ones that are currently operating.

Example: Connecting Access Switches with ELS Support to a Distribution Switch with ELS Support

IN THIS SECTION

NOTE: This example uses Junos OS for EX Series switches with support for the Enhanced Layer 2 Software (ELS) configuration style. For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15.

In large local area networks (LANs), you commonly need to aggregate traffic from a number of access switches into a distribution switch.

This example describes how to connect access switches to a distribution switch:

Requirements

This example uses the following hardware and software components:

- Three EX Series access switches.

- One EX Series distribution switch.

  NOTE: In an access switch-distribution switch topology, you can connect EX Series switches that run a version of Junos OS that supports ELS with EX Series switches that do not run a version of Junos OS that supports ELS. However, this example uses switches running ELS only to show how to configure this topology using the ELS CLI.

- Junos OS Release 12.3R2 or later that supports ELS for EX Series switches.

Before you connect an access switch to a distribution switch, be sure you have:

- Installed the switches. See the installation instructions for your switch.
- Performed the initial software configuration on both switches. For information about the initial software configuration for all EX Series switches except the EX9200 Series switches, see *Connecting and Configuring an EX Series Switch (CLI Procedure)*. For information about the initial software configuration for the EX9200 Series switches, see *Connecting and Configuring an EX9200 Switch (CLI Procedure)*.

Overview and Topology

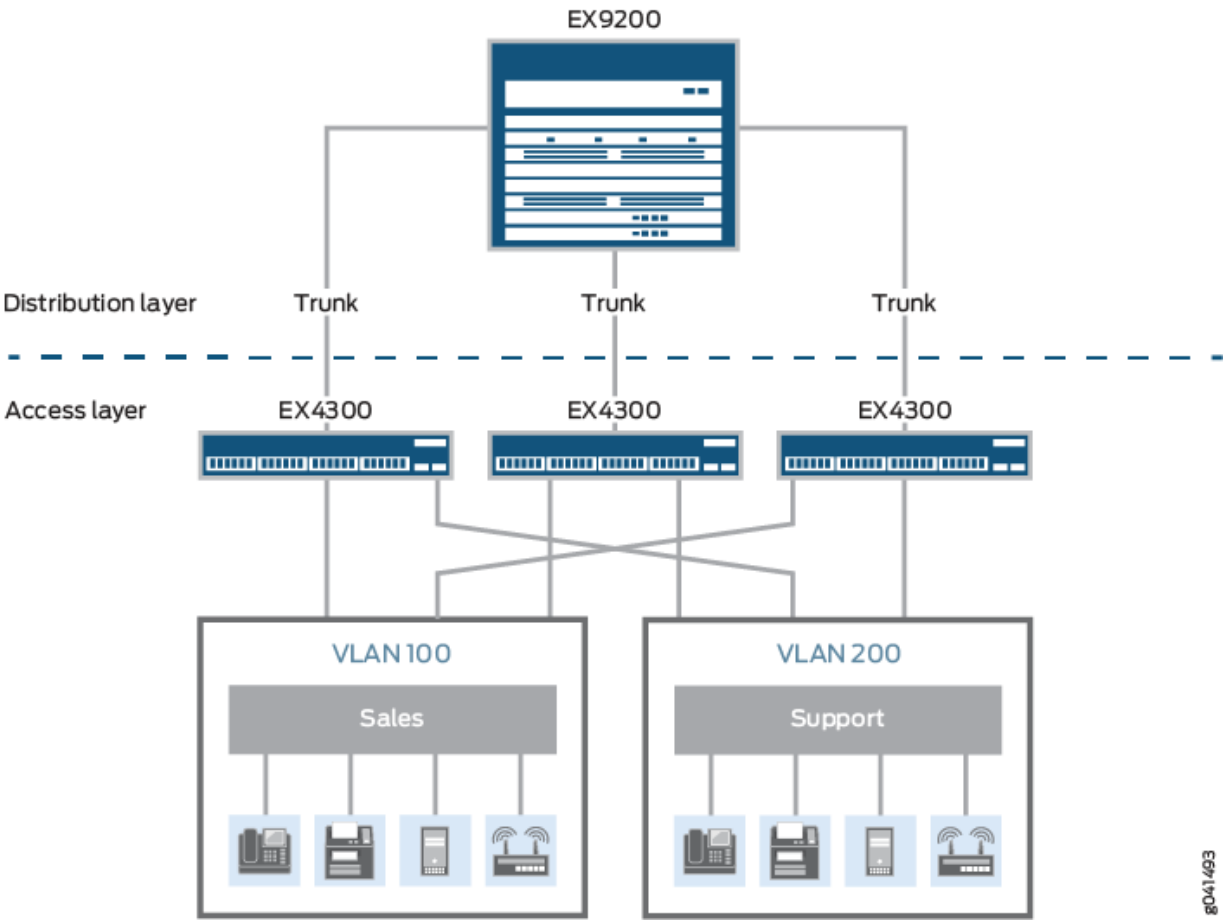
IN THIS SECTION

In a large office that is spread across several floors or buildings, or in a data center, you commonly aggregate traffic from a number of access switches into a distribution switch. This configuration example shows a simple topology to illustrate how to connect three access switches to a distribution switch.

In the topology, the LAN is segmented into two VLANs, one for the sales department and the second for the support team. One 1-Gigabit Ethernet port on one of the access switch's uplink modules connects to the distribution switch, to one 1-Gigabit Ethernet port on the distribution switch.

Figure 1 on page 226 shows an EX9200 distribution switch that is connected to three EX4300 access switches.

Figure 1: Sample Access Switch-Distribution Switch Topology



Topology

Table 53 on page 226 describes the components of the example topology. The example shows how to configure one of the three access switches. The other access switches could be configured in the same manner.

Table 53: Components of the Topology for Connecting an Access Switch to a Distribution Switch

Property	Settings
Access switch hardware	Three EX4300 switches, each with an uplink module with 1-Gigabit Ethernet ports..
Distribution switch hardware	One EX9208 with up to three EX9200-40T line cards installed, which at full duplex, can provide up to 240 1-Gigabit ports.

Table 53: Components of the Topology for Connecting an Access Switch to a Distribution Switch *(Continued)*

Property	Settings
VLAN names and tag IDs	**sales**, tag **100** **support**, tag **200**
VLAN subnets	**sales**: 192.0.2.0/25 (addresses 192.0.2.1 through 192.0.2.126) **support**: 192.0.2.128/25 (addresses 192.0.2.129 through 192.0.2.254)
Trunk port interfaces	On the access switch: ge-0/2/0 On the distribution switch: ge-0/0/0
Access port interfaces in VLAN **sales** (on access switch)	Avaya IP telephones: ge-0/0/3 through ge-0/0/19 Wireless access points: ge-0/0/0 and ge-0/0/1 Printers: ge-0/0/22 and ge-0/0/23 File servers: ge-0/0/20 and ge-0/0/21
Access port interfaces in VLAN **support** (on access switch)	Avaya IP telephones: ge-0/0/25 through ge-0/0/43 Wireless access points: ge-0/0/24 Printers: ge-0/0/44 and ge-0/0/45 File servers: ge-0/0/46 and ge-0/0/47

Configuring the Access Switch

IN THIS SECTION

To configure the access switch:

Procedure

CLI Quick Configuration

To quickly configure the access switch, copy the following commands and paste them into the switch terminal window:

```
[edit]
                        set interfaces ge-0/0/0 unit 0 description "Sales wireless access point
port"
                        set interfaces ge-0/0/0 unit 0 family ethernet-switching vlan members
sales
                        set interfaces ge-0/0/3  unit 0 description "Sales phone
port"
                        set interfaces ge-0/0/3 unit 0 family ethernet-switching vlan members
sales
                        set interfaces ge-0/0/22  unit 0 description "Sales printer
port"
                        set interfaces ge-0/0/22 unit 0 family ethernet-switching vlan members
sales
                        set interfaces ge-0/0/20 unit 0 description "Sales file server
port"
                        set interfaces ge-0/0/20 unit 0 family ethernet-switching vlan members
sales
                        set interfaces ge-0/0/24 unit 0 description "Support wireless access
point port"
                        set interfaces ge-0/0/24 unit 0 family ethernet-switching vlan members
support
                        set interfaces ge-0/0/26  unit 0 description "Support phone
port"
                        set interfaces ge-0/0/26 unit 0 family ethernet-switching vlan members
support
                        set interfaces ge-0/0/44  unit 0 description "Support printer
port"
                        set interfaces ge-0/0/44 unit 0 family ethernet-switching vlan members
support
                        set interfaces ge-0/0/46 unit 0 description "Support file server
port"
                        set interfaces ge-0/0/46 unit 0 family ethernet-switching vlan members
support
                        set interfaces ge-0/2/0 unit 0 description "Uplink module port
connection to distribution switch"
```

```
                        set interfaces ge-0/2/0 unit 0 family ethernet-switching interface-mode
trunk
                        set interfaces ge-0/2/0 native-vlan-id 1
                        set interfaces ge-0/2/0 unit 0 family ethernet-switching vlan members
[sales support]
                        set interfaces ge-0/2/0 unit 0 family ethernet-switching vlan members
1
                        set interfaces irb unit 0 family inet address
192.0.2.1/25
                        set interfaces irb unit 1 family inet address
192.0.2.129/25
                        set vlans sales description “Sales VLAN”
                        set vlans sales l3-interface irb.0
                        set vlans sales vlan-id 100
                        set vlans support description “Support VLAN”
                        set vlans support vlan-id 200
                        set vlans support l3-interface irb.1
```

Step-by-Step Procedure

To configure the access switch:

1. Configure the 1-Gigabit Ethernet interface on the uplink module to be the trunk port that connects to the distribution switch:

```
[edit interfaces]
user@access-switch# set ge-0/2/0 unit 0 description "Uplink module port connection to
distribution switch"
user@access-switch# set ge-0/2/0 unit 0 family ethernet-switching interface-mode
trunk
```

2. Specify the VLANs to be aggregated on the trunk port:

```
[edit interfaces]
user@access-switch# set ge-0/2/0 unit 0 family ethernet-switching vlan members [ sales
support ]
```

3. To handle untagged packets that are received on the trunk port, create a native VLAN by configuring a VLAN ID and specifying that the trunk port is a member of the native VLAN:

```
 [edit interfaces]
user@access-switch# set ge-0/2/0 native-vlan-id 1
user@access-switch# set ge-0/2/0 unit 0 family ethernet-switching vlan members 1
```

4. Configure the sales VLAN:

```
[edit vlans]
user@access-switch# set sales description "Sales VLAN"
user@access-switch# set sales vlan-id 100
user@access-switch# set sales l3-interface irb.0
```

5. Configure the support VLAN:

```
[edit vlans]
user@access-switch# set support description "Support VLAN"
user@access-switch# set support vlan-id 200
user@access-switch# set support l3-interface irb.1
```

6. Create the subnet for the sales VLAN:

```
[edit interfaces]
user@access-switch# set irb unit 0 family inet address 192.0.2.1/25
```

7. Create the subnet for the support VLAN:

```
[edit interfaces]
user@access-switch# set irb unit 1 family inet address 192.0.2.129/25
```

8. Configure the interfaces in the sales VLAN:

```
[edit interfaces]
user@access-switch# set ge-0/0/0 unit 0 description "Sales wireless access point port"
user@access-switch# set ge-0/0/0 unit 0 family ethernet-switching vlan members sales
user@access-switch# set ge-0/0/3 unit 0 description "Sales phone port"
```

```
user@access-switch# set ge-0/0/3 unit 0 family ethernet-switching vlan members sales
user@access-switch# set ge-0/0/20 unit 0 description "Sales file server port"
user@access-switch# set ge-0/0/20 unit 0 family ethernet-switching vlan members sales
user@access-switch# set ge-0/0/22 unit 0 description "Sales printer port"
user@access-switch# set ge-0/0/22 unit 0 family ethernet-switching vlan members sales
```

9. Configure the interfaces in the support VLAN:

```
[edit interfaces]
user@access-switch# set ge-0/0/24 unit 0 description "Support wireless access point port"
user@access-switch# set ge-0/0/24 unit 0 family ethernet-switching vlan members support
user@access-switch# set ge-0/0/26 unit 0 description "Support phone port"
user@access-switch# set ge-0/0/26 unit 0 family ethernet-switching vlan members support
user@access-switch# set ge-0/0/44 unit 0 description "Support printer port"
user@access-switch# set ge-0/0/44 unit 0 family ethernet-switching vlan members support
user@access-switch# set ge-0/0/46 unit 0 description "Support file server port"
user@access-switch# set ge-0/0/46 unit 0 family ethernet-switching vlan members support
```

Results

Display the results of the configuration:

```
user@access-switch> show configuration
interfaces {
    ge-0/0/0 {
        unit 0 {
            description "Sales wireless access point port";
            family ethernet-switching {
                vlan {
                    members sales;
                }
            }
        }
    }
    ge-0/0/3 {
        unit 0 {
            description "Sales phone port";
            family ethernet-switching {
```

```
            vlan {
                members sales;
            }
        }
    }
}
ge-0/0/20 {
    unit 0 {
        description "Sales file server port";
        family ethernet-switching {
            vlan {
                members sales;
            }
        }
    }
}
ge-0/0/22 {
    unit 0 {
        description "Sales printer port";
        family ethernet-switching {
            vlan {
                members sales;
            }
        }
    }
}
ge-0/0/24 {
    unit 0 {
        description "Support wireless access point port";
        family ethernet-switching {
            vlan {
                members support;
            }
        }
    }
}
ge-0/0/26 {
    unit 0 {
        description "Support phone port";
        family ethernet-switching {
            vlan {
                members support;
            }
```

```
        }
    }
}
ge-0/0/44 {
    unit 0 {
        description "Support printer port";
        family ethernet-switching {
            vlan {
                members support;
            }
        }
    }
}
ge-0/0/46 {
    unit 0 {
        description "Support file server port";
        family ethernet-switching {
            vlan {
                members support;
            }
        }
    }
}
ge-0/2/0 {
    native-vlan-id 1;
    unit 0 {
        description "Uplinking module connection to distribution switch";
        family ethernet-switching {
            interface-mode trunk;
            vlan {
                members [ 1 sales support ];
            }
        }
    }
}
irb {
    unit 0 {
        family inet {
            address 192.0.2.1/25;
        }
    }
    unit 1 {
        family inet {
```

```
                address 192.0.2.129/25;
            }
        }
    }
    vlans {
         sales {
            description "Sales VLAN";
            vlan-id 100;
            l3-interface irb.0;
        }
        support {
            description "Support VLAN";
            vlan-id 200;
            l3-interface irb.1;
        }
    }
}
```

TIP: To quickly configure the access switch, issue the `load merge terminal` command, then copy the hierarchy and paste it into the switch terminal window.

Configuring the Distribution Switch

IN THIS SECTION

To configure the distribution switch:

Procedure

CLI Quick Configuration

To quickly configure the distribution switch, copy the following commands and paste them into the switch terminal window:

```
                        set interfaces ge-0/0/0 unit 0 description "Connection to access
switch"
                        set interfaces ge-0/0/0 unit 0 family ethernet-switching interface-mode
trunk
                        set interfaces ge-0/0/0 unit 0 family ethernet-switching vlan members
[ sales support ]
                        set interfaces ge-0/0/0 native-vlan-id 1
                        set interfaces ge-0/0/0 unit 0 family ethernet-switching vlan members
1
                        set interfaces irb unit 0 family inet address
192.0.2.2/25
                        set interfaces irb unit 1 family inet address
192.0.2.130/25
                        set vlans sales description "Sales VLAN"
                        set vlans sales vlan-id 100
                        set vlans sales l3-interface irb.0
                        set vlans support description "Support VLAN"
                        set vlans support vlan-id 200
                        set vlans support l3-interface irb.1
```

Step-by-Step Procedure

To configure the distribution switch:

1. Configure the interface on the switch to be the trunk port that connects to the access switch:

```
[edit interfaces]
user@distribution-switch# set ge-0/0/0 unit 0 description "Connection to access switch"
user@distribution-switch# set ge-0/0/0 unit 0 family ethernet-switching interface-mode
trunk
```

2. Specify the VLANs to be aggregated on the trunk port:

```
[edit interfaces]
user@distribution-switch# set ge-0/0/0 unit 0 family ethernet-switching vlan members [ sales
support ]
```

3. To handle untagged packets that are received on the trunk port, create a native VLAN by configuring a VLAN ID and specifying that the trunk port is a member of the native VLAN:

```
[edit interfaces]
user@distribution-switch# set ge-0/0/0 native-vlan-id 1
user@distribution-switch# set ge-0/0/0 unit 0 family ethernet-switching vlan members 1
```

4. Configure the sales VLAN:

```
[edit vlans]
user@distribution-switch# set sales description "Sales VLAN"
user@distribution-switch# set sales vlan-id 100
user@distribution-switch# set sales l3-interface irb.0
```

The VLAN configuration for the distribution switch includes the **set l3-interface irb.0** command to route traffic between the sales and support VLANs. The VLAN configuration for the access switch does not include this statement because the access switch is not monitoring IP addresses. Instead, the access switch is passing the IP addresses to the distribution switch for interpretation.

5. Configure the support VLAN:

```
[edit vlans]
user@distribution-switch# set support description "Support VLAN"
user@distribution-switch# set support vlan-id 200
user@distribution-switch# set support l3-interface irb.1
```

The VLAN configuration for the distribution switch includes the **set l3-interface irb.1** command to route traffic between the sales and support VLANs. The VLAN configuration for the access switch does not include this statement because the access switch is not monitoring IP addresses. Instead, the access switch is passing the IP addresses to the distribution switch for interpretation.

6. Create the subnet for the sales VLAN:

```
[edit interfaces]
user@distribution-switch# set irb unit 0 family inet address
192.0.2.2/25
```

7. Create the subnet for the support VLAN:

```
[edit interfaces]
user@distribution-switch# set irb unit 1 family inet address
192.0.2.130/25
```

Results

Display the results of the configuration:

```
user@distribution-switch> show configuration
interfaces {
    ge-0/0/0 {
        native-vlan-id 1;
        unit 0 {
            description "Connection to access switch";
            family ethernet-switching {
                interface-mode trunk;
                vlan {
                    members [ 1 sales support ];
                }
            }
        }
    }
    irb {
        unit 0 {
            family inet {
                address 192.0.2.2/25;
            }
        }
        unit 1 {
            family inet {
                address 192.0.2.130/25;
            }
```

```
        }
    }
}
vlans {
    sales {
        description "Sales VLAN";
        vlan-id 100;
        l3-interface irb.0;
    }
    support {
        description "Support VLAN";
        vlan-id 200;
        l3-interface irb.1;
    }
}
```

TIP: To quickly configure the distribution switch, issue the `load merge terminal` command, then copy the hierarchy and paste it into the switch terminal window.

Verification

IN THIS SECTION

To confirm that the configuration is working properly, perform these tasks:

Verifying the VLAN Members and Interfaces on the Access Switch

Purpose

Verify that the **sales** and **support** VLANs have been created on the switch.

Action

List all VLANs configured on the switch:

```
user@access-switch>                          show
vlans
Routing instance        VLAN name             Tag         Interfaces
default-switch          sales                    100
                                                            ge-0/0/20.0
                                                            ge-0/0/22.0
                                                            ge-0/0/3.0*
                                                            ge-0/0/0.0*
                                                            ge-0/2/0.0*
default-switch          support                  200
                                                            ge-0/0/24.0
                                                            ge-0/0/26.0
                                                            ge-0/0/44.0*
                                                            ge-0/0/46.0*
                                                            ge-0/2/0.0*
```

Meaning

The output shows the **sales** and **support** VLANs and the interfaces that are configured as members of the respective VLANs.

Verifying the VLAN Members and Interfaces on the Distribution Switch

Purpose

Verify that the **sales** and **support** VLANs have been created on the switch.

Action

List all VLANs configured on the switch:

```
user@distribution-switch>                          show
vlans

Routing instance        VLAN name             Tag         Interfaces
default-switch          sales                    100
```

```
                                                    ge-0/0/0.0*
default-switch          support                 200
                                                    ge-0/0/0.0*
```

Meaning

The output shows the **sales** and **support** VLANs and the interface (ge-0/0/0.0) that is configured as a member of both VLANs. Interface ge-0/0/0.0 is also the trunk interface connected to the access switch.

Example: Setting Up Bridging with Multiple VLANs for EX Series Switches

IN THIS SECTION

To segment traffic on a LAN into separate broadcast domains, you create separate virtual LANs (VLANs) on an EX Series switch. Each VLAN is a collection of network nodes. When you use VLANs, frames whose origin and destination are in the same VLAN are forwarded only within the local VLAN, and only frames not destined for the local VLAN are forwarded to other broadcast domains. VLANs thus limit the amount of traffic flowing across the entire LAN, reducing the possible number of collisions and packet retransmissions within the LAN.

This example describes how to configure bridging for an EX Series switch and how to create two VLANs to segment the LAN:

Requirements

This example uses the following hardware and software components:

- One EX4200-48P Virtual Chassis switch
- Junos OS Release 9.0 or later for EX Series switches

Before you set up bridging and VLANs, be sure you have:

- Performed the initial switch configuration. See *Connecting and Configuring an EX Series Switch (J-Web Procedure)*.

Overview and Topology

IN THIS SECTION

EX Series switches connect all devices in an office or data center into a single LAN to provide sharing of common resources such as printers and file servers and to enable wireless devices to connect to the LAN through wireless access points. The default configuration creates a single VLAN, and all traffic on the switch is part of that broadcast domain. Creating separate network segments reduces the span of the broadcast domain and allows you to group related users and network resources without being limited by physical cabling or by the location of a network device in the building or on the LAN.

This example shows a simple configuration to illustrate the basic steps for creating two VLANs on a single switch. One VLAN, called **sales**, is for the sales and marketing group, and a second, called **support**, is for the customer support team. The sales and support groups each have their own dedicated file servers, printers, and wireless access points. For the switch ports to be segmented across the two VLANs, each VLAN must have its own broadcast domain, identified by a unique name and tag (VLAN ID). In addition, each VLAN must be on its own distinct IP subnet.

Topology

The topology for this example consists of one EX4200-48P switch, which has a total of 48 Gigabit Ethernet ports, all of which support Power over Ethernet (PoE). Most of the switch ports connect to Avaya IP telephones. The remainder of the ports connect to wireless access points, file servers, and printers. Table 54 on page 241 explains the components of the example topology.

Table 54: Components of the Multiple VLAN Topology

Property	Settings
Switch hardware	EX4200-48P, 48 Gigabit Ethernet ports, all PoE-enabled (**ge-0/0/0** through **ge-0/0/47**)

Table 54: Components of the Multiple VLAN Topology *(Continued)*

Property	Settings
VLAN names and tag IDs	**sales**, tag **100** **support**, tag **200**
VLAN subnets	**sales**: **192.0.2.0/25** (addresses **192.0.2.1** through **192.0.2.126**) **support**: **192.0.2.128/25** (addresses **192.0.2.129** through **192.0.2.254**)
Interfaces in VLAN **sales**	Avaya IP telephones: **ge-0/0/3** through **ge-0/0/19** Wireless access points: **ge-0/0/0** and **ge-0/0/1** Printers: **ge-0/0/22** and **ge-0/0/23** File servers: **ge-0/0/20** and **ge-0/0/21**
Interfaces in VLAN **support**	Avaya IP telephones: **ge-0/0/25** through **ge-0/0/43** Wireless access points: **ge-0/0/24** Printers: **ge-0/0/44** and **ge-0/0/45** File servers: **ge-0/0/46** and **ge-0/0/47**
Unused interfaces	**ge-0/0/2** and **ge-0/0/25**

This configuration example creates two IP subnets, one for the sales VLAN and the second for the support VLAN. The switch bridges traffic within a VLAN. For traffic passing between two VLANs, the switch routes the traffic using a Layer 3 routing interface on which you have configured the address of the IP subnet.

To keep the example simple, the configuration steps show only a few devices in each of the VLANs. Use the same configuration procedure to add more LAN devices.

Configuration

IN THIS SECTION

Configure Layer 2 switching for two VLANs:

Procedure

CLI Quick Configuration

To quickly configure Layer 2 switching for the two VLANs (**sales** and **support**) and to quickly configure Layer 3 routing of traffic between the two VLANs, copy the following commands and paste them into the switch terminal window:

```
[edit]
                        set interfaces ge-0/0/0 unit 0 description “Sales wireless access point
port”
                        set interfaces ge-0/0/0 unit 0 family ethernet-switching vlan members
sales
                        set interfaces ge-0/0/3  unit 0 description “Sales phone
port”
                        set interfaces ge-0/0/3 unit 0 family ethernet-switching vlan members
sales
                        set interfaces ge-0/0/22  unit 0 description “Sales printer
port”
                        set interfaces ge-0/0/22 unit 0 family ethernet-switching vlan members
sales
                        set interfaces ge-0/0/20 unit 0 description “Sales file server
port”
                        set interfaces ge-0/0/20 unit 0 family ethernet-switching vlan members
sales
                        set interfaces ge-0/0/24 unit 0 description “Support wireless access
point port”
                        set interfaces ge-0/0/24 unit 0 family ethernet-switching vlan members
support
                        set interfaces ge-0/0/26  unit 0 description “Support phone
port”
                        set interfaces ge-0/0/26 unit 0 family ethernet-switching vlan members
support
                        set interfaces ge-0/0/44  unit 0 description “Support printer
port”
                        set interfaces ge-0/0/44 unit 0 family ethernet-switching vlan members
support
                        set interfaces ge-0/0/46  unit 0 description “Support file server
port”
                        set interfaces ge-0/0/46 unit 0 family ethernet-switching vlan members
support
                        set interfaces irb unit 0 family inet address
```

```
192.0.2.0/25
                        set interfaces irb unit 1 family inet address
192.0.2.128/25
                        set vlans sales l3-interface irb.0
                        set vlans sales vlan-id 100
                        set vlans support vlan-id 200
                        set vlans support l3-interface irb.1
```

Step-by-Step Procedure

Configure the switch interfaces and the VLANs to which they belong. By default, all interfaces are in access mode, so you do not have to configure the port mode.

1. Configure the interface for the wireless access point in the sales VLAN:

```
[edit interfaces ge-0/0/0 unit 0]
user@switch# set description “Sales wireless access point port”
user@switch# set family ethernet-switching vlan members sales
```

2. Configure the interface for the Avaya IP phone in the sales VLAN:

```
[edit interfaces ge-0/0/3 unit 0]
user@switch# set description “Sales phone port”
user@switch# set family ethernet-switching vlan members sales
```

3. Configure the interface for the printer in the sales VLAN:

```
[edit interfaces ge-0/0/22 unit 0]
user@switch# set description “Sales printer port”
user@switch# set family ethernet-switching vlan members sales
```

4. Configure the interface for the file server in the sales VLAN:

```
[edit interfaces ge-0/0/20 unit 0]
user@switch# set description “Sales file server port”
user@switch# set family ethernet-switching vlan members sales
```

5. Configure the interface for the wireless access point in the support VLAN:

```
[edit interfaces ge-0/0/24 unit 0]
user@switch# set description “Support wireless access point port”
user@switch# set family ethernet-switching vlan members support
```

6. Configure the interface for the Avaya IP phone in the support VLAN:

```
[edit interfaces ge-0/0/26 unit 0]
user@switch# set description “Support phone port”
user@switch# set family ethernet-switching vlan members support
```

7. Configure the interface for the printer in the support VLAN:

```
[edit interfaces ge-0/0/44 unit 0]
user@switch# set description “Support printer port”
user@switch# set family ethernet-switching vlan members support
```

8. Configure the interface for the file server in the support VLAN:

```
[edit interfaces ge-0/0/46 unit 0]
user@switch# set description “Support file server port”
user@switch# set family ethernet-switching vlan members support
```

9. Create the subnet for the sales broadcast domain:

```
[edit interfaces]
user@switch# set irb unit 0 family inet address 192.0.2.1/25
```

10. Create the subnet for the support broadcast domain:

```
[edit interfaces]
user@switch# set irb unit 1 family inet address 192.0.2.129/25
```

11. Configure the VLAN tag IDs for the sales and support VLANs:

```
[edit vlans]
user@switch# set sales vlan-id 100
user@switch# set support vlan-id 200
```

12. To route traffic between the sales and support VLANs, define the interfaces that are members of each VLAN and associate a Layer 3 interface:

```
[edit vlans]
user@switch# set sales l3-interface irb.0
user@switch# set support l3-interface irb.1
```

Results

Display the results of the configuration:

```
user@switch> show configuration
interfaces {
    ge-0/0/0 {
        unit 0 {
            description “Sales wireless access point port”;
            family ethernet-switching {
                vlan members sales;
            }
        }
    }
    ge-0/0/3 {
        unit 0 {
            description “Sales phone port”;
            family ethernet-switching {
                vlan members sales;
            }
        }
    }
    ge-0/0/22 {
        unit 0 {
            description “Sales printer port”;
            family ethernet-switching {
                vlan members sales;
```

```
        }
    }
}
ge-0/0/20 {
    unit 0 {
        description “Sales file server port”;
        family ethernet-switching {
            vlan members sales;
        }
    }
}
ge-0/0/24 {
    unit 0 {
        description “Support wireless access point port”;
        family ethernet-switching {
            vlan members support;
        }
    }
}
ge-0/0/26 {
    unit 0 {
        description “Support phone port”;
        family ethernet-switching {
            vlan members support;
        }
    }
}
ge-0/0/44 {
    unit 0 {
        description “Support printer port”;
        family ethernet-switching {
            vlan members support;
        }
    }
}
ge-0/0/46 {
    unit 0 {
        description “Support file server port”;
        family ethernet-switching {
            vlan members support;
        }
    }
}
```

```
    irb {
            unit 0 {
                family inet address 192.0.2.0/25;
            }
            unit 1 {
                family inet address 192.0.2.128/25;
            }
    }
}
vlans {
        sales {
                vlan-id 100;
                interface xe-0/0/0.0:
                interface xe-0/0/3/0;
                interface xe-0/0/20.0;
                interface xe-0/0/22.0;
                l3-interface irb.0;
        }
        support {
                vlan-id 200;
                interface xe-0/0/24.0:
                interface xe-0/0/26.0;
                interface xe-0/0/44.0;
                interface xe-0/0/46.0;
                l3-interface irb.1;
        }
}
```

TIP: To quickly configure the sales and support VLAN interfaces, issue the `load merge terminal` command, then copy the hierarchy and paste it into the switch terminal window.

Verification

IN THIS SECTION

To verify that the “sales” and “support” VLANs have been created and are operating properly, perform these tasks:

Verifying That the VLANs Have Been Created and Associated to the Correct Interfaces

Purpose

Verify that the VLANs **sales** and **support** have been created on the switch and that all connected interfaces on the switch are members of the correct VLAN.

Action

List all VLANs configured on the switch:

Use the operational mode commands:

```
user@switch>                            show
vlans
Name            Tag     Interfaces
default
                        ge-0/0/1.0, ge-0/0/2.0, ge-0/0/4.0, ge-0/0/5.0,
                        ge-0/0/6.0,  ge-0/0/7.0, ge-0/0/8.0, ge-0/0/9.0,
                        ge-0/0/10.0*, ge-0/0/11.0, ge-0/0/12.0, ge-0/0/13.0*,
                        ge-0/0/14.0, ge-0/0/15.0, ge-0/0/16.0, ge-0/0/17.0,
                        ge-0/0/18.0, ge-0/0/19.0, ge-0/0/21.0, ge-0/0/23.0*,
                        ge-0/0/25.0, ge-0/0/27.0, ge-0/0/28.0, ge-0/0/29.0,
                        ge-0/0/30.0, ge-0/0/31.0, ge-0/0/32.0, ge-0/0/33.0,
                        ge-0/0/34.0, ge-0/0/35.0, ge-0/0/36.0, ge-0/0/37.0,
                        ge-0/0/38.0, ge-0/0/39.0, ge-0/0/40.0, ge-0/0/41.0,
                        ge-0/0/42.0, ge-0/0/43.0, ge-0/0/45.0, ge-0/0/47.0,
                        ge-0/1/0.0*, ge-0/1/1.0*, ge-0/1/2.0*, ge-0/1/3.0*

sales       100
                        ge-0/0/0.0*, ge-0/0/3.0, ge-0/0/20.0, ge-0/0/22.0

support     200
                        ge-0/0/24.0, ge-0/0/26.0, ge-0/0/44.0, ge-0/0/46.0*
mgmt
                        me0.0*
```

Meaning

The `show vlans` command lists all VLANs configured on the switch and which interfaces are members of each VLAN. This command output shows that the **sales** and **support** VLANs have been created. The **sales** VLAN has a tag ID of 100 and is associated with interfaces **ge-0/0/0.0**, **ge-0/0/3.0**, **ge-0/0/20.0**, and **ge-0/0/22.0**. VLAN **support** has a tag ID of 200 and is associated with interfaces **ge-0/0/24.0**, **ge-0/0/26.0**, **ge-0/0/44.0**, and **ge-0/0/46.0**.

Verifying That Traffic Is Being Routed Between the Two VLANs

Purpose

Verify routing between the two VLANs.

Action

List the Layer 3 routes in the switch's Address Resolution Protocol (ARP) table:

```
user@switch> show arp
MAC Address       Address         Name                    Flags

00:00:0c:06:2c:0d        192.0.2.3       irb.0                   None
00:13:e2:50:62:e0        192.0.2.11      irb.1                   None
```

Meaning

Sending IP packets on a multiaccess network requires mapping from an IP address to a MAC address (the physical or hardware address). The ARP table displays the mapping between the IP address and MAC address for both **irb.0** (associated with **sales**) and **irb.1** (associated with **support**). These VLANs can route traffic to each other.

Verifying That Traffic Is Being Switched Between the Two VLANs

Purpose

Verify that learned entries are being added to the Ethernet switching table.

Action

List the contents of the Ethernet switching table:

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

Ethernet-switching table: 8 entries, 5 learned
  VLAN              MAC address       Type         Age Interfaces
  default           *                 Flood          - All-members
  default           00:00:05:00:00:01 Learn          - ge-0/0/10.0
  default           00:00:5e:00:01:09 Learn          - ge-0/0/13.0
  default           00:19:e2:50:63:e0 Learn          - ge-0/0/23.0
  sales             *                 Flood          - All-members
  sales             00:00:5e:00:07:09 Learn          - ge-0/0/0.0
  support           *                 Flood          - All-members
  support           00:00:5e:00:01:01 Learn          - ge-0/0/46.0
```

Meaning

The output shows that learned entries for the **sales** and **support** VLANs have been added to the Ethernet switching table, and are associated with interfaces **ge-0/0/0.0** and **ge-0/0/46.0**. Even though the VLANs were associated with more than one interface in the configuration, these interfaces are the only ones that are currently operating.

SEE ALSO

Example: Setting Up Basic Bridging and a VLAN for an EX Series Switch | **195**

Example: Connecting an EX Series Access Switch to a Distribution Switch

Understanding Bridging and VLANs on Switches | **140**

Example: Connecting an Access Switch to a Distribution Switch

IN THIS SECTION

In large local area networks (LANs), you commonly need to aggregate traffic from a number of access switches into a distribution switch.

This example describes how to connect an access switch to a distribution switch:

Requirements

This example uses the following hardware and software components:

- For the distribution switch, one EX 4200-24F switch. This model is designed to be used as a distribution switch for aggregation or collapsed core network topologies and in space-constrained data centers. It has twenty-four 1-Gigabit Ethernet fiber SFP ports and an EX-UM-2XFP uplink module with two 10-Gigabit Ethernet XFP ports.
- For the access switch, one EX 3200-24P, which has twenty-four 1-Gigabit Ethernet ports, all of which support Power over Ethernet (PoE), and an uplink module with four 1-Gigabit Ethernet ports.
- Junos OS Release 11.1 or later for the QFX Series

Overview and Topology

IN THIS SECTION

In a large office that is spread across several floors or buildings, or in a data center, you commonly aggregate traffic from a number of access switches into a distribution switch. This configuration example shows a simple topology to illustrate how to connect a single access switch to a distribution switch.

In the topology, the LAN is segmented into two VLANs, one for the sales department and the second for the support team. One 1-Gigabit Ethernet port on the access switch's uplink module connects to the distribution switch, to one 1-Gigabit Ethernet port on the distribution switch.

Topology

Table 55 on page 253 explains the components of the example topology. The example shows how to configure one of the three access switches. The other access switches could be configured in the same manner.

Table 55: Components of the Topology for Connecting an Access Switch to a Distribution Switch

Property	Settings
Access switch hardware	EX 3200-24P, 24 1-Gigabit Ethernet ports, all PoE-enabled (**ge-0/0/0** through **ge-0/0/23**); one 4-port 1–Gigabit Ethernet uplink module (EX-UM-4SFP)
Distribution switch hardware	EX 4200-24F, 24 1-Gigabit Ethernet fiber SFP ports (**ge-0/0/0** through **ge-0/0/23**); one 2–port 10–Gigabit Ethernet XFP uplink module (EX-UM-4SFP)
VLAN names and tag IDs	**sales**, tag **100support**, tag **200**
VLAN subnets	**sales**: **192.0.2.0/25** (addresses **192.0.2.1** through **192.0.2.126**)**support**: **192.0.2.128/25** (addresses **192.0.2.129** through **192.0.2.254**)
Trunk port interfaces	On the access switch: **ge-0/1/0**On the distribution switch: **ge-0/0/0**
Access port interfaces in VLAN **sales** (on access switch)	Avaya IP telephones: **ge-0/0/3** through **ge-0/0/19**Wireless access points: **ge-0/0/0** and **ge-0/0/1**Printers: **ge-0/0/22** and **ge-0/0/23**File servers: **ge-0/0/20** and **ge-0/0/21**
Access port interfaces in VLAN **support** (on access switch)	Avaya IP telephones: **ge-0/0/25** through **ge-0/0/43**Wireless access points: **ge-0/0/24**Printers: **ge-0/0/44** and **ge-0/0/45**File servers: **ge-0/0/46** and **ge-0/0/47**

Unused interfaces on access switch	**ge-0/0/2** and **ge-0/0/25**

Configuring the Access Switch

IN THIS SECTION

- Procedure | 254

To configure the access switch:

Procedure

CLI Quick Configuration

To quickly configure the access switch, copy the following commands and paste them into the switch terminal window:

```
[edit]
                        set interfaces ge-0/0/0 unit 0 description “Sales Wireless access point
port”
                        set interfaces ge-0/0/0 unit 0 family ethernet-switching vlan members
sales
                        set interfaces ge-0/0/3  unit 0 description “Sales phone
port”
                        set interfaces ge-0/0/3 unit 0 family ethernet-switching vlan members
sales
                        set interfaces ge-0/0/22  unit 0 description “Sales printer
port”
                        set interfaces ge-0/0/22 unit 0 family ethernet-switching vlan members
sales
                        set interfaces ge-0/0/20 unit 0 description “Sales file server
port”
                        set interfaces ge-0/0/20 unit 0 family ethernet-switching vlan members
sales
                        set interfaces ge-0/0/24 unit 0 description “Support wireless access
point port”
```

```
                        set interfaces ge-0/0/24 unit 0 family ethernet-switching vlan members
support
                        set interfaces ge-0/0/26  unit 0 description “Support phone
port”
                        set interfaces ge-0/0/26 unit 0 family ethernet-switching vlan members
support
                        set interfaces ge-0/0/44  unit 0 description “Support printer
port”
                        set interfaces ge-0/0/44 unit 0 family ethernet-switching vlan members
support
                        set interfaces ge-0/0/46 unit 0 description “Support file server
port”
                        set interfaces ge-0/0/46 unit 0 family ethernet-switching vlan members
support
                        set interfaces ge-0/1/0 unit 0 description “Uplink module port
connection to distribution switch”
                        set interfaces ge-0/1/0 unit 0 family ethernet-switching port-mode
trunk
                        set interfaces ge-0/1/0 unit 0 family ethernet-switching native-vlan-id
1
                        set interfaces ge-0/1/0 unit 0 family ethernet switching vlan members
[sales support]
                        set interfaces irb unit 0 family inet address
192.0.2.1/25
                        set interfaces irb unit 1 family inet address
192.0.2.129/25
                        set vlans sales interface ge-0/0/0.0
                        set vlans sales interface ge-0/0/3.0
                        set vlans sales interface ge-0/0/22.0
                        set vlans sales interface ge-0/0/20.0
                        set vlans sales l3-interface irb.0
                        set vlans sales vlan-id 100
                        set vlans sales vlan-description “Sales VLAN”
                        set vlans support interface ge-0/0/24.0
                        set vlans support interface ge-0/0/26.0
                        set vlans support interface ge-0/0/44.0
                        set vlans support interface ge-0/0/46.0
                        set vlans support vlan-id 200
                        set vlans support l3-interface irb.1
                        set vlans support vlan-description “Support VLAN”
```

Step-by-Step Procedure

To configure the access switch:

1. Configure the 1-Gigabit Ethernet interface on the uplink module to be the trunk port that connects to the distribution switch:

```
[edit interfaces ge-0/1/0 unit 0]user@access-switch# setdescription “Uplink module port
connection to distribution switch”
user@access-switch# set ethernet-switching port-mode trunk
```

2. Specify the VLANs to be aggregated on the trunk port:

```
[edit interfaces ge-0/1/0 unit 0]user@access-switch# set ethernet-switching vlanmembers
[ sales support ]
```

3. Configure the VLAN ID to use for packets that are received with no dot1q tag (untagged packets):

```
 [edit interfaces ge-0/1/0 unit 0]user@access-switch# set ethernet-switching native-vlan-id
1
```

4. Configure the sales VLAN:

```
[edit vlans sales]user@access-switch# set vlan-description “Sales VLAN”
user@access-switch# set vlan-id  100user@access-switch# set l3-interface (IRB)
irb.0
```

5. Configure the support VLAN:

```
[edit vlans support]user@access-switch# set vlan-description “Support VLAN”
user@access-switch# set vlan-id  200user@access-switch# set l3-interface (IRB)
irb.1
```

6. Create the subnet for the sales broadcast domain:

```
[edit interfaces]user@access-switch# set irb unit 0 family inet address 192.0.2.1/25
```

7. Create the subnet for the support broadcast domain:

```
[edit interfaces]user@access-switch# set irb unit 1 family inet address 192.0.2.129/25
```

8. Configure the interfaces in the sales VLAN:

```
[edit interfaces]user@access-switch# set ge-0/0/0 unit 0 description “Sales wireless access
point port”
user@access-switch# set ge-0/0/0 unit 0 family ethernet-switching vlan members sales
user@access-switch# set ge-0/0/3 unit 0 description “Sales phone port”
user@access-switch# set ge-0/0/3 unit 0 family ethernet-switching vlan members sales
user@access-switch# set ge-0/0/20 unit 0 description “Sales file server port”
user@access—switch# set ge-0/0/20 unit 0 family ethernet-switching vlan members sales
user@access-switch# set ge-0/0/22 unit 0 description “Sales printer port”
user@access-switch# set ge-0/0/22 unit 0 family ethernet-switching vlan members sales
```

9. Configure the interfaces in the support VLAN:

```
[edit interfaces]user@access-switch# set ge-0/0/24 unit 0 description “Support wireless
access point port”
user@access-switch# set ge-0/0/24 unit 0 family ethernet-switching vlan members support
user@access-switch# set ge-0/0/26 unit 0 description “Support phone port”
user@access-switch# set ge-0/0/26 unit 0 family ethernet-switching vlan members support
user@access-switch# set ge-0/0/44 unit 0 description “Support printer port”
user@access-switch# set ge-0/0/44 unit 0 family ethernet-switching vlan members support
user@access-switch# set ge-0/0/46 unit 0 description “Support file server port”
user@access-switch# set ge-0/0/46 unit 0 family ethernet-switching vlan members support
```

10. Configure descriptions and VLAN tag IDs for the sales and support VLANs:

```
[edit vlans]user@access-switch# set sales vlan-description “Sales VLAN”
user@access-switch# set sales vlan-id 100
user@access-switch# set support vlan-description “Support VLAN”
user@access-switch# set support vlan-id 200
```

11. To route traffic between the sales and support VLANs and associate a Layer 3 interface with each VLAN:

```
[edit vlans]user@access-switch# set sales l3-interface irb.0
user@access-switch# set support l3-interface irb.1
```

Results

Display the results of the configuration:

```
user@access-switch> show
interfaces {
    ge-0/0/0 {
        unit 0 {
            description “Sales wireless access point port”;
            family ethernet-switching {
                vlan members sales;
            }
        }
    }
    ge-0/0/3 {
        unit 0 {
            description “Sales phone port”;
                family ethernet-switching {
                vlan members sales;
            }
        }
    }
    ge-0/0/20 {
        unit 0 {
            description “Sales file server port”;
                family ethernet-switching {
                vlan members sales;
            }
        }
    }
    ge-0/0/22 {
        unit 0 {
            description “Sales printer port”;
                family ethernet-switching {
                vlan members sales;
```

```
        }
    }
}
ge-0/0/24 {
    unit 0 {
        description “Support wireless access point port”;
            family ethernet-switching {
            vlan members support;
        }
    }
}
ge-0/0/26 {
    unit 0 {
        description “Support phone port”;
            family ethernet-switching {
            vlan members support;
        }
    }
}
ge-0/0/44 {
    unit 0 {
        description “Support printer port”;
            family ethernet-switching {
            vlan members sales;
        }
    }
}
ge-0/0/46 {
    unit 0 {
        description “Support file server port”;
            family ethernet-switching {
            vlan members support;
        }
    }
}
ge-0/1/0 {
    unit 0 {
        description “Uplink module port connection to distribution switch”;
            family ethernet-switching {
            port-mode trunk;
                vlan members [ sales support ];
            native-vlan-id 1;
        }
```

```
        }
    }
    irb {
        unit 0 {
            family inet address 192.0.2.1/25;
                }
        unit 1 {
            family inet address 192.0.2.129/25;
                }
      }
    vlans {
        sales {
            vlan-id 100;
            vlan-description “Sales VLAN”;
            l3-interface irb.0;
         }
        support {
            vlan-id 200;
            vlan-description “Support VLAN”;
            l3-interface irb.1;
        }
     }
}
```

TIP: To quickly configure the distribution switch, issue the `load merge terminal` command, then copy the hierarchy and paste it into the switch terminal window.

Configuring the Distribution Switch

IN THIS SECTION

To configure the distribution switch:

Procedure

CLI Quick Configuration

To quickly configure the distribution switch, copy the following commands and paste them into the switch terminal window:

```
                        set interfaces ge-0/0/0 description “Connection to access
switch”
                        set interfaces ge-0/0/0 ethernet-switching port-mode
trunk
                        set interfaces ge-0/0/0 ethernet-switching vlan members [ sales
support ]
                        set interfaces ge-0/0/0 ethernet-switching native-vlan-id
1
                        set interfaces irb unit 0 family inet address
192.0.2.2/25
                        set interfaces irb unit 1 family inet address
192.0.2.130/25
                        set vlans sales vlan-description “Sales VLAN”
                        set vlans sales vlan-id 100
                        set vlans sales l3-interface irb.0
                        set vlans support vlan-description “Support VLAN”
                        set vlans support vlan-id 200
                        set vlans support l3-interface irb.1
```

Step-by-Step Procedure

To configure the distribution switch:

1. Configure the interface on the switch to be the trunk port that connects to the access switch:

```
[edit interfaces ge-0/0/0 unit 0]user@distribution-switch# set description “Connection to
access switch”
user@distribution-switch# set ethernet-switching port-mode trunk
```

2. Specify the VLANs to be aggregated on the trunk port:

```
[edit interfaces ge-0/0/0 unit 0]user@distribution-switch# set ethernet-switching vlanmembers
[ sales support ]
```

3. Configure the VLAN ID to use for packets that are received with no dot1q tag (untagged packets):

```
[edit interfaces]user@distribution-switch# set ge-0/0/0 ethernet-switching native-vlan-id 1
```

4. Configure the sales VLAN:

```
[edit vlans sales]user@distribution-switch# set vlan-description “Sales VLAN”
user@distribution-switch# set vlan-id  100user@distribution-switch# set l3-interface (IRB)
irb.0
```

5. Configure the support VLAN:

```
[edit vlans support]user@distribution-switch# set vlan-description “Support VLAN”
user@distribution-switch# set vlan-id  200user@distribution-switch# set l3-interface (IRB)
irb.1
```

6. Create the subnet for the sales broadcast domain:

```
[edit interfaces]user@distribution-switch# set irb unit 0 family inet address
192.0.2.2/25
```

7. Create the subnet for the support broadcast domain:

```
[edit interfaces] user@distribution-switch# set irb unit 1 family inet address
192.0.2.130/25
```

Results

Display the results of the configuration:

```
user@distribution-switch> show
interfaces {
    ge-0/0/0 {
        description “Connection to access switch”;
        unit 0 {
            family ethernet-switching {
                port-mode trunk;
                vlan members [ sales support ];
                native-vlan-id 1;
            }
        }
    }
    irb {
       unit 0 {
            family inet address 192.0.2.2/25;
        }
        unit 1 {
            family inet address 192.0.2.130/25;
        }
    }
    vlans {
        sales {
            vlan-id 100;
            vlan-description “Sales VLAN”;
            l3-interface irb.0;
        }
        support {
            vlan-id 200;
            vlan-description “Support VLAN”;
            l3-interface irb.1;
        }
    }
}
```

TIP: To quickly configure the distribution switch, issue the `load merge terminal` command, then copy the hierarchy and paste it into the switch terminal window.

Verification

IN THIS SECTION

To confirm that the configuration is working properly, perform these tasks:

Verifying the VLAN Members and Interfaces on the Access Switch

Purpose

Verify that the **sales** and **support** have been created on the switch.

Action

List all VLANs configured on the switch:

```
user@switch> show vlans

Name            Tag     Interfaces
default
                        ge-0/0/1.0, ge-0/0/2.0, ge-0/0/4.0, ge-0/0/5.0,
                        ge-0/0/6.0, ge-0/0/7.0, ge-0/0/8.0*, ge-0/0/9.0,
                        ge-0/0/10.0, ge-0/0/11.0*, ge-0/0/12.0, ge-0/0/13.0,
                        ge-0/0/14.0, ge-0/0/15.0, ge-0/0/16.0, ge-0/0/17.0,
                        ge-0/0/18.0, ge-0/0/19.0*,ge-0/0/21.0, ge-0/0/23.0,
                        ge-0/0/25.0, ge-0/0/27.0*,ge-0/0/28.0, ge-0/0/29.0,
                        ge-0/0/30.0, ge-0/0/31.0*,ge-0/0/32.0, ge-0/0/33.0,
                        ge-0/0/34.0, ge-0/0/35.0*,ge-0/0/36.0, ge-0/0/37.0,
                        ge-0/0/38.0, ge-0/0/39.0*,ge-0/0/40.0, ge-0/0/41.0,
                        ge-0/0/42.0, ge-0/0/43.0*,ge-0/0/45.0, ge-0/0/47.0,
                        ge-0/1/1.0*, ge-0/1/2.0*, ge-0/1/3.0*

sales           100
                        ge-0/0/0.0*, ge-0/0/3.0, ge-0/0/20.0, ge-0/0/22.0,
                        ge-0/1/0.0*,
```

```
support       200
                        ge-0/0/24.0*, ge-0/0/26.0, ge-0/0/44.0, ge-0/0/46.0,

mgmt
                        me0.0*
```

Meaning

The output shows the **sales** and **support** VLANs and the interfaces associated with them.

Verifying the VLAN Members and Interfaces on the Distribution Switch

Purpose

Verify that the **sales** and **support** have been created on the switch.

Action

List all VLANs configured on the switch:

```
user@switch> show vlans

Name            Tag     Interfaces
default
                        ge-0/0/1.0, ge-0/0/2.0, ge-0/0/3.0, ge-0/0/4.0,
                        ge-0/0/5.0, ge-0/0/6.0, ge-0/0/7.0*, ge-0/0/8.0,
                        ge-0/0/9.0, ge-0/0/10.0*, ge-0/0/11.0, ge-0/0/12.0,
                        ge-0/0/13.0, ge-0/0/14.0, ge-0/0/15.0, ge-0/0/16.0,
                        ge-0/0/17.0, ge-0/0/18.0*, ge-0/0/19.0, ge-0/0/20.0,
                        ge-0/0/21.0, ge-0/0/22.0*, ge-0/0/23.0, ge-0/1/1.0*,
                        ge-0/1/2.0*, ge-0/1/3.0*

sales         100
                        ge-0/0/0.0*

support       200
                        ge-0/0/0.0*
```

```
mgmt
                        me0.0*
```

Meaning

The output shows the **sales** and **support** VLANs associated to interface **ge-0/0/0.0**. Interface **ge-0/0/0.0** is the trunk interface connected to the access switch.

Configuring a Logical Interface for Access Mode

Enterprise network administrators can configure a single logical interface to accept untagged packets and forward the packets within a specified VLAN. A logical interface configured to accept untagged packets is called an *access interface* or *access port*.

```
interface-mode access;
```

You can include this statement at the following hierarchy levels:

- [edit interfaces *interface-name* unit *logical-unit-number* family ethernet-switching]
- [edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number* family ethernet-switching]

When an untagged or tagged packet is received on an access interface, the packet is accepted, the VLAN ID is added to the packet, and the packet is forwarded within the VLAN that is configured with the matching VLAN ID.

The following example configures a logical interface as an access port with a VLAN ID of 20 on routers and switches that support the enhanced Layer 2 software:

```
[edit interfaces ge-1/2/0]
unit 1 {
    family ethernet-switching {
        interface-mode access;
        vlan members 20;
    }
}
```

SEE ALSO

Ethernet Interfaces User Guide for Routing Devices

Configuring the Native VLAN Identifier

NOTE: This task uses Junos OS for EX Series switches and Junos OS for QFX3500 and QFX3600 switches that does not support the Enhanced Layer 2 Software (ELS) configuration style. If your switch runs software that supports ELS, see "Configuring the Native VLAN Identifier on Switches With ELS Support" on page 267. For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15.

EX Series switches support receiving and forwarding routed or bridged Ethernet frames with 802.1Q VLAN tags. The logical interface on which untagged packets are to be received must be configured with the same native VLAN ID as that configured on the physical interface.

To configure the native VLAN ID using the CLI:

1. Configure the port mode so that the interface is in multiple VLANs and can multiplex traffic between different VLANs. Trunk interfaces typically connect to other switches and to routers on the LAN. Configure the port mode as **trunk**:

```
[edit interfaces ge-0/0/3 unit 0 family ethernet-switching]
user@switch# set port-mode trunk
```

2. Configure the native VLAN ID:

```
[edit interfaces ge-0/0/3 unit 0 family ethernet-switching]
user@switch# set native-vlan-id 3
```

Configuring the Native VLAN Identifier on Switches With ELS Support

NOTE: This task uses Junos OS for EX Series switches and Junos OS for QFX3500 and QFX3600 switches with support for the Enhanced Layer 2 Software (ELS) configuration

style. If your switch runs software that does not support ELS, see "Configuring the Native VLAN Identifier" on page 267. For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15.

Switches can receive and forward routed or bridged Ethernet frames with 802.1Q VLAN tags. Typically, trunk ports, which connect switches to each other, accept untagged control packets but do not accept untagged data packets. You can enable a trunk port to accept untagged data packets by configuring a native VLAN ID on the interface on which you want the untagged data packets to be received. The logical interface on which untagged packets are to be received must be configured with the same VLAN ID as the native VLAN ID configured on the physical interface.

To configure the native VLAN ID by using the command-line interface (CLI):

1. On the interface on which you want untagged data packets to be received, set the interface mode to `trunk`, which specifies that the interface is in multiple VLANs and can multiplex traffic between different VLANs.:

```
[edit interfaces]
user@switch# set interface-name unit logical-unit-number family ethernet-switching interface-mode trunk
```

2. Configure the native VLAN ID:

```
[edit interfaces]
user@switch# set interface-name native-vlan-id vlan-id
```

3. Specify that the logical interface that will receive the untagged data packets is a member of the native VLAN:

```
[edit interfaces]
user@switch# set interface-name unit logical-unit-number family ethernet-switching vlan members vlan-id
```

Configuring VLAN Encapsulation

IN THIS SECTION

To configure encapsulation on an interface, enter the `encapsulation` statement at the `[edit interfaces` *`interface-name`*`]` hierarchy level:

```
[edit interfaces interface-name]
encapsulation type;
```

The following list contains important notes regarding encapsulation:

- Ethernet interfaces in VLAN mode can have multiple logical interfaces. In CCC and VPLS modes, VLAN IDs from 1 through 511 are reserved for normal VLANs, and VLAN IDs 512 through 4094 are reserved for CCC or VPLS VLANs. For 4-port Fast Ethernet interfaces, you can use VLAN IDs 512 through 1024 for CCC or VPLS VLANs.
- For encapsulation type **flexible-ethernet-services**, all VLAN IDs are valid.
- For some encapsulation types, including flexible Ethernet services, Ethernet VLAN CCC, and VLAN VPLS, you can also configure the encapsulation type that is used inside the VLAN circuit itself. To do this, include the `encapsulation` statement:

  ```
  encapsulation (vlan-ccc | vlan-tcc | vlan-vpls);
  ```

  You can include this statement at the following hierarchy levels:

  - `[edit interfaces` *`interface-name`* `unit` *`logical-unit-number`*`]`
  - `[edit logical-systems` *`logical-system-name`* `interfaces` *`interface-name`* `unit` *`logical-unit-number`*`]`
- You cannot configure a logical interface with VLAN CCC or VLAN VPLS encapsulation unless you also configure the physical device with the same encapsulation or with flexible Ethernet services encapsulation. In general, the logical interface must have a VLAN ID of 512 or higher; if the VLAN ID is 511 or lower, it will be subject to the normal destination filter lookups in addition to source

address filtering. However if you configure flexible Ethernet services encapsulation, this VLAN ID restriction is removed.

In general, you configure an interface's encapsulation at the [edit interfaces *interface-name*] hierarchy level.

Example: Configuring VLAN Encapsulation on a Gigabit Ethernet Interface

Configure VLAN CCC encapsulation on a Gigabit Ethernet interface:

```
interfaces ge-2/1/0 {
    vlan-tagging;
    encapsulation vlan-ccc;
    unit 0 {
        encapsulation vlan-ccc;
        vlan-id 600;
    }
}
```

Example: Configuring VLAN Encapsulation on an Aggregated Ethernet Interface

Configure VLAN CCC encapsulation on an aggregated Gigabit Ethernet interface:

```
interfaces ae0 {
    vlan-tagging;
    encapsulation vlan-vpls;
    unit 0 {
        vlan-id 100;
    }
}
```

SEE ALSO

Ethernet Interfaces User Guide for Routing Devices

Change History Table

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

Release	Description
17.3R1	Starting in Junos OS Release 17.3 on QFX10000 switches, the number of vmembers has increased to 256k for integrated routing and bridging interfaces and aggregated Ethernet interfaces.
17.1R3	Starting with Junos OS Release 17.1R3, on QFX10000 switches, you cannot configure an interface with both `family ethernet-switching` and `flexible-vlan-tagging`.

10
CHAPTER

Configuring 802.1Q VLANs

IN THIS CHAPTER

802.1Q VLANs Overview

A VLAN (virtual LAN) abstracts the idea of the local area network (LAN) by providing data link connectivity for a subnet. VLANs make it easy for network administrators to partition a single switched network to match the functional and security requirements of their systems without having to run new cables or make major changes in their current network infrastructure. Each VLAN can be uniquely identified by VLAN ID, which is transmitted and received as IEEE 802.1Q tag in an Ethernet frame.

You can partition the router into up to 4095 different VLANs—depending on the router model and the physical interface types—by associating logical interfaces with specific VLAN IDs.

For Ethernet, Tri-Rate Ethernet copper, Gigabit Ethernet, 10-Gigabit Ethernet, and aggregated Ethernet interfaces supporting VPLS, the Junos OS supports a subset of the IEEE 802.1Q standard for channelizing an Ethernet interface into multiple logical interfaces, allowing many hosts to be connected to the same Gigabit Ethernet switch, but preventing them from being in the same routing or bridging domain.

RELATED DOCUMENTATION

- Configuring Dynamic 802.1Q VLANs | **276**
- 802.1Q VLAN IDs and Ethernet Interface Types | **275**
- Enabling VLAN Tagging | **277**
- Binding VLAN IDs to Logical Interfaces | **292**
- Guidelines for Configuring VLAN ID List-Bundled Logical Interfaces That Connect CCCs | **321**
- Configuring a Layer 2 VPN Routing Instance on a VLAN-Bundled Logical Interface | **301**
- Configuring a VLAN-Bundled Logical Interface to Support a Layer 2 VPN Routing Instance | **314**
- Specifying the Interface Over Which VPN Traffic Travels to the CE Router | **307**
- Specifying the Interface to Handle Traffic for a CCC | **324**
- Configuring a Layer 2 Circuit on a VLAN-Bundled Logical Interface | **316**
- Configuring a VLAN-Bundled Logical Interface to Support a Layer 2 Circuit | **315**
- Specifying the Interface to Handle Traffic for a CCC Connected to the Layer 2 Circuit | **325**
- Example: Configuring a Layer 2 VPN Routing Instance on a VLAN-Bundled Logical Interface | **304**
- Example: Configuring a Layer 2 Circuit on a VLAN-Bundled Logical Interface | **319**
- Configuring Access Mode on a Logical Interface | **307**
- Configuring a Logical Interface for Trunk Mode | **309**
- Configuring the VLAN ID List for a Trunk Interface | **310**

Configuring a Trunk Interface on a Bridge Network | 311

Ethernet Interfaces User Guide for Routing Devices

802.1Q VLAN IDs and Ethernet Interface Types

VLAN ID 0 is reserved for tagging the priority of frames. VLAN IDs 1 through 511 are reserved for normal VLANs. VLAN IDs 512 and above are reserved for VLAN circuit cross-connect (CCCs).

For Gigabit Ethernet IQ interfaces and Gigabit Ethernet PICs with SFPs , you can configure flexible Ethernet services encapsulation on the physical interface. With flexible Ethernet services encapsulation, VLAN IDs from 1 through 511 are no longer reserved for normal VLANs.

The maximum number of user-configurable VLANs is 15 on each port of the Dense-FE PIC (8-port/12-port/48-port).

Table 56 on page 275 lists VLAN ID range by interface type.

Table 56: VLAN ID Range by Interface Type

Interface Type	VLAN ID Range
Aggregated Ethernet for Fast Ethernet	1 through 1023
Aggregate Ethernet for Gigabit Ethernet	1 through 4094
4-port, 8-port, and 12-port Fast Ethernet	1 through 1023
48-port Fast Ethernet	1 through 4094
Tri-Rate Ethernet copper	1 through 4094
Gigabit Ethernet	1 through 4094
Gigabit Ethernet IQ	1 through 4094
10-Gigabit Ethernet	1 through 4094

Table 56: VLAN ID Range by Interface Type *(Continued)*

Interface Type	VLAN ID Range
100-Gigabit Ethernet	1 through 4094
Management and internal Ethernet interfaces	1 through 1023

NOTE: For Gigabit Ethernet IQ and Gigabit Ethernet PICs with SFPs , VLAN IDs on a single interface can differ from each other.

Because IS-IS has an 8-bit limit for broadcast multiaccess media, you cannot set up more than 255 adjacencies over Gigabit Ethernet using VLAN tagging. For more information, see the Junos OS Routing Protocols Library for Routing Devices.

RELATED DOCUMENTATION

802.1Q VLANs Overview | 274

Ethernet Interfaces User Guide for Routing Devices

Configuring Dynamic 802.1Q VLANs

You can configure the router to dynamically create VLANs when a client accesses an interface and requests a VLAN ID that does not yet exist. When a client accesses a VLAN interface, the router instantiates a VLAN dynamic profile that you have associated with the interface. Using the settings in the dynamic profile, the router extracts information about the client from the incoming packet (for example, the interface and unit values), saves this information in the routing table, and creates a VLAN or stacked VLAN ID for the client from a range of VLAN IDs that you configure for the interface.

Dynamically configuring VLANs or stacked VLANs requires the following general steps:

1. Configure a dynamic profile for dynamic VLAN or dynamic stacked VLAN creation.
2. Associate the VLAN or stacked VLAN dynamic profile with the interface.
3. Specify the Ethernet packet type that the VLAN dynamic profile accepts.
4. Define VLAN ranges for use by the dynamic profile when creating VLAN IDs.

For procedures on how to configure dynamic VLANs and dynamic stacked VLANs for client access, see the Junos OS Subscriber Management and Services Library .

RELATED DOCUMENTATION

802.1Q VLANs Overview | 274

Ethernet Interfaces User Guide for Routing Devices

Enabling VLAN Tagging

You can configure the router to receive and forward single-tag frames, dual-tag frames, or a mixture of single-tag and dual-tag frames.

1. To configure the router to receive and forward single-tag frames with 802.1Q VLAN tags, include the `vlan-tagging` statement at the `[edit interfaces interface-name]` hierarchy level:

```
[edit interfaces interface-name]
user@host# vlan-tagging;
```

2. To configure the router to receive and forward dual-tag frames with 802.1Q VLAN tags, include the `stacked-vlan-tagging` statement at the `[edit interfaces interface-name]` hierarchy level:

```
[edit interfaces interface-name]
user@host# stacked-vlan-tagging;
```

3. Mixed tagging is supported for all router Gigabit and 10-Gigabit Ethernet interfaces on MX Series routers and for aggregated Ethernet interfaces with member links in IQ2 and IQ2-E PICs or in MX Series DPCs. Mixed tagging enables to configure two logical interfaces on the same Ethernet port, one with single-tag framing and one with dual-tag framing.

   [upoovaiahpk/appu 04/12/25] Removed reference to fast ethernet per Junos Detox Phase 2.

   To configure mixed tagging:

   a. Configure the `flexible-vlan-tagging` statement at the `[edit interfaces ge-fpc/pic/port]` hierarchy level.

```
[edit interfaces ge-fpc/pic/port]
user@host#flexible-vlan-tagging;
```

b. Configure the `vlan-tags` statement with `inner` and `outer` options or the `vlan-id` statement at the `[edit interfaces ge-fpc/pic/port unit logical-unit-number]` hierarchy level:

```
[edit interfaces ge-fpc/pic/port unit logival-unit-number]
user@host# vlan-id number;
family family {
    address address;
}
user@host# vlan-tags inner tpid.vlan-id outer tpid.vlan-id;
family family {
    address address;
}
```

NOTE: If you configure the physical interface MTU for mixed tagging, then you must increase the MTU to 4 bytes more than the MTU value you would configure for a standard VLAN-tagged interface.

For example, if the MTU value is configured to be 1018 on a VLAN-tagged interface, then the MTU value on a flexible VLAN tagged interface must be 1022—4 bytes more. The additional 4 bytes accommodates the future addition of a stacked VLAN tag configuration on the same physical interface.

If the same physical interface MTU value is configured on both the VLAN and flexible VLAN-tag routers, the L2 circuit configuration does not come up and a MTU mismatch is logged. However, normal traffic flow is unaffected.

For encapsulation type `flexible-ethernet-services`, all VLAN IDs are valid.

RELATED DOCUMENTATION

802.1Q VLANs Overview | 274

Configuring VLAN and Extended VLAN Encapsulation | 299

Stacking a VLAN Tag | 399

Ethernet Interfaces User Guide for Routing Devices

Sending Untagged Traffic Without VLAN ID to Remote End | 280

Configuring Tagged Interface with multiple tagged vlans and native vlan

You can configure the router to receive and forward single-tag frames, dual-tag frames, or a mixture of single-tag and dual-tag frames.

1. To configure the router to receive and forward single-tag frames with 802.1Q VLAN tags, include the `vlan-tagging` statement at the `[edit interfaces interface-name]` hierarchy level:

```
[edit interfaces interface-name]
user@host# vlan-tagging;
```

2. Configure the `flexible-vlan-tagging` statement at the `[edit interfaces ge-fpc/pic/port]` hierarchy level.

```
[edit interfaces ge-fpc/pic/port]
user@host#flexible-vlan-tagging;
```

   For encapsulation type `flexible-ethernet-services`, all VLAN IDs are valid.

3. To accept untagged packets, include the `native-vlan-id` statement and the `flexible-vlan-tagging` statement at the `[edit interfaces interface-name]` hierarchy level:

```
[edit interfaces ge-fpc/pic/port]
flexible-vlan-tagging;
native-vlan-id number;
```

   The range for `native-vlan-id` is 0 to 4094.

   The logical interface on which untagged packets are to be received must be configured with the same native VLAN ID as that configured on the physical interface. To configure the logical interface, include the `vlan-id` statement (matching the `native-vlan-id` statement on the physical interface) at the `[edit interfaces interface-name unit logical-unit-number]` hierarchy level.

4. Configure the `vlan-id range` providing the range at the `[edit interfaces ge-fpc/pic/port unit logical-unit-number]` hierarchy level:

```
[edit interfaces ge-fpc/pic/port unit logical-unit-number]
user@host# vlan-id range number;
```

5. Configure the `vlan-tags` statement with `inner` and `outer` options or the `vlan-id range` statement at the `[edit interfaces ge-fpc/pic/port unit logical-unit-number]` hierarchy level:

```
[edit interfaces ge-fpc/pic/port unit logical-unit-number]
user@host# vlan-id range number;
user@host# vlan-tags outer tpid.vlan-id inner tpid.vlan-id;
```

   The range for `inner` and `outer` option 32 to 4094.

To verify the configuration execute the `show` command.

```
user@host> show configuration

set interfaces ge-1/0/3 flexible-vlan-tagging
set interfaces ge-1/0/3 native-vlan-id 1010
set interfaces ge-1/0/3 unit 1 vlan-id-range 100-200
set interfaces ge-1/0/3 unit 2 vlan-tags outer 300
set interfaces ge-1/0/3 unit 2 vlan-tags inner 123
set interfaces ge-1/0/3 unit 3 vlan-tags outer 400
set interfaces ge-1/0/3 unit 3 vlan-tags inner 323
```

RELATED DOCUMENTATION

- 802.1Q VLANs Overview | 274
- Configuring VLAN and Extended VLAN Encapsulation | 299
- Stacking a VLAN Tag | 399
- Ethernet Interfaces User Guide for Routing Devices
- Sending Untagged Traffic Without VLAN ID to Remote End | 280

Sending Untagged Traffic Without VLAN ID to Remote End

Send traffic without the native VLAN ID (*native-vlan-id*) to the remote end of the network if untagged traffic is received.

If this option is not configured, then *native-vlan-id* is added to untagged traffic. But if this option is configured, then *native-vlan-id* is not added to untagged traffic.

NOTE:

- Configuring this option with DPC results in no behavior change. But, if this option is configured with Aggregated Ethernet (AE) in which the sub interfaces reside across MPCs/MICs and DPC, MPCs/MICs and DPC will show a different behavior. Use Feature Explorer to confirm platform and release support for specific features.
- In the egress direction, this feature is disrupted by VLAN normalization. Because of normalization, the egress interface cannot distinguish between untagged traffic and tagged traffic. And untagged traffic is sent out with *native-vlan-id*. Consider this while configuring both VLAN normalization and new *native-vlan-id* option.

  There will be a problem with ingress firewall filter if filter term includes *native-vlan-id*. With `no-native-vlan-insert` option configured, *native-vlan-id* will not be inserted to untagged traffic. So, firewall filter term will not match with untagged traffic. But if incoming traffic have VLAN ID which is equal to *native-vlan-id*, then firewall filter term will match and firewall will work.
- When this feature is used with AE, all sub-interfaces of AE should be in same type of FPC.

RELATED DOCUMENTATION

802.1Q VLANs Overview | **274**

Configuring VPLS Interface Encapsulation

native-vlan-id

no-native-vlan-insert

Enabling VLAN Tagging | **277**

Configuring Tag Protocol IDs (TPIDs) on QFX Series Switches

IN THIS SECTION

This topic describes how to configure the TPIDs expected to be sent or received on a particular VLAN for QFX series switches.

The QFX5100, QFX5110, QFX5120, QFX5200, and QFX5210 devices provide support for up to 4 TPID configurations per VLAN on an interface device. A TPID needs to be defined in the command-line interface (CLI) for aggregated ethernet interfaces and ether interfaces for the QFX device before it can be sent or received. The following values are supported for the TPID configuration:

- 0x8100
- 0x9200
- 0x88a8
- 0x9100

When a TPID is configured for a VLAN on a logical interface, the traffic packets that ingress on the port with TPIDs 0x8100, 0x88a8, 0x9100, and 0x9200 are accepted, and the packets that egress from the port for a particular VLAN carry the packets with the TPID configured on that VLAN.

To configure the TPID values, use the `tag-protocol-id` statement. See, *tag-protocol-id TPID (TPIDs Expected to Be Sent or Received)*.

The TPID values should be configured at the device interface level to be able to implement TPID support. The `vlan-tags outer` statement can be used for configuring a TPID per VLAN. See, *vlan-tags*.

Conditions for Configuring Tag Protocol IDs (TPIDs) on QFX Series Switches

Note the following conditions before configuring TPIDs on QFX series switches:

1. The TPID configured for the VLAN profile for the logical interface takes precedence over the TPID configured for the port.

   In the following example, it is assumed that the traffic that egress from unit 200 will have the TPID 0x9200 as that is the TPID set at the port level. However, since TPIDs are configured for logical interfaces, the traffic takes the value of the TPID from the VLAN profile rather than the port profile for all the logical interfaces configured on the interface device. Therefore, though `ae1` is set with TPID 0x9200 at the port level, packets that egress out of `ae1.200` will have the VLAN's default TPID 0x8100. The same concept applies to `ae1.400` as well.

```
xe-0/0/0 {
                 gigether-options {
        802.3ad ae1;
                  }
              }
         ae1 {
                 flexible-vlan-tagging;
                  encapsulation flexible-ethernet-services;
                  aggregated-ether-options {
                    ethernet-switch-profile {
                    tag-protocol-id 0x9200;
           }
                }
             unit 100 {
                     encapsulation vlan-bridge;
                     vlan-tags outer 0x9200.100;
              }
              unit 200 {
                     encapsulation vlan-bridge;
                  vlan-id 200;
              }
    unit 400 {
             family ethernet-switching {
                 interface-mode trunk;
                 vlan {
                         members 400;
                 }
           }
}
```

2. VXLAN scenarios are **not** supported.

3. If two logical interfaces with different TPIDs are a part of a VLAN, then the most recently configured TPID value is configured in the VLAN profile, as the hardware register supports only one TPID index per VLAN profile. In the example below, the TPID value for VLAN 100 is 0x9200 as `ae2.100` is the last configured value.

```
ae1 {
                flexible-vlan-tagging;
                 encapsulation flexible-ethernet-services;
                 aggregated-ether-options {
        ethernet-switch-profile {
                    tag-protocol-id 0x9100;
                     }
              }
             unit 100 {
                     encapsulation vlan-bridge;
                     vlan-tags outer 0x9100.100;
              }

         ae2 {
                 flexible-vlan-tagging;
                  encapsulation flexible-ethernet-services;
                  aggregated-ether-options {
                    ethernet-switch-profile {
                    tag-protocol-id 0x9200;
                     }
              }
             unit 100 {
                     encapsulation vlan-bridge;
                     vlan-tags outer 0x9200.100;
              }
vlans {
    v_100 {
        vlan-id 100;
        interface ae1.100;
        interface ae2.100;
    }
```

4. If multiple TPIDs are configured under the interface device with no logical interface level setting, then the first TPID value in the list is set at the port level.

In the example below, 0x9200 is set at the port level.

```
xe-0/0/0 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    ether-options {
        ethernet-switch-profile {
            tag-protocol-id [0x9200 0x88a8 0x9100];
        }
    }
}
```

5. If only a single logical interface is configured on an interface device with 0x8100, then the packets egress with the TPID value configured at the port level. However, as shown in the example below, if multiple logical interfaces are configured with different TPIDs from the list at the interface device level, then packets going out with VLAN 100 will egress with the TPID 0x8100.

```
xe-0/0/0 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    ether-options {
        ethernet-switch-profile {
            tag-protocol-id [0x9200 0x88a8 0x8100];
        }
    }
}
            unit 100 {
                encapsulation vlan-bridge;
                vlan-tags outer 0x8100.100;
            }
```

RELATED DOCUMENTATION

Configuring Frames with Particular TPIDs to Be Processed as Tagged Frames | 389

Configuring Flexible VLAN Tagging on PTX Series Packet Transport Routers

This topic describes how to configure flexible VLAN tagging on PTX Series Packet Transport Routers. In addition to VLAN tagging and stacked VLAN tagging, you can configure a port for flexible tagging. With flexible VLAN tagging, you can configure two logical interfaces on the same Ethernet port, one with single-tag framing and one with dual-tag framing.

To configure mixed tagging, include the `flexible-vlan-tagging` statement at the `[edit interfaces et-`*`fpc/pic/port`*` ]` hierarchy level. You must also include the `vlan-tags` statement with `inner` and `outer` options or the `vlan-id` statement at the `[edit interfaces et-`*`fpc/pic/port`*` unit `*`logical-unit-number`*`]` hierarchy level:

```
[edit interfaces et-fpc/pic/port]
flexible-vlan-tagging;
unit logical-unit-number {
    vlan-id number;
}
unit logical-unit-number {
    vlan-tags inner tpid.vlan-id outer tpid.vlan-id;
}
```

NOTE: For PTX EVO platforms, if you have configured an Interface Device (IFD) with the family ethernet switching vlan members configuration, you cannot use both VLAN tagging and flexible VLAN tagging CLI commands on the IFD at the same time. This configuration is not supported, and a warning is issued if you try to commit this configuration.

RELATED DOCUMENTATION

Enabling VLAN Tagging | 277

Configuring an MPLS-Based VLAN CCC with Pop, Push, and Swap and Control Passthrough

IN THIS SECTION

Overview

For providing Layer 2 VPN services across your network, you might want to configure the ability to push, pop or swap 802.1Q tags on frames entering and leaving edge routers, allowing you to use a single VLAN-circuit cross-connect (CCC) [VLAN-CCC] logical interface to handle both dual-tag and single-tag packets. This feature thus provides interoperability between Layer 2 services with a distinct VLAN at the local or remote end or in instances where a Layer 2 service comes with a certain VLAN, but the remote peer has a different VLAN or no VLAN.

This feature includes the ability to enable passthrough of certain Ethertype/DMAC-matched frames over the Layer 2 circuit after successful VLAN tag operations on the VLAN CCC logical interface.

If you configure this feature, VLAN tags are applied when traffic is sent to and from the Layer 2 circuit interface. The pop, push, and swap operations are performed only on the outer tag. The pop VLAN tag removes the VLAN tag from the top of the VLAN tag stack. The push VLAN tag adds a new outer VLAN tag, and the swap VLAN tag replaces the existing outer VLAN tag with the new VLAN tag.

You can configure inet, inet6, or VLAN-CCC connections on a single Ethernet network interface or an aggregated Ethernet interface, enabling you to set different forwarding rules for tagged and untagged traffic on the same interface. For example, you can forward tagged packets over the Layer 2 circuit and route untagged traffic in native VLAN mode.

Use Feature Explorer to confirm platform and release support for specific features.

Review the "Platform-Specific MPLS-Based VLAN CCC Behavior" on page 288 section for notes related to your platform.

Platform-Specific MPLS-Based VLAN CCC Behavior

Use Feature Explorer to confirm platform and release support for specific features.

Use the following table to review platform-specific behavior for your platforms.

Platform	Difference
ACX Series (Junos OS Evolved)	• The `l2circuit-control-passthrough` statement at the `[edit forwarding-options]` hierarchy level is not applicable.
PTX Series (Junos OS and Junos OS Evolved)	• VLAN operations on STP and CDP packets are not supported. • You can't configure the VLAN-CCC logical interface with the native VLAN ID. • LACP point-to-point connections between PE routers do not work if you configure `l2circuit-control-passthrough`. (Static LAG works, however.)

Procedure

To configure a PE router with a VLAN CCC, an MPLS-based Layer 2 circuit, VLAN pop, push, and swap operations, and enabling passthrough of certain Ethertype/DMAC-matched frames:

NOTE: The following procedure uses actual interface names for the router's network interfaces instead of the variable *interface-name* so that you can quickly see their configuration differences. Remember that you can also configure the feature on aggregated Ethernet interfaces.

1. Configure OSPF on the loopback (or router address) and core interface:

NOTE: The routing protocol can be OSPF or IS-IS.

```
[edit]
user@host# set protocols ospf area 0.0.0.0 interface lo0.0 passive
user@host# set protocols ospf area 0.0.0.0 interface et-0/0/0:0
```

2. Enable traffic engineering for the routing protocol:

```
[edit]
user@host# set protocols ospf traffic-engineering
```

3. Configure an IP address for the loopback interface and for the core interface:

```
[edit]
user@host# set interfaces lo0 unit logical-unit-number family inet address address
user@host# set interfaces et-0/0/0:0 unit 0 family inet address address
```

4. Configure the customer edge interface as a Layer 2 circuit from the local PE router to the other PE router:

TIP: Use the router address of the other router as the neighbor address. It is the virtual circuit identifier together with the neighbor address that provides the unique address for the circuit.

```
[edit]
user@host# set protocols l2circuit neighbor address interface et-0/0/1:1.0 virtual-circuit-
id identifier
```

5. Configure MPLS on the core interfaces:

```
[edit]
user@host# set protocols mpls interface all
```

6. Configure LDP on the loopback interface and the core interfaces:

```
[edit]
user@host# set ldp interface lo0.0
user@host# set ldp interface et-0/0/0.0
user@host# set ldp interface all
```

7. Configure `family mpls` on the logical unit of the core interface:

```
[edit]
user@host# set interfaces et-0/0/0:0 unit 0 family mpls
```

NOTE: You can enable `family mpls` on either individual interfaces, aggregated Ethernet interfaces, or tagged VLAN interfaces.

8. Specify the router ID:

```
[edit]
user@host# set routing-options router-id address
```

9. Enable VLAN tagging on the customer edge interface of the local PE router:

```
[edit]
user@host# set interfaces et-0/0/1:1 vlan-tagging
```

10. Configure the customer edge interface to use flexible Ethernet services encapsulation:

```
[edit]
user@host# set interfaces et-0/0/1:1 encapsulation flexible-ethernet-services
```

11. Configure the logical unit of the customer edge interface with a VLAN ID:

```
[edit]
user@host# set interfaces et-0/0/1:1 unit 0 vlan-id vlan-id
```

12. Configure the logical unit on the customer edge interface to use VLAN CCC encapsulation:

```
[edit]
user@host# set interfaces et-0/0/1:1 unit 0 encapsulation vlan-ccc
```

13. Configure the logical unit on the customer edge interface to pop the tag off the input VLAN and then push the tag to the output VLAN:

```
[edit]
user@host# set interfaces et-0/0/1:1 unit 0 input-vlan-map push
[edit]
user@host# set interfaces et-0/0/1:1 unit 0 output-vlan-map pop
```

14. (Optional) Configure Layer 2 circuit traceoptions:

```
[edit]
user@host# set protocols l2circuit traceoptions file l2ckt.log
user@host# set protocols l2circuit traceoptions flag connections detail
```

15. (Optional) Configure the VLAN CCC logical interface so that encapsulation mismatches and MTU mismatches between this interface and the interface on the other PE router are ignored:

```
[edit]
user@host# set protocolsl2circuit neighbor address interface et-0/0/1:1.0  ignore-
encapsulation-mismatch
user@host# set protocolsl2circuit neighbor address interface et-0/0/1:1.0 ignore-mtu-
mismatch
```

16. On applicable platforms, to enable passthrough of certain Ethertype/DMAC-matched frames, configure Layer 2 circuit control passthrough:

```
[edit]
user@host# set forwarding-options l2circuit-control-passthrough
```

RELATED DOCUMENTATION

CCC Overview

Binding VLAN IDs to Logical Interfaces

This topic describes how to configure logical interfaces to receive and forward VLAN-tagged frames:

To configure a logical interface to receive and forward VLAN-tagged frames, you must bind a VLAN ID, a range of VLAN IDs, or a list of VLAN IDs to the logical interface. Table 57 on page 292 lists the configuration statements you use to bind VLAN IDs to logical interfaces, organized by scope of the VLAN IDs used to match incoming packets. You can configure these statements at the [edit interfaces *interface-name* unit *logical-unit-number*] hierarchy level or at the [edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number*] hierarchy level.

Table 57: Configuration Statements Used to Bind VLAN IDs to Logical Interfaces

Scope of VLAN ID Matching	Type of VLAN Framing Supported on the Logical Interface	
	Single-Tag Framing	**Dual-Tag Framing**
VLAN ID	vlan-id *vlan-id*;	vlan-tags outer *tpid*.<*vlan-id*> inner *tpidvlan-id*;
VLAN ID Range	vlan-id-range *vlan-id*-*vlan-id*;	vlan-tags outer *tpid*.*vlan-id* inner-range *tpid*.*vlan-id*-*vlan-id*;
VLAN ID List	vlan-id-list [*vlan-id vlan-id*-*vlan-id*];	vlan-tags outer <*tpid*.>*vlan-id* inner-list [*vlan-id vlan-id*-*vlan-id*];

NOTE: The inner-list option of the vlan-tags statement does not support Tag Protocol ID (TPID) values.

1. A logical interface that you have associated (bound) to a particular VLAN ID will receive and forward incoming frames that contain a matching VLAN ID. To bind a VLAN ID to a single-tag logical interface, include the vlan-id statement at the [edit interfaces *interface-name* unit *logical-unit-number*] hierarchy level or at the [edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number*] hierarchy level.

```
[edit interfaces interface-name unit logical-unit-number]
user@host# vlan-id vlan-id;
```

To configure an Ethernet interface to support single-tag logical interfaces, include the `vlan-tagging` statement at the [`edit interfaces` *`ethernet-interface-name`*] hierarchy level. To support mixed tagging, include the `flexible-vlan-tagging` statement instead.

2. To bind a VLAN ID to a dual-tag logical interface, include the `vlan-tags` statement at the [`edit interfaces` *`ethernet-interface-name`* `unit` *`logical-unit-number`*] hierarchy level or at the [`edit logical-systems` *`logical-system-name`* `interfaces` *`ethernet-interface-name`* `unit` *`logical-unit-number`*] hierarchy level:

   ```
   [edit interfaces ethernet-interface-name unit logical-unit-number]
   vlan-tags inner <tpid.>vlan-id outer <tpid.>vlan-id;
   ```

   To configure an Ethernet interface to support dual-tag logical interfaces, include the `stacked-vlan-tagging` statement at the [`edit interfaces` *`ethernet-interface-name`*] hierarchy level. To support mixed tagging, include the `flexible-vlan-tagging` statement instead.

3. A VLAN range can be used by service providers to interconnect multiple VLANs belonging to a particular customer over multiple sites. Using a VLAN ID range conserves switch resources and simplifies configuration. To bind a range of VLAN IDs to a single-tag logical interface, include the `vlan-id-range` statement at the [`edit interfaces` *`ethernet-interface-name`* `unit` *`logical-unit-number`*] hierarchy level or at the [`edit logical-systems` *`logical-system-name`* `interfaces` *`ethernet-interface-name`* `unit` *`logical-unit-number`*] hierarchy level.

   ```
   [edit interfaces ethernet-interface-name unit logical-unit-number]
   vlan-id-range vlan-id-vlan-id;
   ```

4. To bind a range of VLAN IDs to a dual-tag logical interface, include the `vlan-tags` statement. Use the `inner-list` option to specify the VLAN IDs as an inclusive range by separating the starting VLAN ID and ending VLAN ID with a hyphen. You can include the statement at the [`edit interfaces` *`ethernet-interface-name`* `unit` *`logical-unit-number`*] hierarchy level or at the [`edit logical-systems` *`logical-system-name`* `interfaces` *`ethernet-interface-name`* `unit` *`logical-unit-number`*] hierarchy level.

   ```
   [edit interfaces ethernet-interface-name unit logical-unit-number]
   vlan-tags inner-list [ vlan-id vlan-id-vlan-id ] outer <tpid.>vlan-id;
   ```

   To configure an Ethernet interface to support dual-tag logical interfaces, include the `stacked-vlan-tagging` statement at the [`edit interfaces` *`ethernet-interface-name`*] hierarchy level. To support mixed tagging, include the `flexible-vlan-tagging` statement instead.

You can use VLAN-bundled logical interfaces to configure circuit cross-connects between Layer 2 VPN routing instances or Layer 2 circuits. Using VLAN-bundled logical interfaces simplifies configuration and reduces use of system resources such as logical interfaces, next hops, and circuits.

As an alternative to configuring multiple logical interfaces (one for each VLAN ID and one for each range of VLAN IDs), you can configure a single VLAN-bundled logical interface based on a list of VLAN IDs.

NOTE: The `vlan-id` option is not supported to achieve VLAN normalization on VPLS instances that are configured with `vlan-id-list`. However, you can use the `vlan-maps` option to achieve VLAN normalization.

1. To bind a list of VLAN IDs to a single-tag logical interface, include the `vlan-id-list` statement at the `[edit interfaces` *`ethernet-interface-name`* `unit` *`logical-unit-number`*`]` hierarchy level or at the `[edit logical-systems` *`logical-system-name`* `interfaces` *`ethernet-interface-name`* `unit` *`logical-unit-number`*`]` hierarchy level. Specify the VLAN IDs in the list individually by using a space to separate each ID, as an inclusive list by separating the starting VLAN ID and ending VLAN ID with a hyphen, or as a combination of both.

   ```
   [edit interfaces ethernet-interface-name unit logical-unit-number]
   user@host# vlan-id-list [ vlan-id vlan-id-vlan-id ];
   ```

   To configure an Ethernet interface to support single-tag logical interfaces, include the `vlan-tagging` statement at the `[edit interfaces` *`ethernet-interface-name`*`]` hierarchy level. To support mixed tagging, include the `flexible-vlan-tagging` statement instead.

2. To bind a list of VLAN IDs to a dual-tag logical interface, include the `vlan-tags` statement at the `[edit interfaces` *`ethernet-interface-name`* `unit` *`logical-unit-number`*`]` hierarchy level or at the `[edit logical-systems` *`logical-system-name`* `interfaces` *`ethernet-interface-name`* `unit` *`logical-unit-number`*`]` hierarchy level. Use the `inner-list` option to specify the VLAN IDs individually by using a space to separate each ID, as an inclusive list by separating the starting VLAN ID and ending VLAN ID with a hyphen, or as a combination of both.

   ```
   [edit interfaces ethernet-interface-name unit logical-unit-number]
   user@host# vlan-tags inner-list [vlan-id vlan-id-vlan-id ] outer <tpid>vlan-id;
   ```

   NOTE: The `inner-list` option of the `vlan-tags` statement does not support Tag Protocol ID (TPID) values.

   To configure an Ethernet interface to support dual-tag logical interfaces, include the `stacked-vlan-tagging` statement at the `[edit interfaces` *`ethernet-interface-name`*`]` hierarchy level. To support mixed tagging, include the `flexible-vlan-tagging` statement instead.

The following sample configuration configures two different lists of VLAN IDs on two different logical ports.

```
[edit interfaces]
ge-1/1/0 {
    vlan-tagging; # Only for single-tagging
    encapsulation flexible-ethernet-services;
    unit 10 {
        encapsulation vlan-ccc;
        vlan-id-list [20 30-40 45];
    }
}
ge-1/1/1 {
    flexible-vlan-tagging; # Only for mixed tagging
    encapsulation flexible-ethernet-services;
    unit 10 {
        encapsulation vlan-ccc;
        vlan-id-list [1 10 20 30-40];
    }
    unit 20 {
        encapsulation vlan-ccc;
        vlan-tags outer 200 inner-list [50-60 80 90-100];
    }
}
```

In the example configuration above, `ge-1/1/0` supports single-tag logical interfaces, and `ge-1/1/1` supports mixed tagging. The single-tag logical interfaces `ge-1/1/0.10` and `ge-1/1/1.10` each bundle lists of VLAN IDs. The dual-tag logical interface `ge-1/1/1.20` bundles lists of inner VLAN IDs.

TIP: You can group a range of identical interfaces into an interface range and then apply a common configuration to that interface range. For example, in the above example configuration, both interfaces ge-1/1/0 and ge-1/1/1 have the same physical encapsulation type of `flexible-ethernet-services`. Thus you can define an interface range with the interfaces ge-1/1/0 and ge-1/1/1 as its members and apply the encapsulation type flexible-ethernet-services to that defined interface range.

Platform-Specific Behavior

Use https://apps.juniper.net/feature-explorer/home.html to confirm platform and release support for specific features.

Use the following table to review platform-specific behaviors for your platforms.

Platform	Difference
MX Series Routers, EX Series Routers	Bind a list of VLAN IDs to a single logical interface, eliminating the need to configure a separate logical interface for every VLAN or VLAN range. A logical interface that accepts packets tagged with any VLAN ID specified in a VLAN ID list is called a *VLAN-bundled* logical interface.

RELATED DOCUMENTATION

- 802.1Q VLANs Overview | **274**
- Interface Ranges for Physical Interfaces
- Ethernet Interfaces User Guide for Routing Devices

Associating VLAN IDs to VLAN Demux Interfaces

IN THIS SECTION

The following sections describe how to configure VLAN demux interfaces to receive and forward VLAN-tagged frames:

Associating VLAN IDs to VLAN Demux Interfaces Overview

To configure a VLAN demux interface to receive and forward VLAN-tagged frames, you must associate a VLAN ID or dual tagged (stacked) VLAN ID to the interface. Table 58 on page 297 shows the

configuration statements you use to associate VLAN IDs to VLAN demux interfaces, depending on the VLAN tag framing you use:

NOTE: The demux logical interface cannot be the same as its underlying logical interface.

Table 58: Configuration Statements Used to Associate VLAN IDs to VLAN Demux Interfaces

	Single-Tag Framing	Dual-Tag Framing
Statement Format	vlan-id *vlan-id*;	vlan-tags outer *tpid*.<*vlan-id*> inner *tpidvlan-id*;

You can include all of the statements at the following hierarchy levels:

- [edit interfaces *interface-name* unit *logical-unit-number*]
- [edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number*]
- [edit interfaces *demux0* unit *logical-unit-number*]

Associating a VLAN ID to a VLAN Demux Interface

IN THIS SECTION

A VLAN demux interface that you have associated to a particular VLAN ID receives and forwards incoming frames that contain a matching VLAN ID. You can associate a VLAN ID to a single-tag logical interface or to a dual-tagged (stacked) logical interface.

Associating a VLAN ID to a Single-Tag VLAN Demux Interface

To associate a VLAN ID to a single-tag VLAN demux interface, include the `vlan-id` statement at the `[edit interfaces demux0 unit logical-unit-number]` hierarchy level:

```
vlan-id vlan-id;
```

To configure an interface to support single-tag logical interfaces, you must also include the `vlan-tagging` statement at the `[edit interfaces interface-name]` hierarchy level. To support mixed tagging, include the `flexible-vlan-tagging` statement instead.

SEE ALSO

> *Configuring a VLAN Demultiplexing Interface*

Associating a VLAN ID to a Dual-Tag VLAN Demux Interface

To associate a VLAN ID to a dual-tag VLAN demux interface, include the `vlan-tags` statement at the `[edit interfaces demux0 unit logical-unit-number]` hierarchy level:

```
vlan-tags inner <tpid.>vlan-id outer <tpid.>vlan-id;
```

To configure an interface to support dual-tag logical interfaces, include the `stacked-vlan-tagging` statement at the `[edit interfaces interface-name]` hierarchy level. To support mixed tagging, include the `flexible-vlan-tagging` statement instead.

SEE ALSO

> 802.1Q VLANs Overview | **274**
>
> *Configuring a VLAN Demultiplexing Interface*
>
> Ethernet Interfaces User Guide for Routing Devices

Configuring VLAN and Extended VLAN Encapsulation

IN THIS SECTION

To configure encapsulation on an interface, enter the `encapsulation` statement at the `[edit interfaces` *`interface-name`*`]` hierarchy level:

```
[edit interfaces interface-name]
user@host# encapsulation type
```

The following list contains important notes regarding VLAN encapsulation:

- Ethernet interfaces in VLAN mode can have multiple logical interfaces. In CCC and VPLS modes, VLAN IDs from 1 through 511 are reserved for normal VLANs, and VLAN IDs 512 through 4094 are reserved for CCC or VPLS VLANs. For encapsulation type `flexible-ethernet-services`, all VLAN IDs are valid.
- For flexible Ethernet services, Ethernet VLAN CCC and VLAN VPLS, you can also configure the encapsulation type that is used inside the VLAN circuit itself. To do this, include the `encapsulation` statement at the `[edit interfaces` *`interface-name`* `unit` *`logical-unit-number`*`]` hierarchy level or at the `[edit logical-systems` *`logical-system-name`* `interfaces` *`interface-name`* `unit` *`logical-unit-number`*`]` hierarchy level.
- You cannot configure a logical interface with VLAN CCC or VLAN VPLS encapsulation unless you also configure the physical device with the same encapsulation or with flexible Ethernet services encapsulation. In general, the logical interface must have a VLAN ID of 512 or higher; if the VLAN ID is 511 or lower, it will be subject to the normal destination filter lookups in addition to source address filtering. However if you configure flexible Ethernet services encapsulation, this VLAN ID restriction is removed.
- Gigabit Ethernet, MX Series router Gigabit Ethernet, Tri-Rate Ethernet copper, 10-Gigabit Ethernet, and aggregated Ethernet interfaces with VLAN tagging enabled can use `extended-vlan-ccc` or `extended-vlan-vpls`, which allow 802.1Q tagging.
- For extended VLAN CCC and extended VLAN VPLS encapsulation, all VLAN IDs 1 and higher are valid. VLAN ID 0 is reserved for tagging the priority of frames.

- For extended VLAN CCC, the VLAN IDs on ingress and egress interfaces must be the same. For back-to-back connections, all VLAN IDs must be the same.

> **NOTE**: For EVO platforms, if you have configured any Logical Interface (IFL) on an Interface Device (IFD) with the family ethernet-switching configuration, you cannot configure any other families on a different IFL unless you configure the IFD with the flexible-ethernet-services encapsulation type. This configuration is not supported, and a warning is issued if you try to commit this configuration.

Platform-Specific VLAN and Extended VLAN Encapsulation Behavior

Table 59: Platform-Specific Behavior

Platform	Difference
Junos Evolved ACX7000 series	Flexible Ethernet Services encapsulation is required when mixing interface families (e.g., `ethernet-switching` with others) on the same physical interface. Extended VLAN encapsulation (`extended-vlan-ccc`, `extended-vlan-vpls`) supports all VLAN IDs ≥ 1; VLAN ID 0 is reserved for priority tagging. For extended VLAN CCC, ingress and egress VLAN IDs must match. For back-to-back connections, VLAN IDs must be identical on both ends. Logical interfaces using `vlan-ccc` or `vlan-vpls` must match the encapsulation type of the physical interface or use `flexible-ethernet-services`. VLAN IDs 1–511 are reserved for normal VLANs; 512–4094 are used for CCC/VPLS unless `flexible-ethernet-services` is configured, which removes this restriction.

Change History Table

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

Release	Description
9.5	Starting with Junos OS Release 9.5, aggregated Ethernet interfaces configured for VPLS can use `flexible-ethernet-services`, `vlan-ccc`, or `vlan-vpls`.
8.1	Starting with Junos OS Release 8.1, Gigabit Ethernet IQ, Gigabit Ethernet PICs with small form-factor pluggable optics (SFPs), and MX Series router Gigabit Ethernet, Tri-Rate Ethernet copper, and 10-Gigabit Ethernet interfaces with VLAN tagging enabled can use `flexible-ethernet-services`, `vlan-ccc` , or `vlan-vpls` encapsulation.

RELATED DOCUMENTATION

802.1Q VLANs Overview | **274**

Configuring VPLS Interface Encapsulation

Ethernet Interfaces User Guide for Routing Devices

Configuring a Layer 2 VPN Routing Instance on a VLAN-Bundled Logical Interface

IN THIS SECTION

This topic describes how to configure a Layer 2 VPN routing instance on a logical interface bound to a list of VLAN IDs.

Configuring a VLAN-Bundled Logical Interface to Support a Layer 2 VPN Routing Instance

To configure a VLAN-bundled logical interface, specify the list of VLAN IDs by including the `vlan-id-list` statement or the `vlan-tags` statement on a provider edge (PE) router:

```
interfaces {
    ethernet-interface-name {
        vlan-tagging; # Support single- or dual-tag logical interfaces
        flexible-vlan-tagging; # Support mixed tagging
        encapsulation (extended-vlan-ccc | flexible-ethernet-services);
        unit logical-unit-number {
            vlan-id-list [vlan-id vlan-id-vlan-id]; # For single-tag
            vlan-tags outer <tpid.>vlan-id inner-list [vlan-id vlan-id-vlan-id]; # For dual-tag
        }
        . . .
    }
}
```

You can include the statements at the following hierarchy levels:

- `[edit]`
- `[edit logical-systems logical-system-name]`

SEE ALSO

802.1Q VLANs Overview | 274

Ethernet Interfaces User Guide for Routing Devices

Specifying the Interface Over Which VPN Traffic Travels to the CE Router

To configure a Layer 2 VPN routing instance on a PE router, include the `instance-type` statement and specify the value `l2vpn`. To specify an interface connected to the router, include the `interface` statement and specify the VLAN-bundled logical interface:

```
instance-type l2vpn;
interface logical-interface-name;
```

You can include the statements at the following hierarchy levels:

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

SEE ALSO

802.1Q VLANs Overview | 274

Ethernet Interfaces User Guide for Routing Devices

Specifying the Interface to Handle Traffic for a CCC

To configure the VLAN-bundled logical interface as the interface to handle traffic for a circuit connected to the Layer 2 VPN routing instance, include the following statements:

```
protocols {
    l2vpn {
        (control-word | no-control-word);
        encapsulation-type (ethernet | ethernet-vlan);
        site site-name {
            site-identifier identifier;
            interface logical-interface-name { # VLAN-bundled logical interface
                . . . interface-options . . .
            }
        }
    }
}
```

You can include the statements at the same hierarchy level at which you include the `instance-type l2vpn` and `interface` *`logical-interface-name`* statements:

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

To enable a Layer 2 VPN routing instance on a PE router, include the `l2vpn` statement. For more information, see the Junos OS VPNs Library for Routing Devices.

The `encapsulation-type` statement specifies the Layer 2 protocol used for traffic from the customer edge (CE) router. If the Layer 2 VPN routing instance is being connected to a single-tag Layer 2 circuit, specify `ethernet` as the encapsulation type. If the Layer 2 VPN routing instance is being connected to a dual-tag Layer 2 circuit, specify `ethernet-vlan` as the encapsulation type.

To specify the interface to handle traffic for a circuit connected to the Layer 2 VPN routing instance, include the `interface` statement and specify the VLAN-bundled logical interface.

SEE ALSO

802.1Q VLANs Overview | **274**

Example: Configuring a Layer 2 VPN Routing Instance on a VLAN-Bundled Logical Interface | **304**

Ethernet Interfaces User Guide for Routing Devices

Example: Configuring a Layer 2 VPN Routing Instance on a VLAN-Bundled Logical Interface

The following configuration shows that the single-tag logical interface `ge-1/0/5.0` bundles a list of VLAN IDs, and the logical interface `ge-1/1/1.0` supports IPv4 traffic using IP address 10.30.1.130 and can participate in an MPLS path.

```
[edit interfaces]
ge-1/0/5 {
    vlan-tagging;
    encapsulation extended-vlan-ccc;
    unit 0 { # VLAN-bundled logical interface
        vlan-id-list [513 516 520-525];
    }
```

```
}
    ge-1/1/1 {
        unit 0 {
            family inet {
                address 10.30.1.1/30;
            }
            family mpls;
        }
    }
```

The following configuration shows the type of traffic supported on the Layer 2 VPN routing instance:

```
[edit protocols]
rsvp {
    interface all;
    interface lo0.0;
}
mpls {
    label-switched-path lsp {
        to 10.255.69.128;
    }
    interface all;
}
bgp {
    group g1 {
        type internal;
        local-address 10.255.69.96;
        family l2vpn {
            signaling;
        }
        neighbor 10.255.69.128;
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface lo0.0;
        interface ge-1/1/1.0;
    }
}
```

The following configuration shows that the VLAN-bundled logical interface is the interface over which VPN traffic travels to the CE router and handles traffic for a CCC to which the VPN connects.

```
[edit routing-instances]
red {
    instance-type l2vpn;
    interface ge-1/0/5.0; # VLAN-bundled logical interface
    route-distinguisher 10.255.69.96:100;
    vrf-target target:1:1;
    protocols {
        l2vpn {
            encapsulation-type ethernet; # For single-tag VLAN logical interface
            site CE_ultima {
                site-identifier 1;
                interface ge-1/0/5.0;
            }
        }
    }
}
```

NOTE: Because the VLAN-bundled logical interface supports single-tag frames, Ethernet is the Layer 2 protocol used to encapsulate incoming traffic. Although the connection spans multiple VLANs, the VLANs are bundled and therefore can be encapsulated as a single VLAN.

However, with Ethernet encapsulation, the circuit signal processing does not check that the VLAN ID list is the same at both ends of the CCC connection.

RELATED DOCUMENTATION

802.1Q VLANs Overview | **274**

Ethernet Interfaces User Guide for Routing Devices

Specifying the Interface Over Which VPN Traffic Travels to the CE Router

To configure a Layer 2 VPN routing instance on a PE router, include the `instance-type` statement and specify the value `l2vpn`. To specify an interface connected to the router, include the `interface` statement and specify the VLAN-bundled logical interface:

```
instance-type l2vpn;
interface logical-interface-name;
```

You can include the statements at the following hierarchy levels:

- `[edit routing-instances routing-instance-name]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name]`

RELATED DOCUMENTATION

802.1Q VLANs Overview | 274

Ethernet Interfaces User Guide for Routing Devices

Configuring Access Mode on a Logical Interface

IN THIS SECTION

Enterprise network administrators can configure a single logical interface to accept untagged packets and forward the packets within a specified bridge domain. A logical interface configured to accept untagged packets is called an *access interface* or *access port*.

Use 802.1Q VLAN Trunk to confirm platform and release support for specific features.

Review the "Platform-Specific Access Mode Configuration Behavior" on page 308 section for notes related to your platform.

To configure access mode on a logical interface, use the `interface-mode access` statement at the `[edit interfaces interface-name unit logical-unit-number family bridge]` hierarchy level or at the `[edit logical-systems logical-system-name interfaces interface-name unit logical-unit-number family bridge]` hierarchy level.

When an untagged packet is received on an access interface, the packet is accepted, the configured VLAN ID is added to the packet, and the packet is forwarded within the bridge domain that is configured with the matching VLAN ID.

The following example configures a logical interface as an access port with a VLAN ID of 20:

```
[edit interfaces ge-1/2/0]
unit 0 {
    family bridge {
        interface-mode access;
        vlan-id 20;
    }
}
```

Platform-Specific Access Mode Configuration Behavior

Use Feature Explorer to confirm platform and release support for specific features.

Use the following table to review platform-specific behavior for your platform:

Platform	Difference
ACX Series	• ACX7000 Series routers that support access mode on logical interface must use the following hierarchy: `[edit interfaces interface-name unit logical-unit-number family ethernet-switching]`

RELATED DOCUMENTATION

802.1Q VLANs Overview | 274

Ethernet Interfaces User Guide for Routing Devices

Configuring a Logical Interface for Trunk Mode

IN THIS SECTION

As an alternative to configuring a logical interface for each VLAN, enterprise network administrators can configure a single logical interface to accept untagged packets or packets tagged with any VLAN ID specified in a list of VLAN IDs. Using a VLAN ID list conserves switch resources and simplifies configuration. A logical interface configured to accept packets tagged with any VLAN ID specified in a list is called a *trunk interface* or *trunk port*. See vlan members (VLANs).

Use 802.1Q VLAN Trunk to confirm platform and release support for specific features.

Review the "Platform-Specific Trunk Mode Logical Interface Configuration Behavior" on page 309 section for notes related to your platform.

To configure a logical interface to accept any packet tagged with a VLAN ID that matches the list of VLAN IDs, include the `interface-mode` statement and specify the `trunk` option:

```
interface-mode trunk;
```

You can include this statement at the following hierarchy levels:

- `[edit interfaces interface-name unit logical-unit-number family bridge]`
- `[edit logical-systems logical-system-name interfaces interface-name unit logical-unit-number family bridge]`

Platform-Specific Trunk Mode Logical Interface Configuration Behavior

Use Feature Explorer to confirm platform and release support for specific features.

Use the following table to review platform-specific behavior for your platform:

Platform	Difference
• ACX Series	• ACX7000 Series routers that support a logical interface for trunk mode accept any packet tagged with a VLAN ID that matches the list of VLAN IDs. Include the `interface-mode` statement and specify the `trunk` option: `[edit interfaces` *`interface-name`* `unit` *`logical-unit-number`* `family ethernet-switching]`.

RELATED DOCUMENTATION

802.1Q VLANs Overview | 274

Ethernet Interfaces User Guide for Routing Devices

Configuring the VLAN ID List for a Trunk Interface

To configure the list of VLAN IDs to be accepted by the trunk port, include the `vlan-id-list` statement and specify the list of VLAN IDs. You can specify individual VLAN IDs with a space separating the ID numbers, specify a range of VLAN IDs with a dash separating the ID numbers, or specify a combination of individual VLAN IDs and a range of VLAN IDs.

```
vlan-id-list [number number-number];
```

You can include this statement at the following hierarchy levels:

- `[edit interfaces` *`interface-name`* `unit` *`logical-unit-number`* `family bridge interface-mode trunk]`
- `[edit logical-systems` *`logical-system-name`* `interfaces` *`interface-name`* `unit` *`logical-unit-number`* `family bridge interface-mode trunk]`

When a packet is received that is tagged with a VLAN ID specified in the trunk interface list of VLAN IDs, the packet is accepted and forwarded within the bridge domain that is configured with the matching VLAN ID.

When a packet is received that is tagged with a VLAN ID not specified in the trunk interface list of VLAN IDs, the native VLAN ID is pushed in front of the existing VLAN tag or tags and the packet is forwarded within the bridge domain that is configured with the matching VLAN ID.

When an untagged packet is received on a trunk interface, the native VLAN ID is added to the packet and the packet is forwarded within the bridge domain that is configured with the matching VLAN ID.

A bridge domain configured with a matching VLAN ID must be configured before the trunk interface is configured. To learn more about configuring bridge domains, see the *Junos Routing Protocols Configuration Guide*.

RELATED DOCUMENTATION

802.1Q VLANs Overview | **274**

Ethernet Interfaces User Guide for Routing Devices

Configuring a Trunk Interface on a Bridge Network

On MX Series routers, you can configure a trunk interface on a bridge network.

The following output sample shows trunk port configuration on a bridge network:

```
user@host# run show interfaces
ge-0/0/0 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 0 {
        encapsulation vlan-bridge;
        vlan-id 1;
    }
}
ge-2/0/0 {
    unit 0 {
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-200;
        }
    }
}
```

```
ge-2/0/1 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 0 {
        encapsulation vlan-bridge;
        vlan-id 1;
    }
}
```

If you want `igmp-snooping` to be functional for a bridge domain, then you should not configure `interface-mode` and `irb` for that bridge domain. Such a configuration commit succeeds, but IGMP snooping is not functional, and a message informing the same is displayed as shown after the sample configuration below:

```
user@host# run show configuration
interfaces {
    ge-5/1/1 {
        flexible-vlan-tagging;
        native-vlan-id 1;
        unit 0 {
            family bridge {
                interface-mode trunk;
                vlan-id-list 401;
            }
        }
    }
    irb {
        unit 401 {
            family inet {
                address 192.168.2.2/27;
            }
        }
    }
}
protocols {
    igmp {
        interface all;
    }
}
bridge-domains {
    VLAN-401 {
        vlan-id 401;
```

```
        routing-interface irb.401;
        protocols {
            igmp-snooping;
        }
    }
}

user@host# commit
[edit bridge-domains]
  'VLAN-401'
    IGMP Snooping not supported with IRB and trunk mode interface ge-5/1/1.0
commit complete
```

To achieve IGMP snooping for a bridge domain, you should use such a configuration as shown in the following example:

```
user@host# run show configuration
interfaces {
    ge-0/0/1 {
        flexible-vlan-tagging;
        native-vlan-id 1;
        encapsulation flexible-ethernet-services;
        unit 0 {
            encapsulation vlan-bridge;
            vlan-id 401;
        }
    }
    irb {
        unit 401 {
            family inet {
                address 192.168.2.2/27;
            }
        }
    }
}
protocols {
    igmp {
        interface all;
    }
}
bridge-domains {
    VLAN-401 {
```

```
        vlan-id 401;
        interface ge-0/0/1.0;
        routing-interface irb.401;
        protocols {
            igmp-snooping;
        }
    }
}

user@host# commit
commit complete
```

RELATED DOCUMENTATION

- 802.1Q VLANs Overview | 274
- *interface-mode*
- Ethernet Interfaces User Guide for Routing Devices

Configuring a VLAN-Bundled Logical Interface to Support a Layer 2 VPN Routing Instance

To configure a VLAN-bundled logical interface, specify the list of VLAN IDs by including the `vlan-id-list` statement or the `vlan-tags` statement on a provider edge (PE) router:

```
interfaces {
    ethernet-interface-name {
        vlan-tagging; # Support single- or dual-tag logical interfaces
        flexible-vlan-tagging; # Support mixed tagging
        encapsulation (extended-vlan-ccc | flexible-ethernet-services);
        unit logical-unit-number {
            vlan-id-list [vlan-id vlan-id-vlan-id]; # For single-tag
            vlan-tags outer <tpid.>vlan-id inner-list [vlan-id vlan-id-vlan-id]; # For dual-tag
        }
        . . .
```

```
    }
}
```

You can include the statements at the following hierarchy levels:

- [edit]
- [edit logical-systems *logical-system-name*]

RELATED DOCUMENTATION

802.1Q VLANs Overview | 274

Ethernet Interfaces User Guide for Routing Devices

Configuring a VLAN-Bundled Logical Interface to Support a Layer 2 Circuit

To configure a VLAN-bundled logical interface, specify the list of VLAN IDs by including the `vlan-id-list` statement or the `vlan-tags` statement:

```
interfaces {
    ethernet-interface-name {
        vlan-tagging; # Support single- or dual-tag logical interfaces
        flexible-vlan-tagging; # Support mixed tagging
        encapsulation (extended-vlan-ccc | flexible-ethernet-services);
        unit logical-unit-number {
            encapsulation vlan-ccc; # Required for single-tag
            vlan-id-list [vlan-id vlan-id-vlan-id]; # For single-tag
            vlan-id-range vlan-id-range; # For single-tag with outer VLAN as a range of VLANs
            vlan-tags outer tpid.vlan-id inner-list [vlan-id vlan-id-vlan-id]; # For dual-tag
            vlan-tags outer vlan-id inner-range vlan-id-range # For dual-tag with outer VLAN and
inner VLAN as a range of VLANs
        }
        . . .
    }
}
```

You can include the statements at the following hierarchy levels:

- `[edit]`
- `[edit logical-systems logical-system-name]`

For a single-tag logical interface, include the `encapsulation` statement and specify `vlan-ccc` so that CCC circuit encapsulation is used inside the Layer 2 circuit.

For platforms that support `vlan-id-range` configurations, you can configure ranges in two ways:

- Single VLAN tagged: the outer tag is a VLAN range.
- Dual VLAN tagged: the outer tag is a single VLAN ID, and the inner tag is configured as a VLAN range.

NOTE: In the case of a dual-tag logical interface, the Junos OS automatically uses the `vlan-ccc` encapsulation type.

RELATED DOCUMENTATION

802.1Q VLANs Overview | 274

Specifying the Interface to Handle Traffic for a CCC Connected to the Layer 2 Circuit | 325

Ethernet Interfaces User Guide for Routing Devices

Configuring a Layer 2 Circuit on a VLAN-Bundled Logical Interface

IN THIS SECTION

This topic describes how to configure a Layer 2 circuit on a logical interface bound to a list of VLAN IDs.

Configuring a VLAN-Bundled Logical Interface to Support a Layer 2 Circuit

To configure a VLAN-bundled logical interface, specify the list of VLAN IDs by including the `vlan-id-list` statement or the `vlan-tags` statement:

```
interfaces {
    ethernet-interface-name {
        vlan-tagging; # Support single- or dual-tag logical interfaces
        flexible-vlan-tagging; # Support mixed tagging
        encapsulation (extended-vlan-ccc | flexible-ethernet-services);
        unit logical-unit-number {
            encapsulation vlan-ccc; # Required for single-tag
            vlan-id-list [vlan-id vlan-id-vlan-id]; # For single-tag
            vlan-id-range vlan-id-range; # For single-tag with outer VLAN as a range of VLANs
            vlan-tags outer tpid.vlan-id inner-list [vlan-id vlan-id-vlan-id]; # For dual-tag
            vlan-tags outer vlan-id inner-range vlan-id-range # For dual-tag with outer VLAN and
inner VLAN as a range of VLANs
        }
        . . .
    }
}
```

You can include the statements at the following hierarchy levels:

- `[edit]`
- `[edit logical-systems logical-system-name]`

For a single-tag logical interface, include the `encapsulation` statement and specify `vlan-ccc` so that CCC circuit encapsulation is used inside the Layer 2 circuit.

For platforms that support `vlan-id-range` configurations, you can configure ranges in two ways:

- Single VLAN tagged: the outer tag is a VLAN range.
- Dual VLAN tagged: the outer tag is a single VLAN ID, and the inner tag is configured as a VLAN range.

NOTE: In the case of a dual-tag logical interface, the Junos OS automatically uses the `vlan-ccc` encapsulation type.

SEE ALSO

802.1Q VLANs Overview | 274

Specifying the Interface to Handle Traffic for a CCC Connected to the Layer 2 Circuit | 325

Ethernet Interfaces User Guide for Routing Devices

Specifying the Interface to Handle Traffic for a CCC Connected to the Layer 2 Circuit

To configure the VLAN-bundled logical interface as the interface to handle traffic for a circuit connected to the Layer 2 circuit, include the following statements:

```
l2circuit {
    neighbor address {
        interface logical-interface-name {
            virtual-circuit-id number;
            no-control-word;
        }
    }
}
```

You can include the statements at the following hierarchy levels:

- `[edit protocols]`
- `[edit logical-systems logical-system-name protocols]`

To enable a Layer 2 circuit, include the `l2circuit` statement.

To configure the router as a neighbor for a Layer 2 circuit, specify the neighbor address using the `neighbor` statement.

To specify the interface to handle traffic for a circuit connected to the Layer 2 circuit, include the `interface` statement and specify the VLAN-bundled logical interface.

SEE ALSO

802.1Q VLANs Overview | 274

Example: Configuring a Layer 2 Circuit on a VLAN-Bundled Logical Interface | 319

Configuring a VLAN-Bundled Logical Interface to Support a Layer 2 Circuit | 315

Example: Configuring a Layer 2 Circuit on a VLAN-Bundled Logical Interface

The following configuration shows that the single-tag logical interface `ge-1/0/5.0` bundles a list of VLAN IDs, and the logical interface `ge-1/1/1.0` supports IPv4 traffic using IP address 10.30.1.1/30 and can participate in an MPLS path.

```
[edit interfaces]
ge-1/0/5 {
    vlan-tagging;
    encapsulation extended-vlan-ccc;
    unit 0 { # VLAN-bundled logical interface
        vlan-id-list [513 516 520-525];
    }
}
ge-1/1/1 {
    unit 0 {
        family inet {
            address 10.30.1.1/30;
        }
        family mpls;
    }
}
```

The following configuration shows the type of traffic supported on the Layer 2 VPN routing instance, and shows that the VLAN-bundled logical interface handles traffic for a CCC to which the Layer 2 circuit connects:

```
[edit protocols]
rsvp {
    interface all;
    interface lo0.0;
}
mpls {
    label-switched-path lsp {
```

```
        to 10.255.69.128;
    }
    interface all;
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface lo0.0;
        interface ge-1/1/1.0;
    }
}
ldp {
    interface ge-1/1/1.0;
    interface ge-1/0/5.0; # VLAN-bundled logical interface
    interface lo0.0;
}
l2circuit {
    neighbor 10.255.69.128 {
        interface ge-1/0/5.0 { # VLAN-bundled logical interface
            virtual-circuit-id 3;
            no-control-word;
        }
    }
}
```

RELATED DOCUMENTATION

802.1Q VLANs Overview | 274

Ethernet Interfaces User Guide for Routing Devices

Guidelines for Configuring VLAN ID List-Bundled Logical Interfaces That Connect CCCs

IN THIS SECTION

For MX Series routers, you can bind a list of VLAN IDs to a *logical interface*, configure a Layer 2 VPN routing instance or Layer 2 circuit on the logical interface, and then use the logical interface to configure a circuit cross-connect (CCC) to another Layer 2 VPN routing instance or Layer 2 circuit.

A CCC allows you to configure transparent connections between two circuits so that packets from the source circuit are delivered to the destination circuit with, at most, the Layer 2 address being changed. You configure a CCC by connecting circuit interfaces of the same type. For more information, see Circuit and Translational Cross-Connects Overview.

NOTE: The Junos OS supports binding of Ethernet logical interfaces to lists of VLAN IDs on MX Series routers only. For all other routers, you can bind an Ethernet logical interface to only a single VLAN ID or to a single range of VLAN IDs.

The following configuration guidelines apply to bundling lists of VLAN IDs to Ethernet logical interfaces used to configure CCCs:

Guidelines for Configuring Physical Link-Layer Encapsulation to Support CCCs

To enable a physical interface to support VLAN-bundled logical interfaces that you will use to configure a CCC, you must specify one of the following physical link-layer encapsulation types as the value of the `encapsulation` statement at the `[edit interfaces interface-name]` hierarchy level:

```
[edit interfaces interface-name]
encapsulation (extended-vlan-ccc | flexible-ethernet-services);
```

- `extended-vlan-ccc`—For Ethernet interfaces with standard TPID tagging.
- `flexible-ethernet-services`—For supported Gigabit Ethernet interfaces for which you want to configure multiple per-unit Ethernet encapsulations.

For more information about configuring the encapsulation on a physical interface, see .

Guidelines for Configuring Logical Link-Layer Encapsulation to Support CCCs

For VLAN-bundled logical interfaces that you use to configure a CCC, specific logical link-layer encapsulation types are used inside the circuits themselves.

Table 60 on page 322 describes the logical link-layer encapsulation types used within circuits connected using VLAN-bundled logical interfaces of the same type.

Table 60: Encapsulation Inside Circuits CCC-Connected by VLAN-Bundled Logical Interfaces

Encapsulation Inside the Circuit	Layer 2 Circuit Joined by Configuring an Interface-to-Interface CCC Connection	
	Layer 2 VPN Routing Instance	Layer 2 Circuit
Syntax	`encapsulation-type (ethernet | ethernet-vlan);`	`encapsulation vlan-ccc;`

Table 60: Encapsulation Inside Circuits CCC-Connected by VLAN-Bundled Logical Interfaces *(Continued)*

Encapsulation Inside the Circuit	Layer 2 Circuit Joined by Configuring an Interface-to-Interface CCC Connection	
	Layer 2 VPN Routing Instance	**Layer 2 Circuit**
Hierarchy Level	`[edit routing-instances` *`routing-instance-name`* `protocols l2vpn]`, `[edit logical-systems` *`logical-system-name`* `routing-instances` *`routing-instance-name`* `protocols l2vpn]`	`[edit interfaces` *`ethernet-interface-name`* `unit` *`lo unit-number`*`]`, `[edit logical-systems` *`logical-system-name`* `interfa ethernet-interface-name unit logical-unit-number`
Usage Guidelines	See the Junos OS VPNs Library for Routing Devices.	See *Configuring Interface Encapsulation on Logica Interfaces*, Circuit and Translational Cross-Connec Overview, and Defining the Encapsulation for Swi Cross-Connects.
For a Single-Tag Logical Interface	The MX Series router automatically uses `ethernet` as the Layer 2 protocol used to encapsulate incoming traffic. Although the connection spans multiple VLANs, the VLANs are bundled and therefore can be encapsulated as a single VLAN. **NOTE:** With `ethernet` encapsulation, the circuit signal processing does not check that the VLAN ID list is the same at both ends of the CCC connection.	Configure the MX Series router to use `vlan-ccc` as logical link-layer encapsulation type.
For a Dual-Tag Logical Interface	Configure the MX Series router to use `ethernet-vlan` as the Layer 2 protocol to encapsulate incoming traffic. With `ethernet-vlan` encapsulation, circuit signal processing checks that the VLAN ID list is the same at both ends of the CCC connection. If a VLAN ID list mismatch is detected, you can view the error condition in the `show interfaces` command output.	The MX Series router automatically uses `vlan-ccc` logical link-layer encapsulation type, regardless of value configured.

RELATED DOCUMENTATION

802.1Q VLANs Overview | 274

Binding VLAN IDs to Logical Interfaces | 292

Defining the Encapsulation for Switching Cross-Connects

Specifying the Interface to Handle Traffic for a CCC

To configure the VLAN-bundled logical interface as the interface to handle traffic for a circuit connected to the Layer 2 VPN routing instance, include the following statements:

```
protocols {
    l2vpn {
        (control-word | no-control-word);
        encapsulation-type (ethernet | ethernet-vlan);
        site site-name {
            site-identifier identifier;
            interface logical-interface-name { # VLAN-bundled logical interface
                . . . interface-options . . .
            }
        }
    }
}
```

You can include the statements at the same hierarchy level at which you include the `instance-type l2vpn` and `interface` *`logical-interface-name`* statements:

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

To enable a Layer 2 VPN routing instance on a PE router, include the `l2vpn` statement. For more information, see the Junos OS VPNs Library for Routing Devices.

The `encapsulation-type` statement specifies the Layer 2 protocol used for traffic from the customer edge (CE) router. If the Layer 2 VPN routing instance is being connected to a single-tag Layer 2 circuit, specify `ethernet` as the encapsulation type. If the Layer 2 VPN routing instance is being connected to a dual-tag Layer 2 circuit, specify `ethernet-vlan` as the encapsulation type.

To specify the interface to handle traffic for a circuit connected to the Layer 2 VPN routing instance, include the `interface` statement and specify the VLAN-bundled logical interface.

RELATED DOCUMENTATION

802.1Q VLANs Overview | 274

Example: Configuring a Layer 2 VPN Routing Instance on a VLAN-Bundled Logical Interface | 304

Ethernet Interfaces User Guide for Routing Devices

Specifying the Interface to Handle Traffic for a CCC Connected to the Layer 2 Circuit

To configure the VLAN-bundled logical interface as the interface to handle traffic for a circuit connected to the Layer 2 circuit, include the following statements:

```
l2circuit {
    neighbor address {
        interface logical-interface-name {
            virtual-circuit-id number;
            no-control-word;
        }
    }
}
```

You can include the statements at the following hierarchy levels:

- `[edit protocols]`
- `[edit logical-systems logical-system-name protocols]`

To enable a Layer 2 circuit, include the `l2circuit` statement.

To configure the router as a neighbor for a Layer 2 circuit, specify the neighbor address using the `neighbor` statement.

To specify the interface to handle traffic for a circuit connected to the Layer 2 circuit, include the `interface` statement and specify the VLAN-bundled logical interface.

RELATED DOCUMENTATION

802.1Q VLANs Overview | 274

11

CHAPTER

Configuring Static ARP Table Entries

IN THIS CHAPTER

Static ARP Table Entries Overview

For Gigabit Ethernet, Tri-Rate Ethernet copper, and 10-Gigabit Ethernet interfaces, you can configure static ARP table entries, defining mappings between IP and MAC addresses.

RELATED DOCUMENTATION

Ethernet Interfaces User Guide for Routing Devices

Static ARP Entries on Ethernet Interfaces

You can provision static Address Resolution Protocol (ARP) entries for a device instead of dynamically resolving an IP address to a MAC address. Note that dynamic resolution of an IP address is the default behavior. These static ARP entries enable the device to respond to ARP requests even if the destination address of the request is not local to the Ethernet interface that receives the incoming traffic.

Configuring Static ARP Table Entries For Mapping IP Addresses to MAC Addresses

By default, the device responds to an Address Resolution Protocol (ARP) request only if the destination address of the ARP request is on the local network of the incoming interface. For Gigabit Ethernet interfaces, you can configure static ARP entries that associate the IP addresses of nodes on the same Ethernet subnet with their media access control (MAC) addresses. These static ARP entries enable the device to respond to ARP requests even if the destination address of the ARP request is not local to the incoming Ethernet interface.

Also, unlike dynamically learned ARP entries, static ARP entries do not age out. You can also configure static ARP entries in a troubleshooting situation or if your device is unable to learn a MAC address dynamically.

NOTE: By default, an ARP policer is installed that is shared among all the Ethernet interfaces on which you have configured the `family inet` statement. By including the `arp`

statement at the [edit interfaces *interface-name* unit *logical-unit-number* family inet policer] hierarchy level, you can apply a specific ARP-packet policer to an interface. This feature is not available on EX Series switches.

To configure static ARP entries:

1. In the configuration mode, at the [edit] hierarchy level, configure the router interface on which the ARP table entries for the router is configured.

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

2. Configure the protocol family, the logical unit of the interface, and the interface address of the router interface at the [edit interfaces *interface-name*] hierarchy level. While configuring the protocol family, specify inet as the protocol family.

NOTE: When you need to conserve IP addresses, you can configure an Ethernet interface to be unnumbered by including the unnumbered-address statement at the [edit interfaces *interface-name* unit *logical-unit-number* family inet] hierarchy level.

```
[edit interfaces interface-name]
user@host# edit unit logical-unit-number family inet address interface-address
```

3. Configure a static ARP entry by specifying the IP address and the MAC address that are to be mapped to each other. The IP address specified must be part of the subnet defined in the enclosing address statement. The MAC address must be specified as hexadecimal bytes in the following formats: *nnnn*.*nnnn*.*nnnn* or *nn*:*nn*:*nn*:*nn*:*nn*:*nn*format. For instance, you can use either 0011.2233.4455 or 00:11:22:33:44:55.

```
[edit interfaces interface-name unit logical-unit-number family inet address interface-address
user@host# set arp ip-address mac mac-address
```

4. Configure another static ARP entry by specifying the IP address and the MAC address that are to be mapped to each other. You can also associate a multicast MAC address with a unicast IP address by including the multicast-mac option with the arp statement. You can optionally configure the router to respond to ARP requests for the specified IP address by using the publish option with the arp statement.

NOTE: For unicast MAC addresses only, if you include the `publish` option, the router or switch replies to proxy ARP requests.

```
[edit interfaces interface-name unit logical-unit-number family inet address interface-address
user@host# set arp ip-address multicast-mac mac-address publish
```

NOTE: The Junos OS supports the IPv6 static neighbor discovery cache entries, similar to the static ARP entries in IPv4.

RELATED DOCUMENTATION

- arp
- Management Ethernet Interface Overview
- *Applying Policers*
- Configuring an Unnumbered Interface
- Ethernet Interfaces User Guide for Routing Devices

Example: Configuring Static ARP Entries on Ethernet Interfaces

IN THIS SECTION

Requirements

No special configuration beyond device initialization is required before creating an interface.

Overview

In this example, you configure a static ARP entry on the logical unit 0 of the ge-0/0/3 Gigabit Ethernet interface. The entry consists of the interface's IP address (10.1.1.1/24) and the corresponding MAC address of a node on the same Ethernet subnet (00:ff:85:7f:78:03). The example also configures the device to reply to ARP requests from the node using the publish option.

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

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

```
set interfaces ge-0/0/3 unit 0 family inet address 10.1.1.1/24 arp 10.1.1.3 mac
00:ff:85:7f:78:03
set interfaces ge-0/0/3 unit 0 family inet address 10.1.1.1/24 arp 10.1.1.3 publish
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode*.

To configure a static ARP entry on an Ethernet interface:

1. Create the Gigabit Ethernet interface.

```
[edit]
user@host# edit interfaces ge-0/0/3
```

2. Configure a static ARP entry.

```
[edit interfaces ge-0/0/3]
user@host# edit unit 0 family inet address 10.1.1.1/24
```

3. Set the IP address of the subnet node and the corresponding MAC address.

```
[edit interfaces ge-0/0/3 unit 0 family inet address 10.1.1.1/24]
user@host# set arp 10.1.1.3 mac 00:ff:85:7f:78:03 publish
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces ge-0/0/3` command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
[edit]
user@host#show interfaces ge-0/0/3
unit 0 {
    family inet {
        address 10.1.1.1/24 {
            arp 10.1.1.3 mac 00:ff:85:7f:78:03 publish;
        }
    }
}
```

If you are done configuring the device, enter `commit` from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying Static ARP Configurations

Purpose

Verify the IP address and MAC (hardware) address of the node.

Action

From operational mode, enter the `show interfaces ge-0/0/3` command.

Verifying the Link State of All Interfaces

Purpose

Verify that all interfaces on the device are operational using the ping tool on each peer address in the network.

Action

For each interface on the device:

1. In the J-Web interface, select `Troubleshoot>Ping Host`.
2. In the Remote Host box, type the address of the interface for which you want to verify the link state.

3. Click `Start`. The output appears on a separate page.

```
PING 10.1.1.3 : 56 data bytes
64 bytes from 10.1.1.3: icmp_seq=0 ttl=255 time=0.382 ms
64 bytes from 10.1.1.3: icmp_seq=1 ttl=255 time=0.266 ms
```

If the interface is operational, it generates an ICMP response. If this response is received, the round-trip time in milliseconds is listed in the time field..

Verifying Interface Properties

Purpose

Verify that the interface properties are correct.

Action

From operational mode, enter the `show interfaces detail` command.

```
user@host> show interfaces detail
Physical interface: ge-0/0/3, Enabled, Physical link is Up
  Interface index: 134, SNMP ifIndex: 27, Generation: 17
  Link-level type: Ethernet, MTU: 1514, Speed: 100mbps, Loopback: Disabled,
  Source filtering: Disabled, Flow control: Enabled
  Device flags   : Present Running
  Interface flags: SNMP-Traps 16384
  Link flags     : None
  CoS queues     : 4 supported
  Hold-times     : Up 0 ms, Down 0 ms
  Current address: 00:90:69:87:44:9d, Hardware address: 00:90:69:87:44:9d
  Last flapped   : 2004-08-25 15:42:30 PDT (4w5d 22:49 ago)
  Statistics last cleared: Never
  Traffic statistics:
   Input  bytes  :                   0                   0 bps
   Output bytes  :                   0                   0 bps
   Input  packets:                   0                   0 pps
   Output packets:                   0                   0 pps
  Queue counters:       Queued packets  Transmitted packets      Dropped packets
    0 best-effort                    0                    0                    0
    1 expedited-fo                   0                    0                    0
```

```
    2 assured-forw                 0                 0                 0
    3 network-cont                 0                 0                 0
  Active alarms  : None
  Active defects : None
```

The output shows a summary of interface information. Verify the following information:

- The physical interface is Enabled. If the interface is shown as Disabled, do one of the following:
  - In the CLI configuration editor, delete the `disable` statement at the [edit interfaces ge-0/0/3] level of the configuration hierarchy.
  - In the J-Web configuration editor, clear the `Disable` check box on the Interfaces> ge-0/0/3 page.
- The physical link is Up. A link state of Down indicates a problem with the interface module, interface port, or physical connection (link-layer errors).
- The Last Flapped time is an expected value. The Last Flapped time indicates the last time the physical interface became unavailable and then available again. Unexpected flapping indicates likely link-layer errors.
- The traffic statistics reflect expected input and output rates. Verify that the number of inbound and outbound bytes and packets matches expected throughput for the physical interface. To clear the statistics and see only new changes, use the `clear interfaces statistics ge-0/0/3` command.

12

CHAPTER

Configuring Restricted and Unrestricted Proxy ARP

IN THIS CHAPTER

Restricted and Unrestricted Proxy ARP Overview

IN THIS SECTION

By default, the Junos OS responds to an Address Resolution Protocol (ARP) request only if the destination address of the ARP request is local to the incoming interface.

For Ethernet Interfaces, you can configure the router or switches to proxy-reply to the ARP requests using the restricted or unrestricted proxy ARP configuration.

You might want to configure restricted or unrestricted proxy ARP for routers that act as provider edge (PE) devices in Ethernet Layer 2 LAN switching domains.

Restricted Proxy ARP

Restricted proxy ARP enables the router or switch to respond to the ARP requests in which the physical networks of the source and target are not the same and the router or switch has an active route to the target address in the ARP request. The router does not reply if the target address is on the same subnet and the same interface as the ARP requestor.

Unrestricted Proxy ARP

Unrestricted proxy ARP enables the router or switch to respond to any ARP request, on condition that the router has an active route to the destination address of the ARP request. The route is not limited to the incoming interface of the request, nor is it required to be a direct route.

WARNING: If you configure unrestricted proxy ARP, the proxy router replies to ARP requests for the target IP address on the same interface as the incoming ARP request.

This behavior is appropriate for cable modem termination system (CMTS) environments, but might cause Layer 2 reachability problems if you enable unrestricted proxy ARP in other environments.

When an IP client broadcasts the ARP request across the Ethernet wire, the end node with the correct IP address responds to the ARP request and provides the correct MAC address. If the unrestricted proxy ARP feature is enabled, the router response is redundant and might fool the IP client into determining that the destination MAC address within its own subnet is the same as the address of the router.

NOTE: While the destination address can be remote, the source address of the ARP request must be on the same subnet as the interface upon which the ARP request is received. For security reasons, this rule applies to both unrestricted and restricted proxy ARP.

Topology Considerations for Unrestricted Proxy ARP

In most situations, you should not configure the router or switch to perform unrestricted proxy ARP. Do so only for special situations, such as when cable modems are used. Figure 2 on page 339 and Figure 3 on page 339 show examples of situations in which you might want to configure unrestricted proxy ARP.

In Figure 2 on page 339, the edge device is not running any IP protocols. In this case, you configure the core router to perform unrestricted proxy ARP. The edge device is the client of the proxy.

In Figure 3 on page 339, the Broadband Remote Access Server (B-RAS) routers are not running any IP protocols. In this case, you configure unrestricted proxy ARP on the B-RAS interfaces. This allows the core device to behave as though it is directly connected to the end users.

Figure 2: Edge Device Case for Unrestricted Proxy ARP

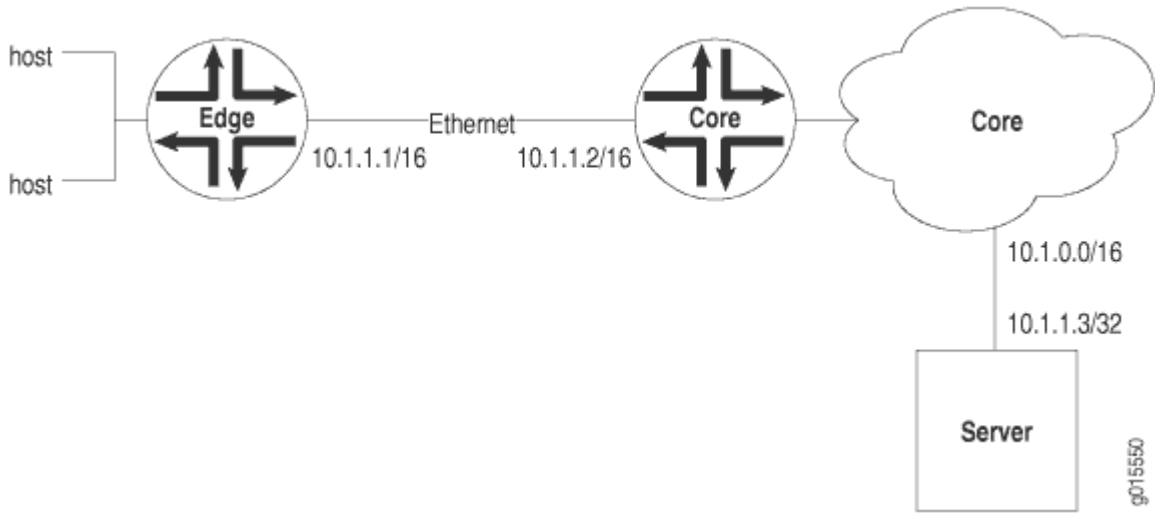
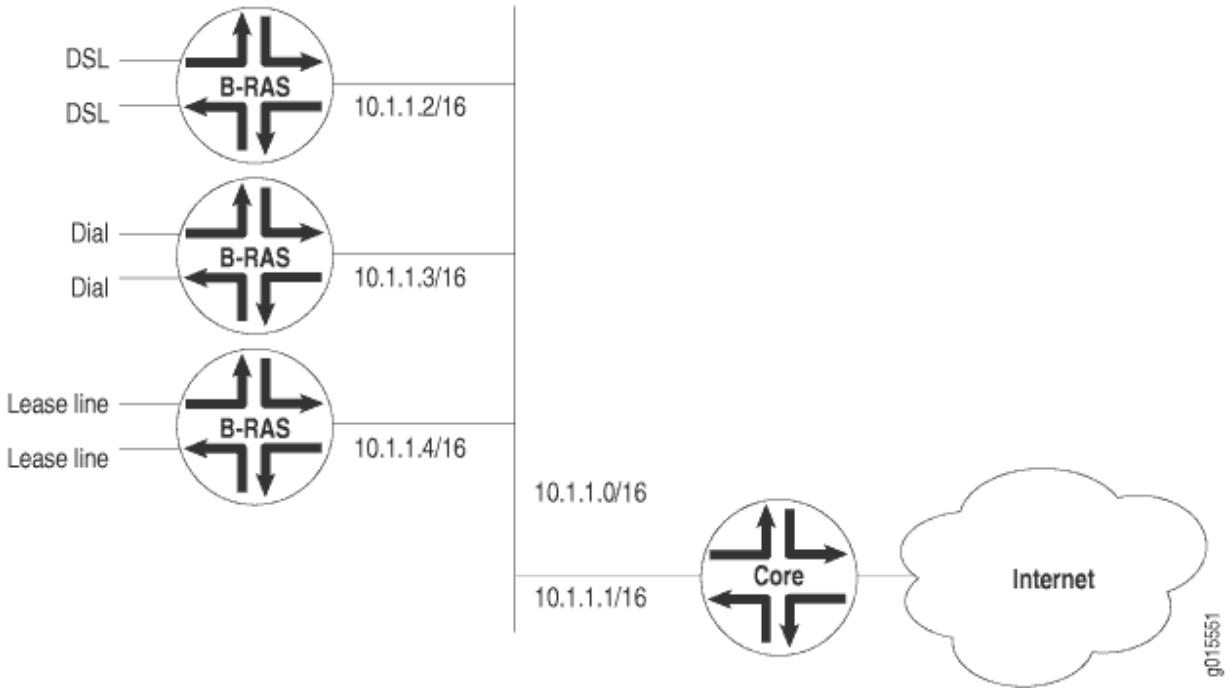


Figure 3: Core Device Case for Unrestricted Proxy ARP



RELATED DOCUMENTATION

- Ethernet Interfaces User Guide for Routing Devices

Configuring Restricted and Unrestricted Proxy ARP

To configure restricted or unrestricted proxy ARP, include the `proxy-arp` statement:

```
proxy-arp  (restricted |unrestricted);
```

You can include this statement at the following hierarchy levels:

- `[edit interfaces` *`interface-name`* `unit` *`logical-unit-number`* `]`
- `[edit logical-systems` *`logical-system-name`* `interfaces` *`interface-name`* `unit` *`logical-unit-number`*`]`

To return to the default—that is, to disable restricted or unrestricted proxy ARP—delete the `proxy-arp` statement from the configuration:

```
[edit]
user@host# delete interfaces interface-name unit logical-unit-number proxy-arp
```

You can track the number of restricted or unrestricted proxy ARP requests processed by the router or switch by issuing the `show system statistics arp` operational mode command.

NOTE: When proxy ARP is enabled as default or unrestricted, the router or switch responds to any ARP request as long as the device has an active route to the target address of the ARP request. This gratuitous ARP behavior can result in an error when the receiving interface and target response interface are the same and the end device (for example, a client) performs a duplicate address check. To prevent this error, configure the router or switch interface with the `no-gratuitous-arp-request` statement. See *Configuring Gratuitous ARP* for information about how to disable responses to gratuitous ARP requests.

RELATED DOCUMENTATION

Ethernet Interfaces User Guide for Routing Devices

13

CHAPTER

Configuring Gratuitous ARP

IN THIS CHAPTER

Configuring Gratuitous ARP

Gratuitous Address Resolution Protocol (ARP) requests help detect duplicate IP addresses. A gratuitous ARP is a broadcast request for a router's own IP address. If a router or switch sends an ARP request for its own IP address and no ARP replies are received, the router- or switch-assigned IP address is not being used by other nodes. However, if a router or switch sends an ARP request for its own IP address and an ARP reply is received, the router- or switch-assigned IP address is already being used by another node.

Gratuitous ARP replies are reply packets sent to the broadcast MAC address with the target IP address set to be the same as the sender's IP address. When the router or switch receives a gratuitous ARP reply, the router or switch can insert an entry for that reply in the ARP cache. By default, updating the ARP cache on gratuitous ARP replies is disabled on the router or switch.

To enable updating of the ARP cache for gratuitous ARPs:

1. In configuration mode, go to the `[edit interfaces interface-name]` hierarchy level.

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

2. Include the `gratuitous-arp-reply` statement.

```
[edit interfaces interface-name]
user@host# set gratuitous-arp-reply
```

To restore the default behavior, that is, to disable updating of the ARP cache for gratuitous ARP, delete the `gratuitous-arp-reply` statement from the configuration:

```
[edit interfaces interface-name]
user@host# delete gratuitous-arp-reply;
```

By default, the router or switch responds to gratuitous ARP requests. However, on Ethernet interfaces, you can disable responses to gratuitous ARP requests.

To disable responses to gratuitous ARP requests:

1. In configuration mode, go to the `[edit interfaces interface-name]` hierarchy level.

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

2. Include the `no-gratuitous-arp-request` statement.

```
[edit interfaces interface-name]
user@host# set no-gratuitous-arp-request
```

To return to the default—that is, to respond to gratuitous ARP requests—delete the `no-gratuitous-arp-request` statement from the configuration:

```
[edit interfaces interface-name]
user@host# delete no-gratuitous-arp-request
```

RELATED DOCUMENTATION

- gratuitous-arp-reply
- *no-gratuitous-arp-request*
- *Ethernet Interfaces Overview*
- Ethernet Interfaces User Guide for Routing Devices

14

CHAPTER

Adjusting the ARP Aging Timer

IN THIS CHAPTER

Adjusting the ARP Aging Timer

By default, the ARP aging timer is set at 20 minutes. In environments with many directly attached hosts, such as metro Ethernet environments, increasing the amount of time between ARP updates by configuring the ARP aging timer can improve performance in an event where having thousands of clients time out at the same time might impact packet forwarding performance. In environments where there are devices connected with lower ARP aging timers (less than 20 minutes), decreasing the ARP aging timer can improve performance by preventing the flooding of traffic toward next hops with expired ARP entries. In most environments, the default ARP aging timer value does not need to be adjusted.

To configure the system-wide ARP aging timer, include the `aging-timer` statement at the `[edit system arp]` hierarchy level:

```
[edit system arp]
user@host# aging-timer minutes
```

The aging timer range is from 1 through 240 minutes. The timer value you configure takes effect as ARP entries expire. In other words, each subsequent refreshed ARP entry receives the new timer value. The new timer value does not apply to ARP entries that exist at the time you commit the configuration.

For more information about statements you can configure at the `[edit system]` hierarchy level, see the CLI User Guide for Junos OS.

RELATED DOCUMENTATION

> *arp*
>
> *Ethernet Interfaces Overview*
>
> Ethernet Interfaces User Guide for Routing Devices

CHAPTER 15

Configuring Tagged VLANs

IN THIS CHAPTER

Configuring Tagged VLANs

IN THIS SECTION

Creating a Series of Tagged VLANs

When you divide an Ethernet LAN into multiple VLANs, each VLAN is assigned a unique IEEE 802.1Q tag. This tag is associated with each frame in the VLAN, and the network nodes receiving the traffic can use the tag to identify which VLAN a frame is associated with.

Instead of configuring VLANs and 802.1Q tags one at a time for a trunk interface, you can configure a VLAN range to create a series of tagged VLANs.

When an Ethernet LAN is divided into VLANs, each VLAN is identified by a unique 802.1Q tag. The tag is applied to all frames so that the network nodes receiving the frames can detect which VLAN the frames belong to. Trunk ports, which multiplex traffic among a number of VLANs, use the tag to determine the origin of frames and where to forward them.

For example, you could configure the VLAN **employee** and specify a tag range of **10 through 12**. This creates the following VLANs and tags:

- VLAN **employee-10**, tag **10**
- VLAN **employee-11**, tag **11**
- VLAN **employee-12**, tag **12**

Creating tagged VLANs in a series has the following limitations:

- Layer 3 interfaces do not support this feature.
- Because an access interface can only support one VLAN member, access interfaces also do not support this feature.

NOTE: This task uses Junos OS for QFX3500 and QFX3600 switches that does not support the Enhanced Layer 2 Software (ELS) configuration style. If your switch runs software that does support ELS, see "Creating a Series of Tagged VLANs on Switches with ELS Support" on page 351.

To configure a series of tagged VLANs using the CLI (here, the VLAN is **employee**):

1. Configure the series (here, a VLAN series from 120 through 130):

```
[edit]
user@switch# set vlans employee vlan-range 120-130
```

2. Associate a series of tagged VLANs when you configure an interface in one of two ways:

- Include the name of the series:

```
[edit interfaces]
user@switch# set interfaces xe-0/0/22.0 family ethernet-switching vlan members
employee
```

- Include the VLAN range:

```
[edit interfaces]
user@switch# set interfaces xe-0/0/22.0 family ethernet-switching vlan members 120-
130
```

Associating a series of tagged VLANS to an interface by name or by VLAN range has the same result: VLANs **__employee_120__** through **__employee_130__** are created.

NOTE: When a series of VLANs is created using the `vlan-range` command, the VLAN names are preceded and followed by a double underscore.

Creating a Series of Tagged VLANs on EX Series Switches (CLI Procedure)

To identify which VLAN traffic belongs to, all frames on an Ethernet VLAN are identified by a tag, as defined in the IEEE 802.1Q standard. These frames are *tagged* and are encapsulated with 802.1Q tags. For a simple network that has only a single VLAN, all traffic has the same 802.1Q tag.

Instead of configuring VLANS and 802.1Q tags one at a time for a trunk interface, you can configure a VLAN range to create a series of tagged VLANs.

When an Ethernet LAN is divided into VLANs, each VLAN is identified by a unique 802.1Q tag. The tag is applied to all frames so that the network nodes receiving the frames know which VLAN the frames belong to. Trunk ports, which multiplex traffic among a number of VLANs, use the tag to determine the origin of frames and where to forward them.

For example, you could configure the VLAN **employee** and specify a tag range of **10-12**. This creates the following VLANs and tags:

- VLAN **employee-10**, tag **10**
- VLAN **employee-11**, tag **11**
- VLAN **employee-12**, tag **12**

Creating tagged VLANs in a series has the following limitations:

- Layer 3 interfaces do not support this feature.
- Because an access interface can only support one VLAN member, access interfaces also do not support this feature.
- Voice over IP (VoIP) configurations do not support a range of tagged VLANs.

To configure a series of tagged VLANs using the CLI (here, the VLAN is **employee**):

1. Configure the series (here, a VLAN series from 120 through 130):

```
[edit]
user@switch# set vlans employee vlan-range 120-130
```

2. Associate a series of tagged VLANs when you configure an interface in one of two ways:

- Include the name of the series:

```
[edit interfaces]
user@switch# set interfaces ge-0/0/22.0 family ethernet-switching vlan members
employee
```

- Include the VLAN range:

```
[edit interfaces]
user@switch# set interfaces ge-0/0/22.0 family ethernet-switching vlan members 120-
130
```

Associating a series of tagged VLANS to an interface by name or by VLAN range have the same result: VLANs **__employee_120__** through **__employee_130__** are created.

NOTE: When a series of VLANs are created using the `vlan-range` command, the VLAN names are prefixed and suffixed with a double underscore.

SEE ALSO

- Verifying That a Series of Tagged VLANs Has Been Created on an EX Series Switch | 355
- Example: Setting Up Basic Bridging and a VLAN for an EX Series Switch | 195
- Example: Setting Up Bridging with Multiple VLANs for EX Series Switches | 240
- Example: Connecting an EX Series Access Switch to a Distribution Switch
- Understanding Bridging and VLANs on Switches | 140

Creating a Series of Tagged VLANs on Switches with ELS Support

When you divide an Ethernet LAN into multiple VLANs, each VLAN is assigned a unique IEEE 802.1Q tag. This tag is associated with each frame in the VLAN, and the network nodes receiving the traffic can use the tag to identify which VLAN a frame is associated with.

Instead of configuring VLANs and 802.1Q tags one at a time for a trunk interface, you can configure a VLAN range to create a series of tagged VLANs.

When an Ethernet LAN is divided into VLANs, each VLAN is identified by a unique 802.1Q tag. The tag is applied to all frames so that the network nodes receiving the frames can detect which VLAN the frames belong to. Trunk ports, which multiplex traffic among a number of VLANs, use the tag to determine the origin of frames and where to forward them.

For example, you could configure the VLAN **employee** and specify a tag range of **10 through 12**. This creates the following VLANs and tags:

- VLAN **employee-10**, tag **10**
- VLAN **employee-11**, tag **11**
- VLAN **employee-12**, tag **12**

Creating tagged VLANs in a series has the following limitations:

- Layer 3 interfaces do not support this feature.
- Because an access interface can only support one VLAN member, access interfaces also do not support this feature.

NOTE: This task uses Junos OS for Junos OS for QFX3500 and QFX3600 switches with support for the Enhanced Layer 2 Software (ELS) configuration style. If your switch runs software that does not support ELS, see "Creating a Series of Tagged VLANs" on page 347. For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15.

To configure a series of tagged VLANs using the CLI (here, the VLAN is **employee**):

1. Configure the series (here, a VLAN series from 120 through 130):

```
[edit]
user@switch# set vlans employee vlan-id-list [ 120-130 ]
```

2. Associate a series of tagged VLANs when you configure an interface in one of two ways:

- Include the name of the series:

```
[edit interfaces]
user@switch# set interfaces xe-0/0/22.0 family ethernet-switching vlan members
employee
```

- Include the VLAN range:

```
[edit interfaces]
user@switch# set interfaces xe-0/0/22.0 family ethernet-switching vlan members 120-
130
```

Associating a series of tagged VLANS to an interface by name or by VLAN range the same result: VLANs **__employee_120__** through **__employee_130__** are created.

NOTE: When a series of VLANs is created using the `vlan-id-list` command, the VLAN names are preceded and followed by a double underscore.

SEE ALSO

Example: Setting Up Bridging with Multiple VLANs on Switches | 215

Understanding Bridging and VLANs on Switches | 140

Verifying That a Series of Tagged VLANs Has Been Created

IN THIS SECTION

Purpose

Verify that a series of tagged VLANs has been created on the switch.

Action

1. Display the VLANs in the ascending order of their VLAN ID:

```
user@switch> show vlans sort-by tag

Name             Tag        Interfaces
__employee_120__  120
                            xe-0/0/22.0*
__employee_121__  121
                            xe-0/0/22.0*
__employee_122__  122
                            xe-0/0/22.0*
__employee_123__  123
                            xe-0/0/22.0*
__employee_124__  124
                            xe-0/0/22.0*
__employee_125__  125
                            xe-0/0/22.0*
__employee_126__  126
                            xe-0/0/22.0*
__employee_127__  127
                            xe-0/0/22.0*
__employee_128__  128
                            xe-0/0/22.0*
__employee_129__  129
                            xe-0/0/22.0*
__employee_130__  130
                            xe-0/0/22.0*
```

2. Display the VLANs by the alphabetical order of the VLAN name:

```
user@switch> show vlans sort-by name

Name             Tag   Interfaces

__employee_120__ 120
```

```
                        xe-0/0/22.0*
__employee_121__ 121
                        xe-0/0/22.0*
__employee_122__ 122
                        xe-0/0/22.0*
__employee_123__ 123
                        xe-0/0/22.0*
__employee_124__ 124
                        xe-0/0/22.0*
__employee_125__ 125
                        xe-0/0/22.0*
__employee_126__ 126
                        xe-0/0/22.0*
__employee_127__ 127
                        xe-0/0/22.0*
__employee_128__ 128
                        xe-0/0/22.0*
__employee_129__ 129
                        xe-0/0/22.0*
__employee_130__ 130
                        xe-0/0/22.0*
```

3. Display the VLANs by specifying the VLAN range name (here, the VLAN range name is **employee**):

```
user@switch> show vlans employee

Name             Tag   Interfaces

__employee_120__ 120
                       xe-0/0/22.0*
__employee_121__ 121
                       xe-0/0/22.0*
__employee_122__ 122
                       xe-0/0/22.0*
__employee_123__ 123
                       xe-0/0/22.0*
__employee_124__ 124
                       xe-0/0/22.0*
__employee_125__ 125
                       xe-0/0/22.0*
__employee_126__ 126
                       xe-0/0/22.0*
```

```
__employee_127__ 127
                        xe-0/0/22.0*
__employee_128__ 128
                        xe-0/0/22.0*
__employee_129__ 129
                        xe-0/0/22.0*
__employee_130__ 130
                        xe-0/0/22.0*
```

Meaning

The sample output shows the VLANs configured on the switch. The series of tagged VLANs is displayed: **__employee__120__** through **__employee_130__**. Each of the tagged VLANs is configured on the trunk interface **xe-0/0/22.0**. The asterisk (*****) next to the interface name indicates that the interface is **UP**.

When a series of VLANs is created using the `vlan-range` statement, the VLAN names are preceded and followed by a double underscore.

Verifying That a Series of Tagged VLANs Has Been Created on an EX Series Switch

IN THIS SECTION

Purpose

Verify that a series of tagged VLANs is created on the switch.

Action

Display the VLANs in the ascending order of their VLAN ID:

```
user@switch> show vlans sort-by tag

Name             Tag        Interfaces
__employee_120__  120
                            ge-0/0/22.0*
__employee_121__  121
                            ge-0/0/22.0*
__employee_122__  122
                            ge-0/0/22.0*
__employee_123__  123
                            ge-0/0/22.0*
__employee_124__  124
                            ge-0/0/22.0*
__employee_125__  125
                            ge-0/0/22.0*
__employee_126__  126
                            ge-0/0/22.0*
__employee_127__  127
                            ge-0/0/22.0*
__employee_128__  128
                            ge-0/0/22.0*
__employee_129__  129
                            ge-0/0/22.0*
__employee_130__  130
                            ge-0/0/22.0*
```

Display the VLANs by the alphabetical order of the VLAN name:

```
user@switch> show vlans sort-by name

Name             Tag   Interfaces

__employee_120__ 120
                       ge-0/0/22.0*
__employee_121__ 121
                       ge-0/0/22.0*
__employee_122__ 122
```

```
                      ge-0/0/22.0*
__employee_123__ 123
                      ge-0/0/22.0*
__employee_124__ 124
                      ge-0/0/22.0*
__employee_125__ 125
                      ge-0/0/22.0*
__employee_126__ 126
                      ge-0/0/22.0*
__employee_127__ 127
                      ge-0/0/22.0*
__employee_128__ 128
                      ge-0/0/22.0*
__employee_129__ 129
                      ge-0/0/22.0*
__employee_130__ 130
                      ge-0/0/22.0*
```

Display the VLANs by specifying the VLAN-range name (here, the VLAN-range name is **employee**):

```
user@switch> show vlans employee

Name             Tag   Interfaces

__employee_120__ 120
                       ge-0/0/22.0*
__employee_121__ 121
                       ge-0/0/22.0*
__employee_122__ 122
                       ge-0/0/22.0*
__employee_123__ 123
                       ge-0/0/22.0*
__employee_124__ 124
                       ge-0/0/22.0*
__employee_125__ 125
                       ge-0/0/22.0*
__employee_126__ 126
                       ge-0/0/22.0*
__employee_127__ 127
                       ge-0/0/22.0*
__employee_128__ 128
                       ge-0/0/22.0*
```

```
__employee_129__ 129
                        ge-0/0/22.0*
__employee_130__ 130
                        ge-0/0/22.0*
```

Meaning

The sample output shows the VLANs configured on the switch. The series of tagged VLANs is displayed: **__employee__120__** through **__employee_130__**. Each of the tagged VLANs is configured on the trunk interface **ge-0/0/22.0**. The asterisk (*****) beside the interface name indicates that the interface is **UP**.

When a series of VLANs is created using the **vlan-range** statement, the VLAN names are prefixed and suffixed with a double underscore.

Configuring Double-Tagged VLANs on Layer 3 Logical Interfaces

Junos OS supports a subset of the IEEE 802.1Q standard for channelizing an Ethernet interface into multiple logical interfaces, allowing many hosts to be connected to the same switch but preventing them from being in the same routing or bridging domain. When an Ethernet LAN is divided into VLANs, each VLAN is identified by a unique 802.1Q tag. The tag is applied to all frames so that network nodes receiving the frames can detect which VLAN the frames belong to.

You can configure double VLAN tags (that is, an inner and an outer tag) on a Layer 3 logical interface (sometimes called a “Layer 3 subinterface”).

Support for double-tagging VLANs on Layer 3 logical interfaces includes:

- Configuration of an IPv4, an IPv6, or an `mpls` family on the logical interface
- Configuration over an aggregated Ethernet interface
- Configuration of multiple logical interfaces on a single physical interface

NOTE: This feature does not include support for the following:

- VLAN rewrite (`input-vlan-map` or `output-vlan-map`)
- TPID configuration (on physical or logical interfaces)
- `native-inner-vlan-id`; `outer-vlan-id-list`; `inner-vlan-id-list`; or `vlan-id-range`

To configure a double-tagged Layer 3 logical interface:

1. Apply flexible VLAN tagging to the physical interface:

```
[edit]
user@switch# set interfaces interface-name flexible-vlan-tagging
```

2. Configure inner and outer VLAN tags on the logical interface:

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

3. Set the family type and, if needed, the address on the logical interface:

```
[edit]
user@switch# set interfaces interface-name unit logical-unit-number family family-type
address address
```

SEE ALSO

Configuring VLANs on Switches with Enhanced Layer 2 Support | 150

Stacking a VLAN Tag

To stack a VLAN tag on all tagged frames entering or exiting the interface, include the **push**, **vlan-id**, and `tag-protocol-id` statements in the input VLAN map or the output VLAN map:

```
input-vlan-map input-vlan-map {
    push;
    vlan-id number;
    tag-protocol-id tpid;
}
output-vlan-map {
    push;
    tag-protocol-id tpid;
}
```

You can include these statements at the following hierarchy levels:

- [edit interfaces *interface-name* unit *logical-unit-number*]
- [edit interfaces *interface-name* unit *logical-unit-number*]
- [edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number*]
- [edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number*]

The VLAN IDs you define in the input VLAN maps are stacked on top of the VLAN ID bound to the logical interface.

All TPIDs you include in input and output VLAN maps must be among those you specify at the [edit interfaces *interface-name* ether-options ethernet-switch-profile tag-protocol-id [*tpids*]] hierarchy level.

Rewriting a VLAN Tag and Adding a New Tag

On Ethernet IQ, IQ2 and IQ2-E interfaces, on MX Series router Gigabit Ethernet, Tri-Rate Ethernet copper, and 10-Gigabit Ethernet interfaces, on aggregated Ethernet interfaces using Gigabit Ethernet IQ2 and IQ2-E or 10-Gigabit Ethernet PICs on MX Series routers, and on Gigabit Ethernet and 10-Gigabit Ethernet interfaces on EX Series switches, to replace the outer VLAN tag of the incoming frame with a user-specified VLAN tag value, include the swap-push statement in the input VLAN map or output VLAN map:

```
swap-push
```

A user-specified outer VLAN tag is pushed in front. The outer tag becomes an inner tag in the final frame. The stacked and rewriting Gigabit-Ethernet VLAN Tags are also referred to as Q-in-Q tunneling.

You can include this statement at the following hierarchy levels:

- [edit interfaces *interface-name* unit *logical-unit-number* input-vlan-map]
- [edit interfaces *interface-name* unit *logical-unit-number* output-vlan-map]
- [edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number* input-vlan-map]
- [edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number* output-vlan-map]

SEE ALSO

input-vlan-map

output-vlan-map

swap-push

unit

Ethernet Interfaces User Guide for Routing Devices

Rewriting the Inner and Outer VLAN Tags

On Ethernet IQ, IQ2 and IQ2-E interfaces, on MX Series router Gigabit Ethernet, Tri-Rate Ethernet copper, and 10-Gigabit Ethernet interfaces, and on aggregated Ethernet interfaces using Gigabit Ethernet IQ2 and IQ2-E or 10-Gigabit Ethernet PICs on MX Series routers, to replace both the inner and the outer VLAN tags of the incoming frame with a user-specified VLAN tag value, include the `swap-swap` statement in the input VLAN map or output VLAN map: The stacked and rewriting Gigabit-Ethernet VLAN Tags are also referred to as Q-in-Q tunneling.

```
swap-swap;
```

You can include this statement at the following hierarchy levels:

- [edit interfaces *interface-name* unit *logical-unit-number* input-vlan-map]
- [edit interfaces *interface-name* unit *logical-unit-number* output-vlan-map]
- [edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number* input-vlan-map]
- [edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number* output-vlan-map]

SEE ALSO

Ethernet Interfaces User Guide for Routing Devices

Rewriting the VLAN Tag on Tagged Frames

To rewrite the VLAN tag on all tagged frames entering the interface to a specified VLAN ID and TPID, include the `swap`, `tag-protocol-id`, and `vlan-id` statements in the input VLAN map:

```
input-vlan-map {
    swap;
    vlan-id number;
    tag-protocol-id tpid;
}
```

To rewrite the VLAN tag on all tagged frames exiting the interface to a specified VLAN ID and TPID, include the `swap` and `tag-protocol-id` statements in the output VLAN map:

```
output-vlan-map {
    swap;
    vlan-id number;
    tag-protocol-id tpid;
}
```

You can include these statements at the following hierarchy levels:

- `[edit interfaces interface-name unit logical-unit-number input-vlan-map]`
- `[edit logical-systems logical-system-name interfaces interface-name unit logical-unit-number input-vlan-map]`

You cannot include both the `swap` statement and the `vlan-id` statement in the output VLAN map configuration. If you include the `swap` statement in the configuration, the VLAN ID in outgoing frames is rewritten to the VLAN ID bound to the logical interface. For more information about binding a VLAN ID to the logical interface, see "802.1Q VLANs Overview" on page 274.

The swap operation works on the outer tag only, whether or not you include the `stacked-vlan-tagging` statement in the configuration. For more information, see Examples: Stacking and Rewriting Gigabit Ethernet IQ VLAN Tags.

SEE ALSO

- Ethernet Interfaces User Guide for Routing Devices

Configuring VLAN Translation with a VLAN ID List

IN THIS SECTION

In many cases, the VLAN identifiers on the frames of an interface's packets are not correct. VLAN translation, or VLAN rewrite, allows you to configure bidirectional VLAN identifier translation with a list on frames arriving on and leaving from a logical interface. This lets you use unique VLAN identifiers internally and maintain legacy VLAN identifiers on logical interfaces.

To perform VLAN translation on the packets on a trunk interface, insert the `vlan-rewrite` statement at the `[edit interfaces` *`interface-name`* `unit` *`unit-number`*`]` hierarchy level. You must also include the `interface-mode trunk` statement within the `[edit interfaces` *`interface-name`* `unit` *`unit-number`* `family ethernet-switching]` hierarchy because VLAN translation is only supported on trunk interfaces. The reverse translation takes place on traffic exiting the interface. In other words, if VLAN 200 is translated to 500 on traffic entering the interface, VLAN 500 is translated to VLAN 200 on traffic leaving the interface.

Use VLAN ID translation to confirm platform and release support for specific features.

Review the "Platform-Specific VLAN Translation Configuration Behavior" on page 364 section for notes related to your platform.

NOTE:

- You can configure either flexible VLAN tagging or trunk mode on interfaces. VLAN translation does not support both.

The following example translates incoming trunk packets from VLAN identifier 200 to 500 and 201 to 501 (other valid VLAN identifiers are not affected):

```
[edit interfaces ge-1/0/1]
unit 0 {
    ... # Other logical interface statements
    family ethernet-switching {
        interface-mode trunk # Translation is only for trunks
        vlan {
            members 500-501;
        }
```

```
        vlan-rewrite {
            translate 200 500;
            translate 201 501;
        }
    ... # Other ethernet-switching statements
    }
}
```

NOTE: This example also translates frame VLANs from 500 to 200 and 501 to 201 on egress.

Platform-Specific VLAN Translation Configuration Behavior

Use Feature Explorer to confirm platform and release support for specific features.

Use the following table to review platform-specific behavior for your platform:

Platform	Difference
ACX Series	• On ACX7000 device that support VLAN translation configuration, access and trunk type logical interfaces are not supported.

Configuring VLAN Translation on Security Devices

VLAN translation allow service providers to create a Layer 2 Ethernet connection between two customer sites. Providers can segregate different customers' VLAN traffic on a link (for example, if the customers use overlapping VLAN IDs) or bundle different customer VLANs into a single service VLAN. Data centers can use Q-in-Q tunneling to isolate customer traffic within a single site or when customer traffic flows between cloud data centers in different geographic locations.

Before you begin configuring VLAN translation, make sure you have created and configured the necessary customer VLANs on the neighboring switches. See *Configuring VLANs*.

VLAN translation can be done in two ways:

- To configure VLAN translation in VLAN retagging, an enterprise provider style of VLAN translation can be achieved by following CLI configuration:

```
[edit]
user@host#set interfaces intf-name unit 0 family ethernet-switching interface-mode trunk
user@host#set interfaces intf-name unit 0 family ethernet-switching vlan members v1000
user@host#set interfaces intf-name unit 0 family ethernet-switching vlan-rewrite translate
500 1000
```

- To configure VLAN translation in Q-in-Q, a service provider style of VLAN translation can be achieved by following CLI configuration:

```
[edit]
user@host#set interfaces intf-name flexible-vlan-tagging
user@host#set interfaces intf-name encapsulation extended-vlan-bridge
user@host#set interfaces intf-name unit 100 vlan-id 500
user@host#set interfaces intf-name unit 100 input-vlan-map swap
user@host#set interfaces intf-name unit 100 input-vlan-map tag-protocol-id 0x8100
user@host#set interfaces intf-name unit 100 output-vlan-map swap
user@host#set interfaces intf-name unit 100 family ethernet-switching vlan members v1000
```

SEE ALSO

Verifying That Q-in-Q Tunneling Is Working on Switches | 991

Example: Configuring VLAN Retagging for Layer 2 Transparent Mode on a Security Device

IN THIS SECTION

This example shows how to configure VLAN retagging on a Layer 2 trunk interface to selectively screen incoming packets and redirect them to a security device without affecting other VLAN traffic.

Requirements

Before you begin, determine the mapping you want to include for the VLAN retagging. See "Understanding VLAN Retagging on Security Devices" on page 1075.

Overview

In this example, you create a Layer 2 trunk interface called ge-3/0/0 and configure it to receive packets with VLAN identifiers 1 through 10. Packets that arrive on the interface with VLAN identifier 11 are retagged with VLAN identifier 2. Before exiting the trunk interface, VLAN identifier 2 in the retagged packets is replaced with VLAN identifier 11. All VLAN identifiers in the retagged packets change back when you exit the trunk interface.

Configuration

IN THIS SECTION

Procedure

Step-by-Step Procedure

To configure VLAN retagging on a Layer 2 trunk interface:

1. Create a Layer 2 trunk interface.

```
[edit]
user@host#set interfaces ge-3/0/0 unit 0 family ethernet-switching interface-mode trunk vlan
members 1-10
```

2. Configure VLAN retagging.

```
[edit]
user@host#set interfaces ge-3/0/0 unit 0 family ethernet-switching vlan-rewrite translate 11 2
```

3. If you are done configuring the device, commit the configuration.

```
[edit]
user@host# commit
```

Verification

To verify the configuration is working properly, enter the `show interfaces ge-3/0/0` command.

SEE ALSO

Ethernet Switching and Layer 2 Transparent Mode Overview | 5

Example: Configuring Layer 2 Logical Interfaces on Security Devices | 1031

Configuring Inner and Outer TPIDs and VLAN IDs

For some rewrite operations, you must configure the inner or outer TPID values and inner or outer VLAN ID values. These values can be applied to either the input VLAN map or the output VLAN map.

On Ethernet IQ, IQ2, and IQ2-E interfaces; on MX Series router Gigabit Ethernet, Tri-Rate Ethernet copper, and 10-Gigabit Ethernet interfaces; and on aggregated Ethernet interfaces using Gigabit Ethernet IQ2 and IQ2-E or 10-Gigabit Ethernet PICs on MX Series routers, to configure the inner TPID, include the `inner-tag-protocol-id` statement:

```
inner-tag-protocol-id tpid;
```

You can include this statement at the following hierarchy levels:

- [edit interfaces *interface-name* unit *logical-unit-number* input-vlan-map]
- [edit interfaces *interface-name* unit *logical-unit-number* output-vlan-map]
- [edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number* input-vlan-map]
- [edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number* output-vlan-map]

For the inner VLAN ID, include the `inner-vlan-id` statement. For the outer TPID, include the `tag-protocol-id` statement. For the outer VLAN ID, include the `vlan-id` statement:

```
input-vlan-map {
    (pop | pop-pop | pop-swap | push | push-push | swap | swap-push | swap-swap);
    inner-tag-protocol-id tpid;
    inner-vlan-id number;
    tag-protocol-id tpid;
    vlan-id number;
}
output-vlan-map {
    (pop | pop-pop | pop-swap | push | push-push | swap | swap-push | swap-swap);
    inner-tag-protocol-id tpid;
    inner-vlan-id number;
    tag-protocol-id tpid;
    vlan-id number;
}
```

For aggregated Ethernet interfaces using Gigabit Ethernet IQ interfaces, include the `tag-protocol-id` statement for the outer TPID. For the outer VLAN ID, include the `vlan-id` statement:

```
input-vlan-map {
    (pop | push | swap);
    tag-protocol-id tpid;
    vlan-id number;
}
output-vlan-map {
    (pop | push | swap);
    tag-protocol-id tpid;
    vlan-id number;
}
```

You can include these statements at the following hierarchy levels:

- [edit interfaces *interface-name* unit *logical-unit-number*]
- [edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number*]

The VLAN IDs you define in the input VLAN maps are stacked on top of the VLAN ID bound to the logical interface. For more information about binding a VLAN ID to the logical interface, see "802.1Q VLANs Overview" on page 274.

All TPIDs you include in input and output VLAN maps must be among those you specify at the `[edit interfaces interface-name ether-options ethernet-switch-profile tag-protocol-id [ tpids ]]` hierarchy level.

Table 61 on page 369 and Table 62 on page 370 specify when these statements are required. Table 61 on page 369 indicates valid statement combinations for rewrite operations for the input VLAN map. “No” means the statement must not be included in the input VLAN map for the rewrite operation. “Optional” means the statement may be optionally specified for the rewrite operation in the input VLAN map. “Any” means that you must include the `vlan-id` statement, `tag-protocol-id` statement, `inner-vlan-id` statement, or `inner-tag-protocol-id` statement.

Table 61: Rewrite Operations and Statement Usage for Input VLAN Maps

	Input VLAN Map Statements			
Rewrite Operation	**vlan-id**	**tag-protocol-id**	**inner-vlan-id**	**inner-tag-protocol-id**
push	Optional	Optional	No	No
pop	No	No	No	No
swap	Any	Any	No	No
push-push	Optional	Optional	Optional	optional
swap-push	Optional	Optional	Any	Any
swap-swap	Optional	Optional	Any	Any
pop-swap	No	No	Any	Any
pop-pop	No	No	No	No

Table 62 on page 370 indicates valid statement combinations for rewrite operations for the output VLAN map. “No” means the statement must not be included in the output VLAN map for the rewrite operation. “Optional” means the statement may be optionally specified for the rewrite operation in the output VLAN map.

Table 62: Rewrite Operations and Statement Usage for Output VLAN Maps

	Output VLAN Map Statements			
Rewrite Operation	**vlan-id**	**tag-protocol-id**	**inner-vlan-id**	**inner-tag-protocol-id**
push	No	Optional	No	No
pop	No	No	No	No
swap	No	Optional	No	No
push-push	No	Optional	No	Optional
swap-push	No	Optional	No	Optional
swap-swap	No	Optional	No	Optional
pop-swap	No	No	No	Optional
pop-pop	No	No	No	No

The following examples use Table 61 on page 369 and Table 62 on page 370 and show how the **pop-swap** operation can be configured in an input VLAN map and an output VLAN map:

Input VLAN Map with inner-vlan-id Statement, Output VLAN Map with Optional inner-tag-protocol-id Statement

```
[edit interfaces interface-name unit logical-unit-number]
input-vlan-map {
    pop-swap;
    inner-vlan-id number;
}
output-vlan-map {
    pop-swap;
    inner-tag-protocol-id tpid;
}
```

Input VLAN Map with inner-tag-protocol-id Statement, Output VLAN Map with Optional inner-tag-protocol-id Statement

```
[edit interfaces interface-name unit logical-unit-number]
input-vlan-map {
    pop-swap;
    inner-tag-protocol-id tpid;
}
output-vlan-map {
    pop-swap;
    inner-tag-protocol-id tpid;
}
```

Input VLAN Map with inner-tag-protocol-id and inner-vlan-id Statements

```
[edit interfaces interface-name unit logical-unit-number]
input-vlan-map {
    pop-swap;
    inner-vlan-id number;
    inner-tag-protocol-id tpid;
}
```

RELATED DOCUMENTATION

Using the Enhanced Layer 2 Software CLI | 15

Example: Configuring Redundant Trunk Links for Faster Recovery on Devices with ELS Support | 998

16
CHAPTER

Stacking and Rewriting Gigabit Ethernet VLAN Tags

IN THIS CHAPTER

Stacking and Rewriting Gigabit Ethernet VLAN Tags Overview

Stacking and rewriting VLAN tags, commonly known as Q-in-Q tunneling, enables the use of an additional (outer) VLAN tag to differentiate between customer edge (CE) routers that share the same VLAN ID.

You can stack and rewrite VLAN tags on the following interfaces:

- Gigabit Ethernet
- Gigabit Ethernet IQ
- 10-Gigabit Ethernet LAN/WAN PIC
- 40-Gigabit Ethernet MIC
- 100-Gigabit Ethernet MIC
- Gigabit Ethernet IQ2 and IQ2-E
- 10-Gigabit Ethernet IQ2 and IQ2-E interfaces, and MX Series router Gigabit Ethernet Interfaces
- Tri-Rate Ethernet copper, and 10-Gigabit Ethernet interfaces with the VLAN encapsulation type configured to support L2 tunneling protocols such as circuit cross-connect (CCC) or virtual private LAN service (VPLS) (as described in "802.1Q VLANs Overview" on page 274)

Stacking VLAN tags encapsulate multiple VLAN identifiers within a single frame. This approach supports hierarchical segmentation of broadcast domains and enables flexible Layer 2 forwarding. Rewriting VLAN tags modifies these identifiers to ensure that frames are correctly associated with the intended broadcast or bridge domain, either within a virtual switch or across network boundaries. This mechanism is fundamental in Juniper's Layer 2 architecture to maintain traffic isolation and efficient forwarding. In modern networking, the concepts of broadcast domains, bridge domains, and virtual switches play a vital role in segmenting and managing traffic efficiently. Technologies such as Integrated Routing and Bridging (IRB) and VLAN tag manipulation, including stacking (Q-in-Q) and rewriting of Gigabit Ethernet VLAN tags, offer advanced capabilities for traffic isolation, service differentiation, and support for multi-tenant environments.

A broadcast domain is a network segment in which all devices receive broadcast frames sent by any other device within the segment. Routers typically define the boundaries of broadcast domains, as routers do not forward broadcast traffic. Switches and bridges can subdivide networks into multiple broadcast domains using VLANs. While a broadcast domain is a network area where all devices receive broadcast traffic, a bridge domain is a logical Layer 2 construct used to group interfaces for forwarding and flooding decisions, often in virtualized or service provider networks.

A bridge domain is a logical grouping of network interfaces that are bridged together. It operates as a Layer 2 segment where devices can communicate directly without routing. Bridge domains often consist of one or more ports or port-VLAN pairs across multiple devices. For example, For example, an IEEE 802.1Q VLAN can span multiple ports on different devices and each port or VLAN can belong to only one bridge domain.

While a bridge domain represents a logical L2 forwarding construct, a virtual switch implements this functionality by connecting virtual interfaces within the same bridge domain to enable intra-domain communication. It functions similarly to a physical switch but operates in a virtualized environment. Virtual switches manage VLANs and isolate traffic between virtual networks. Within a virtual switch, VLAN IDs are unique—no two bridge domains can share the same VLAN ID.

Each bridge domain can optionally include a routing interface. Integrated Routing and Bridging (IRB) integrates Layer 2 and Layer 3 functionalities within a single interface, allowing it to perform both bridging and routing operations. IRB is especially useful in environments that require routing between VLANs while maintaining Layer 2 connectivity within each VLAN.

When a VLAN tag is pushed on IQ2 interfaces, 10-Gigabit Ethernet LAN/WAN PIC, 40-Gigabit Ethernet MIC, 100-Gigabit Ethernet MIC, IQ2-E interfaces, and MX Series interfaces, the inner VLAN IEEE 802.1p bits are copied to the IEEE bits of the VLAN or VLANs being pushed. If the original packet is untagged, the IEEE bits of the VLAN or VLANs being pushed are set to 0.

NOTE: When `swap-by-pop` push is configured on the interface and a VLAN tag is swapped, the inner VLAN IEEE 802.1p bits are copied to the IEEE bits of the VLAN being swapped. If swap-by-poppush is not configured on the interface, the VLAN IEEE 802.1 p bits of the VLAN being swapped remain the same.

RELATED DOCUMENTATION

802.1Q VLANs Overview | **274**

Stacking and Rewriting Gigabit Ethernet VLAN Tags

Ethernet Interfaces User Guide for Routing Devices

Stacking and Rewriting Gigabit Ethernet VLAN Tags

SUMMARY

VLAN stacking encapsulates multiple VLAN tags within Ethernet frames, enabling hierarchical tagging and extending the VLAN space. VLAN tag rewriting facilitates dynamic tag translation across network boundaries.

IN THIS SECTION

Stacking and rewriting Gigabit Ethernet VLAN Tags is commonly known as Q-in-Q tunneling. VLAN stacking encapsulates multiple VLAN tags within Ethernet frames, enabling hierarchical tagging and extending the VLAN space. VLAN tag rewriting facilitates dynamic tag translation across network boundaries. When properly configured, these techniques significantly enhance interoperability and traffic engineering across diverse platforms. You can configure rewrite operations to stack (`push`), remove (`pop`), or rewrite (`swap`) tags on single-tagged frames and dual-tagged frames. If a port is untagged, rewrite operations are not supported on any logical interface associated with that port.

VLAN Tag Manipulation

VLAN tagging is a method used to identify and segregate network traffic based on VLAN IDs (Identifiers). Tag manipulation allows for various operations on VLAN tags. You can configure the following VLAN rewrite operations:

- `pop`—Remove a VLAN tag from the top of the VLAN tag stack. The outer VLAN tag of the frame is removed.
- `pop-pop`—For Ethernet IQ2, 10-Gigabit Ethernet LAN/WAN PIC, and IQ2-E interfaces, remove both the outer and inner VLAN tags of the frame.

- pop-swap—For Ethernet IQ2, 10-Gigabit Ethernet LAN/WAN PIC, and IQ2-E interfaces, remove the outer VLAN tag of the frame, and replace the inner VLAN tag of the frame with a user-specified VLAN tag value. The original inner tag becomes the outer tag in the final frame.
- push—Add a new VLAN tag to the top of the VLAN stack. An outer VLAN tag is pushed in front of the existing VLAN tag.
- push-push—For Ethernet IQ2, 10-Gigabit Ethernet LAN/WAN PIC, and IQ2-E interfaces, push two VLAN tags in front of the frame.
- swap-push—For Ethernet IQ2, 10-Gigabit Ethernet LAN/WAN PIC, and IQ2-E interfaces, replace the outer VLAN tag of the frame with a user-specified VLAN tag value. A user-specified outer VLAN tag is pushed in front. The original outer tag becomes an inner tag in the final frame.
- swap-swap—For Ethernet IQ2, 10-Gigabit Ethernet LAN/WAN PIC, and IQ2-E interfaces, replace both the inner and the outer VLAN tags of the incoming frame with a user-specified VLAN tag value.

You configure VLAN rewrite operations for logical interfaces in the input VLAN map for incoming frames and in the output VLAN map for outgoing frames. To configure the input VLAN map, include the input-vlan-map statement:

```
input-vlan-map {
    ...interface-specific configuration...
}
```

NOTE: An interface can receive a frame, or the frame can originate internally in the system. For example, through the input-vlan-map statement.

To configure the output VLAN map, include the output-vlan-map statement:

```
output-vlan-map {
    ...interface-specific configuration...
}
```

You can include both statements at the following hierarchy levels:

- [edit interfaces *interface-name* unit *logical-unit-number*]
- [edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number*]

VLAN Rewrite Operation Types

The type of VLAN rewrite operation permitted depends upon whether the frame is single-tagged or dual-tagged. https://www.juniper.net/documentation/us/en/software/junos/multicast-l2/topics/task/interfaces-rewriting-the-vlan-tag.html and https://www.juniper.net/documentation/us/en/software/junos/multicast-l2/topics/topic-map/interfaces-rewriting-vlan-tag-untagged-frames.html describe supported rewrite operations and whether they can be applied to single-tagged frames or dual-tagged frames. The table also indicates the number of tags being added or removed during the operation.

Table 63: Rewrite Operations on Untagged, Single-Tagged, and Dual-Tagged Frames

Rewrite Operation	Untagged	Single-Tagged	Dual-Tagged	Number of Tags
pop	No	Yes	Yes	– 1
push	Sometimes	Yes	Yes	+1
swap	No	Yes	Yes	0
push-push	Sometimes	Yes	Yes	+2
swap-push	No	Yes	Yes	+1
swap-swap	No	No	Yes	0
pop-pop	No	No	Yes	– 2
pop-swap	No	No	Yes	– 1

The rewrite operations `push` and `push-push` can be valid in certain circumstances on frames that are not tagged. For example, a single-tagged logical interface (interface 1) and a dual-tagged logical interface (interface 2) have the following configurations:

Interface 1

```
[edit interfaces interface-name unit logical-unit-number]
input-vlan-map {
    pop;
```

```
}
output-vlan-map {
    push;
}
```

Interface 2

```
[edit interfaces interface-name unit logical-unit-number]
input-vlan-map {
    pop-pop;
}
output-vlan-map {
    push-push;
}
```

When a frame is received on the interface as a result of the `input-vlan-map` operation, the frame is not tagged. As it goes out of the second interface, the `output-vlan-map` operation `push-push` is applied to it. The resulting frame will be dual-tagged at the logical interface output.

Depending on the VLAN rewrite operation, you configure the rewrite operation for the interface in the input VLAN map, the output VLAN map, or in both the input VLAN map and the output VLAN map. The following table describes what rewrite operation combinations you can configure. *None* means that no rewrite operation is specified for the VLAN map.

Table 64: Applying Rewrite Operations to VLAN Maps

	Output VLAN Map								
Input VLAN Map	none	push	pop	swap	push-push	swap-push	swap-swap	pop-pop	swap-pop
none	Yes	No	No	Yes	No	No	Yes	No	No
push	No	No	Yes	No	No	No	No	No	No
pop	No	Yes	No	No	No	No	No	No	No
swap	Yes	No	No	Yes	No	No	No	No	No

Table 64: Applying Rewrite Operations to VLAN Maps *(Continued)*

	Output VLAN Map								
Input VLAN Map	none	push	pop	swap	push-push	swap-push	swap-swap	pop-pop	swap-pop
push-push	No	No	No	No	No	No	No	Yes	No
swap-push	No	No	No	No	No	No	No	No	Yes
swap-swap	Yes	No	No	No	No	No	Yes	No	No
pop-pop	No	No	No	No	Yes	No	No	No	No
pop-swap	No	No	No	No	No	Yes	No	No	No

Before you configure the VLAN rewrite operation:

- Ensure that the VLAN rewrite operation is valid and applied to the input or output VLAN maps.
- Check if the rewrite operation requires you to include statements to configure the inner and outer TPIDs and inner and outer VLAN IDs in the input or output VLAN maps.

For information about configuring inner and outer TPIDs and inner and outer VLAN IDs, see "Configuring Inner and Outer TPIDs and VLAN IDs" on page 394.

Table 65:

Automatic and Derived VLAN Translations

Automatic and Derived VLAN translation dynamically maps VLAN IDs based on predefined rules or contextual information. This enables seamless VLAN stacking (Q-in-Q) and Gigabit Ethernet VLAN tag rewriting, supporting scalable, multi-tenant network segmentation and interoperability. VLAN translation involves changing the VLAN ID of a packet as it traverses the network. This translation can be configured manually or derived automatically. Virtual Private LAN Service (VPLS) is set up to use Automatic and Derived VLAN translation in such a way that the customer VLANs can be automatically matched to the provider VLANs .To provide a VPLS service, the network administrator or service provider (referred to as the operator) is responsible for configuring and managing the VPLS instance. The operator typically sets up the VPLS instance and adds logical interfaces to it. In most cases, these logical interfaces carry traffic with similar VLAN tags. If a logical interface uses VLAN tags that differ from those used by other interfaces in the VPLS domain, the operator can normalize the VLAN tags by applying `vlan-map` commands. VPLS is connected to a bridge domain by configuring a routing instance with the `instance-type` set to `vpls`. For more information, see Configuring-vpls-and-integrated-routing-and-bridging.

In Juniper Networks EVPN (Ethernet VPN) environments, automatic or derived VLAN translation enables seamless handling of overlapping or differing VLAN IDs across the EVPN fabric. VLAN tags are translated at the network edge, allowing multiple customer VLANs, whether they use the same VLAN ID (overlapping VLANs) or different VLAN IDs, to be mapped appropriately within the EVPN-VXLAN fabric. This ensures proper traffic segregation and forwarding. For more information, see Overlapping VLAN Support Using VLAN Translation in EVPN-VXLAN Networks.

Bridge Domain Modes

The Bridge Domain can be configured in one of following modes:

Table 66: Bridge Domain Modes

Mode	Description
Default	Valid on MX Series routers. You must configure a VLAN map on an IFL to normalize VLAN IDs. VLAN tags in packets entering or exiting the bridge domain are transparent and depend on the IFL's VLAN configuration and any applied VLAN map.

Table 66: Bridge Domain Modes *(Continued)*

Mode	Description
VLAN none	Normalizes packets to ensure they are untagged. Packets entering or exiting the bridge domain do not carry VLAN tags. This is achieved using automatic VLAN maps (push/pop/swap). Packets from the CE (Customer Edge) to the core network are sent without any VLAN tag, and packets arriving from the core network are also untagged. Example: `vlan-id none`
VLAN <vid>	Normalizes packets to ensure they carry a single, specified VLAN ID <vid>. Packets entering or exiting the bridge domain are tagged with the configured VLAN ID. Packets from the CE to the core network carry the specified VLAN ID, and packets from the core network arrive with the same VLAN ID. Example: `vlan-id 100`
VLAN <dual-tags>	Normalize the packet to include the specified VLAN tag stack, like outer and inner VLAN IDs. Packets that enter or exit the BD will have dual VLAN tag. These VLAN tags are the BD's configured VLAN tags. Packets from the CE to the core network carry both specified VLAN tags, and packets from the core network arrive with the same dual-tag structure. Example: `vlan-id 100, 200`

Table 66: Bridge Domain Modes *(Continued)*

Mode	Description
VLAN all	Transparently carries any VLAN ID present in the packet. Packets entering or exiting the bridge domain retain their VLAN tag. For single-tagged IFLs, this is the single VLAN ID; for double-tagged IFLs, it is the inner VLAN ID. The VLAN ID is used to qualify learned MAC addresses. VLAN tags are preserved and switched transparently across the network. Example: `vlan-id all` **NOTE**: `vlan-id all` is typically used to handle traffic from multiple VLANs without modifying their tags.

Bridge Domain Mode Examples

Example of VLAN none mode

```
interfaces {
    ge-1/0/0 {
        encapsulation ethernet-bridge;
        unit 0;
    }
    ge-1/0/1 {
        encapsulation flexible-ethernet-services;
        flexible-VLAN-tagging;
        unit 0 {
            encapsulation VLAN-bridge;
            VLAN-id 100;
        }
    }
    ge-1/0/2 {
        encapsulation flexible-ethernet-services;
        flexible-VLAN-tagging;
        unit 0 {
```

```
            encapsulation VLAN-bridge;
            VLAN-tags outer 100 inner 200;
        }
    }
}
bridge-domains {
    blue {
        domain-type bridge;
+       VLAN-id none;
        interface ge-1/0/0.0;
        interface ge-1/0/1.0;
        interface ge-1/0/2.0;
   }
}
Automatic VLAN-maps performed for above IFLs
IFL          input     output
--------------------------------
ge-1/0/0.0   none      none
ge-1/0/1.0   pop       push
ge-1/0/2.0   pop-pop   push-push
```

Example of VLAN single mode

```
interfaces {
    ge-1/0/0 {
        encapsulation flexible-ethernet-services;
        flexible-VLAN-tagging;
        unit 0 {
            encapsulation VLAN-bridge;
            VLAN-id 100;
        }
    }
    ge-1/0/1 {
        encapsulation flexible-ethernet-services;
        flexible-VLAN-tagging;
        unit 0 {
            encapsulation VLAN-bridge;
            VLAN-id 200;
        }
    }
    ge-1/0/2 {
       1encapsulation flexible-ethernet-services;
```

```
        flexible-VLAN-tagging;
        unit 0 {
            encapsulation VLAN-bridge;
            VLAN-tags outer 2000 inner 100;
        }
    }
    ge-1/0/3 {
        encapsulation flexible-ethernet-services;
        flexible-VLAN-tagging;
        unit 0 {
            encapsulation VLAN-bridge;
            VLAN-tags outer 2000 inner 200;
        }
    }
}
bridge-domains {
    VLAN-100 {
        domain-type bridge;
+       VLAN-id 100;
        interface ge-1/0/0.0;
        interface ge-1/0/1.0;
        interface ge-1/0/2.0;
        interface ge-1/0/3.0;
    }
}
Automatic VLAN-maps performed for above IFLs
IFL          input     output
--------------------------------
ge-1/0/0.0   none      none
ge-1/0/1.0   swap      swap
ge-1/0/2.0   pop       push
ge-1/0/3.0   pop-swap  swap-push
```

Example of VLAN dual mode

```
interfaces {
    ge-1/0/0 {
        encapsulation flexible-ethernet-services;
        flexible-VLAN-tagging;
        unit 0 {
            encapsulation VLAN-bridge;
            VLAN-tags outer 100 inner 200;
```

```
        }
    }
    ge-1/0/1 {
        encapsulation flexible-ethernet-services;
        flexible-VLAN-tagging;
        unit 0 {
            encapsulation VLAN-bridge;
            VLAN-id 200;
        }
    }
    ge-1/0/2 {
        encapsulation flexible-ethernet-services;
        flexible-VLAN-tagging;
        unit 0 {
            encapsulation VLAN-bridge;
            VLAN-tags outer 100 inner 300;
        }
    }
    ge-1/0/3 {
        encapsulation flexible-ethernet-services;
        flexible-VLAN-tagging;
        unit 0 {
            encapsulation VLAN-bridge;
            VLAN-tags outer 1000 inner 2000;
        }
    }
}
bridge-domains {
    VLAN-100-200 {
        domain-type bridge;
+       VLAN-tags outer 100 inner 200;
        interface ge-1/0/0.0;
        interface ge-1/0/1.0;
        interface ge-1/0/2.0;
        interface ge-1/0/3.0;
    }
}

Automatic VLAN-maps performed for above IFLs
IFL          input     output
--------------------------------
ge-1/0/0.0   none      none
ge-1/0/1.0   push      pop
```

```
ge-1/0/2.0   nop-swap  nop-swap
ge-1/0/3.0   swap-swap swap-swap
```

Example of VLAN all mode

```
interfaces {
    ge-1/0/0 {
        encapsulation flexible-ethernet-services;
        flexible-VLAN-tagging;
        unit 0 {
            encapsulation VLAN-bridge;
            VLAN-id-range 1-4094;
        }
    }
    ge-1/0/1 {
        encapsulation flexible-ethernet-services;
        flexible-VLAN-tagging;
        unit 0 {
            encapsulation VLAN-bridge;
            VLAN-tags outer 200 inner-range 1-4094;
        }
    }
}
bridge-domains {
    all-customer-VLANs {
        domain-type bridge;
+       VLAN-id all;
        interface ge-1/0/0.0;
        interface ge-1/0/1.0;
    }
}
Automatic VLAN-maps performed for above IFLs
IFL          input     output
--------------------------------
ge-1/0/0.0   none      none
ge-1/0/1.0   pop       push
```

IRB Configuration on ACX, MX, and PTX Platforms

When configuring Integrated Routing and Bridging (IRB) with VLAN translation, it's important to note that the `vlan-id all` configuration is not supported with IRB on any platform. This is because IRB must be associated with a single VLAN ID.

- ACX platforms typically support IRB with explicitly defined VLAN IDs. The `vlan-id all` configuration is not supported.
- MX platforms support advanced IRB configurations, including per-VLAN IRB interfaces. However, `vlan-id all` is still not supported with IRB.
- PTX platforms are primarily designed for high-performance routing. They may have limited support for IRB with `vlan-id all`.

VPLS, EVPN, and ELAN/ELINE Handling on ACX, MX, and PTX Platforms

The ACX, MX, and PTX platforms support VPLS, EVPN, and ELAN/ELINE differently based on their architecture, capabilities, and intended use cases:

Table 67: Handling VPLS, EVPN, and ELAN on ACX, MX, and PTX

	VPLS (Virtual Private LAN Service)	EVPN (Ethernet VPN)	ELAN (Ethernet LAN)/ ELINE (Ethernet Line)
ACX	Supports VPLS for multipoint Layer 2 VPN services across a packet-switched network. Ideal for service providers extending LAN services over large areas.	Supports EVPN for Layer 2 and Layer 3 VPN services with features like MAC learning, redundancy, and scalability.	Supports ELAN (multipoint) and ELINE (point-to-point) services for flexible Layer 2 connectivity between customer sites.
MX	Supports advanced VPLS features, including hierarchical VPLS (H-VPLS), for scalable and simplified deployments in service provider networks.	Provides robust EVPN support for both data center interconnects and service provider networks, using MPLS and VXLAN encapsulations.	Offers comprehensive ELAN and ELINE support with high performance and reliability, suitable for services with strict SLAs.

Table 67: Handling VPLS, EVPN, and ELAN on ACX, MX, and PTX *(Continued)*

	VPLS (Virtual Private LAN Service)	EVPN (Ethernet VPN)	ELAN (Ethernet LAN)/ ELINE (Ethernet Line)
PTX	Limited VPLS support; primarily focused on high-capacity IP/MPLS routing in core networks.	Supports EVPN in high-throughput, large-scale environments such as data center interconnects and service provider networks.	Supports ELAN and ELINE, but optimized for high-performance Ethernet transport in backbone and core networks.

RELATED DOCUMENTATION

- Stacking and Rewriting Gigabit Ethernet VLAN Tags Overview | 374
- Understanding swap-by-poppush | 422
- *swap-by-poppush*
- Ethernet Interfaces User Guide for Routing Devices

Configuring Frames with Particular TPIDs to Be Processed as Tagged Frames

IN THIS SECTION

- Platform-Specific Frames with TPIDs Behavior | 390

For Gigabit Ethernet IQ interfaces, aggregated Ethernet with Gigabit Ethernet IQ interfaces, Gigabit Ethernet PICs with SFPs Tri-Rate Ethernet copper, and 10-Gigabit Ethernet interfaces, you can configure frames with particular TPIDs to be processed as tagged frames. To do this, you specify up to eight IEEE 802.1Q TPID values per port; a frame with any of the specified TPIDs is processed as a tagged frame; however, with IQ2 and IQ2-E interfaces, only the first four IEEE 802.1Q TPID values per port are

supported. The stacked and rewriting Gigabit-Ethernet VLAN Tags are also referred to as Q-in-Q tunneling. To configure the TPID values, include the `tag-protocol-id` statement:

```
tag-protocol-id [ tpids ];
```

Use Feature Explorer to confirm platform and release support for specific features.

Review the "Platform-Specific Frames with TPIDs Behavior" on page 390 section for notes related to your platform.

You can include this statement at the following hierarchy levels:

- `[edit interfaces interface-name gigether-options ethernet-switch-profile]`
- `[edit interfaces interface-name aggregated-ether-options ethernet-switch-profile]`
- `[edit interfaces interface-name ether-options ethernet-switch-profile]`

  `[edit interfaces interface-name aggregated-ether-options ethernet-switch-profile]`

All TPIDs you include in input and output VLAN maps must be among those you specify at the `[edit interfaces interface-name gigether-options ethernet-switch-profile tag-protocol-id [ tpids ]]` or `[edit interfaces interface-name aggregated-ether-options ethernet-switch-profile tag-protocol-id [ tpids ]]` hierarchy level.

Platform-Specific Frames with TPIDs Behavior

Use Feature Explorer to confirm platform and release support for specific features.

Use the following table to review platform-specific behavior for your platform:

Platform	Difference
ACX Series	• On ACX7000 device that supports frames with TPIDs: • Use `0x8100` as default TPID, if you do not mention TPID implicitly. • You can configure 3 other TPIDs per IFD apart from 0x8100.

RELATED DOCUMENTATION

Stacking and Rewriting Gigabit Ethernet VLAN Tags Overview | 374

aggregated-ether-options

ethernet-switch-profile

gigether-options

tag-protocol-id

Ethernet Interfaces User Guide for Routing Devices

Configuring Tag Protocol IDs (TPIDs) on PTX Series Packet Transport Routers

This topic describes how to configure the TPIDs expected to be sent or received on a particular VLAN for PTX Series Packet Transport Routers.

For other types of Juniper Networks Ethernet PICs, you could configure 8 TPIDs per port. However, the PTX Series Packet Transport Routers use MTIP and TL to classify a specific TPID and Ethernet type. For MTIP, you can configure a maximum of 8 TPIDs for each MAC chip.

As a consequence, you can specify the `tag-protocol-id` configuration statement only for the first port (`0`) of a PTX Series Ethernet PIC. If you configure `tag-protocol-id` statements on the other port, the configuration is ignored and a system error is recorded.

For example, the following is a supported configuration:

```
[edit interfaces et-2/0/0]
gigether-options {
    ethernet-switch-profile {
        tag-protocol-id [0x8100 0x9100];
    }
}
```

The `tag-protocol-id` configuration statement supports up to eightTPIDs on port `0` of a given Ethernet PIC. All eight TPIDs are populated to the two MTIPs and TLs associated with the Ethernet PIC.

RELATED DOCUMENTATION

Configuring Frames with Particular TPIDs to Be Processed as Tagged Frames | 389

Configuring Flexible VLAN Tagging on PTX Series Packet Transport Routers | 286

Configuring Stacked VLAN Tagging

To configure stacked VLAN tagging for all logical interfaces on a physical interface:

1. In configuration mode, go to the `[edit interfaces interface-name]` hierarchy level.

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

2. Include the `stacked-vlan-tagging` statement.

```
[edit interfaces interface-name]
user@host# set stacked-vlan-tagging
```

If you include the `stacked-vlan-tagging` statement in the configuration, you must configure dual VLAN tags for all logical interfaces on the physical interface. For more information, see "Stacking a VLAN Tag" on page 399.

RELATED DOCUMENTATION

stacked-vlan-tagging

Stacking a VLAN Tag | 399

Ethernet Interfaces User Guide for Routing Devices

Configuring Dual VLAN Tags

IN THIS SECTION

To configure dual VLAN tags on a logical interface, include the `vlan-tags` statement:

```
vlan-tags inner <tpid.>vlan-id outer <tpid.>vlan-id;
```

You can include this statement at the following hierarchy levels:

- `[edit interfaces interface-name unit logical-unit-number]`
- `[edit logical-systems logical-system-name interfaces interface-name unit logical-unit-number]`

Use Feature Explorer to confirm platform and release support for specific features.

Review the "Platform-Specific Dual VLAN Tags Configuration Behavior" on page 393 section for notes related to your platform.

The outer tag VLAN ID range is from 1 through 511 for normal interfaces, and from 512 through 4094 for VLAN CCC or VLAN VPLS interfaces. The inner tag is not restricted.

You must also include the `stacked-vlan-tagging` statement in the configuration. See Examples: Stacking and Rewriting Gigabit Ethernet IQ VLAN Tags.

Platform-Specific Dual VLAN Tags Configuration Behavior

Use Feature Explorer to confirm platform and release support for specific features.

Use the following table to review platform-specific behavior for your platform:

Platform	Difference
ACX Series	• On ACX7000 Series routers that support dual VLAN tags configuration, VLAN ID range restriction for interfaces is not applicable.

RELATED DOCUMENTATION

unit

Ethernet Interfaces User Guide for Routing Devices

Configuring Inner and Outer TPIDs and VLAN IDs

For some rewrite operations, you must configure the inner or outer tag-protocol identifier (TPID) values and inner or outer virtual local area network identifier (VLAN ID) values. These values can be applied to either the input VLAN map or the output VLAN map. The stacked and rewriting Gigabit-Ethernet VLAN Tags are also referred to as Q-in-Q tunneling.

1. On Ethernet IQ, IQ2, and IQ2-E interfaces; on MX Series router Gigabit Ethernet, Tri-Rate Ethernet copper, and 10-Gigabit Ethernet interfaces; and on aggregated Ethernet interfaces using Gigabit Ethernet IQ2 and IQ2-E or 10-Gigabit Ethernet PICs on MX Series routers, to configure the inner TPID, include the `inner-tag-protocol-id` statement at the following hierarchy levels:

   - `[edit interfaces` *`interface-name`* `unit` *`logical-unit-number`* `input-vlan-map]`
   - `[edit interfaces` *`interface-name`* `unit` *`logical-unit-number`* `output-vlan-map]`
   - `[edit logical-systems` *`logical-system-name`* `interfaces` *`interface-name`* `unit` *`logical-unit-number`* `input-vlan-map]`
   - `[edit logical-systems` *`logical-system-name`* `interfaces` *`interface-name`* `unit` *`logical-unit-number`* `output-vlan-map]`

   ```
   user@host# set inner-tag-protocol-id tpid;
   ```

2. For the inner VLAN ID, include the `inner-vlan-id` statement. For the outer TPID, include the `tag-protocol-id` statement. For the outer VLAN ID, include the `vlan-id` statement at the `[edit interfaces`

interface-name unit *logical-unit-number*] hierarchy level or at the [edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number*] hierarchy level.

```
input-vlan-map {
    (pop | pop-pop | pop-swap | push | push-push | swap | swap-push | swap-swap);
    inner-tag-protocol-id tpid;
    inner-vlan-id number;
    tag-protocol-id tpid;
    vlan-id number;
}
output-vlan-map {
    (pop | pop-pop | pop-swap | push | push-push | swap | swap-push | swap-swap);
    inner-tag-protocol-id tpid;
    inner-vlan-id number;
    tag-protocol-id tpid;
    vlan-id number;
}
```

3. For aggregated Ethernet interfaces using Gigabit Ethernet IQ interfaces, include the tag-protocol-id statement for the outer TPID. For the outer VLAN ID, include the vlan-id statement at the [edit interfaces *interface-name* unit *logical-unit-number*] hierarchy level or at the [edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number*] hierarchy level.

```
input-vlan-map {
    (pop | push | swap);
    tag-protocol-id tpid;
    vlan-id number;
}
output-vlan-map {
    (pop | push | swap);
    tag-protocol-id tpid;
    vlan-id number;
}
```

The VLAN IDs you define in the input VLAN maps are stacked on top of the VLAN ID bound to the logical interface. For more information about binding a VLAN ID to the logical interface, see "802.1Q VLANs Overview" on page 274.

All TPIDs you include in input and output VLAN maps must be among those you specify at the [edit interfaces *interface-name* gigether-options ethernet-switch-profile tag-protocol-id [*tpids*]] hierarchy level or [edit interfaces *interface-name* aggregated-ether-options ethernet-switch-profile tag-protocol-id [*tpids*]]

hierarchy level. For more information, see "Configuring Frames with Particular TPIDs to Be Processed as Tagged Frames" on page 389.

Table 68 on page 396 and Table 69 on page 397 specify when these statements are required. Table 68 on page 396 indicates valid statement combinations for rewrite operations for the input VLAN map. “No” means the statement must not be included in the input VLAN map for the rewrite operation. “Optional” means the statement may be optionally specified for the rewrite operation in the input VLAN map. “Any” means that you must include the `vlan-id` statement, `tag-protocol-id` statement, `inner-vlan-id` statement, or `inner-tag-protocol-id` statement.

Table 68: Rewrite Operations and Statement Usage for Input VLAN Maps

	Input VLAN Map Statements			
Rewrite Operation	**vlan-id**	**tag-protocol-id**	**inner-vlan-id**	**inner-tag-protocol-id**
push	Optional	Optional	No	No
pop	No	No	No	No
swap	Any	Any	No	No
push-push	Optional	Optional	Optional	optional
swap-push	Optional	Optional	Any	Any
swap-swap	Optional	Optional	Any	Any
pop-swap	No	No	Any	Any
pop-pop	No	No	No	No

Table 69 on page 397 indicates valid statement combinations for rewrite operations for the output VLAN map. “No” means the statement must not be included in the output VLAN map for the rewrite operation. “Optional” means the statement may be optionally specified for the rewrite operation in the output VLAN map.

Table 69: Rewrite Operations and Statement Usage for Output VLAN Maps

	Output VLAN Map Statements			
Rewrite Operation	**vlan-id**	**tag-protocol-id**	**inner-vlan-id**	**inner-tag-protocol-id**
push	No	Optional	No	No
pop	No	No	No	No
swap	No	Optional	No	No
push-push	No	Optional	No	Optional
swap-push	No	Optional	No	Optional
swap-swap	No	Optional	No	Optional
pop-swap	No	No	No	Optional
pop-pop	No	No	No	No

Input VLAN Map with inner-vlan-id Statement, Output VLAN Map with Optional inner-tag-protocol-id Statement

```
[edit interfaces interface-name unit logical-unit-number]
input-vlan-map {
    pop-swap;
    inner-vlan-id number;
}
output-vlan-map {
    pop-swap;
    inner-tag-protocol-id tpid;
}
```

Input VLAN Map with inner-tag-protocol-id Statement, Output VLAN Map with Optional inner-tag-protocol-id Statement

```
[edit interfaces interface-name unit logical-unit-number]
input-vlan-map {
    pop-swap;
    inner-tag-protocol-id tpid;
}
output-vlan-map {
    pop-swap;
    inner-tag-protocol-id tpid;
}
```

Input VLAN Map with inner-tag-protocol-id and inner-vlan-id Statements

```
[edit interfaces interface-name unit logical-unit-number]
input-vlan-map {
    pop-swap;
    inner-vlan-id number;
    inner-tag-protocol-id tpid;
}
```

RELATED DOCUMENTATION

- *inner-tag-protocol-id*
- *input-vlan-map*
- *output-vlan-map*
- *pop-swap*
- *unit*
- Ethernet Interfaces

Stacking a VLAN Tag

To stack a VLAN tag on all tagged frames entering or exiting the interface, include the `push`, `vlan-id`, and `tag-protocol-id` statements in the input VLAN map or the output VLAN map:

```
input-vlan-map {
    push;
    vlan-id number;
    tag-protocol-id tpid;
}
output-vlan-map {
    push;
    tag-protocol-id tpid;
}
```

You can include these statements at the following hierarchy levels:

- `[edit interfaces interface-name unit logical-unit-number]`
- `[edit interfaces interface-name unit logical-unit-number]`
- `[edit logical-systems logical-system-name interfaces interface-name unit logical-unit-number]`
- `[edit logical-systems logical-system-name interfaces interface-name unit logical-unit-number]`

All TPIDs you include in input and output VLAN maps must be among those you specify at the `[edit interfaces interface-name gigether-options ethernet-switch-profile tag-protocol-id [ tpids ]]` hierarchy level. For more information, see "Configuring Inner and Outer TPIDs and VLAN IDs" on page 394.

RELATED DOCUMENTATION

tag-protocol-id

unit

802.1Q VLANs Overview | **274**

Configuring Inner and Outer TPIDs and VLAN IDs | **394**

Ethernet Interfaces User Guide for Routing Devices

Stacking Two VLAN Tags

On Ethernet IQ, IQ2 and IQ2-E interfaces, on MX Series router Gigabit Ethernet, Tri-Rate Ethernet copper, and 10-Gigabit Ethernet interfaces, and on aggregated Ethernet interfaces using Gigabit Ethernet IQ2 and IQ2-E or 10-Gigabit Ethernet PICs on MX Series routers, to push two VLAN tags in front of tagged frames entering or exiting the interface, include the `push-push` statement in the input VLAN map or the output VLAN map:

```
push-push;
```

You can include this statement at the following hierarchy levels:

- `[edit interfaces` *`interface-name`* `unit` *`logical-unit-number`* `input-vlan-map]`
- `[edit interfaces` *`interface-name`* `unit` *`logical-unit-number`* `output-vlan-map]`
- `[edit logical-systems` *`logical-system-name`* `interfaces` *`interface-name`* `unit` *`logical-unit-number`* `input-vlan-map]`
- `[edit logical-systems` *`logical-system-name`* `interfaces` *`interface-name`* `unit` *`logical-unit-number`* `output-vlan-map]`

RELATED DOCUMENTATION

> *input-vlan-map*
>
> *output-vlan-map*
>
> *pop*
>
> *unit*

Removing a VLAN Tag

To remove a VLAN tag from all tagged frames entering or exiting the interface, include the `pop` statement in the input VLAN map or output VLAN map:

```
pop;
```

You can include this statement at the following hierarchy levels:

- [edit interfaces *interface-name* unit *logical-unit-number* input-vlan-map]
- [edit interfaces *interface-name* unit *logical-unit-number* output-vlan-map]
- [edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number* input-vlan-map]
- [edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number* output-vlan-map]

RELATED DOCUMENTATION

> *input-vlan-map*
>
> *output-vlan-map*
>
> *pop*
>
> *unit*
>
> Ethernet Interfaces User Guide for Routing Devices

Removing the Outer and Inner VLAN Tags

On Ethernet IQ, IQ2 and IQ2-E interfaces, on MX Series router Gigabit Ethernet, Tri-Rate Ethernet copper, and 10-Gigabit Ethernet interfaces, and on aggregated Ethernet interfaces using Gigabit Ethernet IQ2 and IQ2-E or 10-Gigabit Ethernet PICs on MX Series routers, to remove both the outer and inner VLAN tags of the frame, include the pop-pop statement in the input VLAN map or output VLAN map:

```
pop-pop;
```

You can include this statement at the following hierarchy levels:

- [edit interfaces *interface-name* unit *logical-unit-number* input-vlan-map]
- [edit interfaces *interface-name* unit *logical-unit-number* output-vlan-map]
- [edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number* input-vlan-map]
- [edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number* output-vlan-map]

Removing the Outer VLAN Tag and Rewriting the Inner VLAN Tag

On Ethernet IQ, IQ2 and IQ2-E interfaces, on MX Series router Gigabit Ethernet, Tri-Rate Ethernet copper, and 10-Gigabit Ethernet interfaces, and on aggregated Ethernet interfaces using Gigabit Ethernet IQ2 and IQ2-E or 10-Gigabit Ethernet PICs on MX Series routers, to remove the outer VLAN tag of the frame and replace the inner VLAN tag of the frame with a user-specified VLAN tag value, include the `pop-swap` statement in the input VLAN map or output VLAN map:

```
pop-swap;
```

The inner tag becomes the outer tag in the final frame.

You can include this statement at the following hierarchy levels:

- `[edit interfaces` *`interface-name`* `unit` *`logical-unit-number`* `input-vlan-map]`
- `[edit interfaces` *`interface-name`* `unit` *`logical-unit-number`* `output-vlan-map]`
- `[edit logical-systems` *`logical-system-name`* `interfaces` *`interface-name`* `unit` *`logical-unit-number`* `input-vlan-map]`
- `[edit logical-systems` *`logical-system-name`* `interfaces` *`interface-name`* `unit` *`logical-unit-number`* `output-vlan-map]`

RELATED DOCUMENTATION

input-vlan-map

output-vlan-map

pop-swap

unit

Rewriting the VLAN Tag on Tagged Frames

To rewrite the VLAN tag on all tagged frames entering the interface to a specified VLAN ID and TPID, include the `swap`, `tag-protocol-id`, and `vlan-id` statements in the input VLAN map:

```
input-vlan-map {
    swap;
    vlan-id number;
    tag-protocol-id tpid;
}
```

To rewrite the VLAN tag on all tagged frames exiting the interface to a specified VLAN ID and TPID, include the `swap` and `tag-protocol-id` statements in the output VLAN map:

```
output-vlan-map {
    swap;
    vlan-id number;
    tag-protocol-id tpid;
}
```

You can include these statements at the following hierarchy levels:

- `[edit interfaces interface-name unit logical-unit-number input-vlan-map]`
- `[edit logical-systems logical-system-name interfaces interface-name unit logical-unit-number input-vlan-map]`

You cannot include both the `swap` statement and the `vlan-id` statement in the output VLAN map configuration. If you include the `swap` statement in the configuration, the VLAN ID in outgoing frames is rewritten to the VLAN ID bound to the logical interface. For more information about binding a VLAN ID to the logical interface, see "802.1Q VLANs Overview" on page 274.

The swap operation works on the outer tag only, whether or not you include the `stacked-vlan-tagging` statement in the configuration. For more information, see Examples: Stacking and Rewriting Gigabit Ethernet IQ VLAN Tags.

RELATED DOCUMENTATION

- Ethernet Interfaces User Guide for Routing Devices

Rewriting a VLAN Tag on Untagged Frames

IN THIS SECTION

Overview

You can rewrite VLAN tags on untagged incoming and outgoing frames with the ethernet-ccc and the ethernet-vpls encapsulations for the following routers:

- MX240, MX480, and MX960 routers with:
  - Gigabit Ethernet Enhanced DPC with SFP
  - Gigabit Ethernet Enhanced Queuing IP Services DPCs with SFP
  - 10-Gigabit Ethernet Enhanced DPCs with XFP
  - 10-Gigabit Ethernet Enhanced Queuing IP Services DPC with XFP

Consider a network where two provider edges (PE) are connected by a Layer 2 circuit. PE1 is receiving traffic on an untagged port while the corresponding port on PE2 is tagged. In the normal case, packets coming from PE1 will be dropped at PE2 because it is expecting tagged packets. However, if PE1 can push a VLAN tag on the incoming packet before sending it across to PE2, you can ensure that packets are not dropped. To make it work in both directions, PE1 must strip the VLAN tag from outgoing packets. Therefore, a push on the ingress side is always paired with a pop on the egress side.

The rewrite operations represented by the following statement options are supported under `ethernet-ccc` and `ethernet-vpls` encapsulations:

- `push`—A VLAN tag is added to the incoming untagged frame.
- `pop`—VLAN tag is removed from the outgoing frame.

- push-push—An outer and inner VLAN tag are added to the incoming untagged frame.
- pop-pop—Both the outer and inner VLAN tags of the outgoing frame are removed.

IQ2 and 10-Gigabit Ethernet PICs support all rewrite operations described above. Details on the possible combinations of usage are explained later in this section.

NOTE: The push-push and pop-pop operations are not supported on the Gigabit Ethernet IQ PIC.

For the input-vlan-map statement, only the push and push-push options are supported because it does not make sense to remove a VLAN tag from an incoming untagged frame. Similarly, only the pop and pop-pop options are supported for the output-vlan-map statement. Also, with the push and push-push options, the tag parameters have to be explicitly specified. Apart from this, the other rules for configuring the input-vlan-map and output-vlan-map statements are the same as for tagged frames. Table 70 on page 405 through Table 72 on page 406 explain the rules in more detail.

For the input-vlan-map statement, only the push and push-push options are supported because it does not make sense to remove a VLAN tag from an incoming untagged frame. Similarly, only the pop and pop-pop options are supported for the output-vlan-map statement. Also, with the push and push-push options, the vlan-id parameters (vlan-id for push and vlan-id or inner-vlan-id for push-push) have to be explicitly specified. TPID however, is optional and the default value of 0x8100 is set if not configured. Apart from this, the other rules for configuring the input-vlan-map and output-vlan-map statements are the same as for tagged frames.

Table 70: Input VLAN Map Statements Allowed for ethernet-ccc and ethernet-vpls Encapsulations

Operation	vlan-id	tag-protocol-id	inner-vlan-id	inner-tag-protocol-id
push	Yes	Optional	No	Optional
push-push	Yes	Optional	Yes	Optional

Table 71: Output VLAN Map Statements Allowed for ethernet-ccc and ethernet-vpls Encapsulations

Operation	vlan-id	tag-protocol-id	inner-vlan-id	inner-tag-protocol-id
pop	No	No	No	No
pop-pop	No	No	No	No

Table 72: Rules for Applying Rewrite Operations to VLAN Maps

	Output VLAN Map		
Input VLAN Map	**None**	**pop**	**pop-pop**
None	Yes	No	No
push	No	Yes	No
push-push	No	No	Yes

You can use the `show interface` *`interface-name`* command to display the status of a modified VLAN map for the specified interface.

Example: push and pop with Ethernet CCC Encapsulation

```
ge-3/1/0 {
    encapsulation ethernet-ccc;
    unit 0 {
        encapsulation ethernet-ccc;
        input-vlan-map {
            push;
            tag-protocol-id 0x8100;
            vlan-id 600;
        }
        output-vlan-map pop;
        family ccc;
    }
}
```

Example: push-push and pop-pop with Ethernet CCC Encapsulation

```
ge-3/1/0 {
    encapsulation ethernet-ccc;
    unit 0 {
        encapsulation ethernet-ccc;
        input-vlan-map {
            push-push;
            tag-protocol-id 0x8100;
            inner-tag-protocol-id 0x8100;
            vlan-id 600;
            inner-vlan-id 575;
        }
        output-vlan-map pop-pop;
        family ccc;
    }
}
```

Example: push and pop with Ethernet VPLS Encapsulation

```
ge-3/1/0 {
    encapsulation ethernet-vpls;
    unit 0 {
        encapsulation ethernet-vpls;
        input-vlan-map {
            push;
            tag-protocol-id 0x8100;
            vlan-id 700;
        }
        output-vlan-map pop;
        family vpls;
    }
}
```

Example: push-push and pop-pop with Ethernet VPLS Encapsulation

```
ge-3/1/0 {
    encapsulation ethernet-vpls;
    unit 0 {
        encapsulation ethernet-vpls;
        input-vlan-map {
            push-push;
            tag-protocol-id 0x8100;
            inner-tag-protocol-id 0x8100;
            vlan-id 600;
            inner-vlan-id 575;
        }
        output-vlan-map pop-pop;
        family vpls;
    }
}
```

RELATED DOCUMENTATION

- *input-vlan-map*
- *output-vlan-map*
- *pop*
- *pop-pop*
- *push*
- *push-push*
- *unit*
- Ethernet Interfaces User Guide for Routing Devices

Rewriting a VLAN Tag and Adding a New Tag

On Ethernet IQ, IQ2 and IQ2-E interfaces, on MX Series router Gigabit Ethernet, Tri-Rate Ethernet copper, and 10-Gigabit Ethernet interfaces, on aggregated Ethernet interfaces using Gigabit Ethernet IQ2 and IQ2-E or 10-Gigabit Ethernet PICs on MX Series routers, and on Gigabit Ethernet and 10-Gigabit Ethernet interfaces on EX Series switches, to replace the outer VLAN tag of the incoming frame

with a user-specified VLAN tag value, include the swap-push statement in the input VLAN map or output VLAN map:

```
swap-push
```

A user-specified outer VLAN tag is pushed in front. The outer tag becomes an inner tag in the final frame. The stacked and rewriting Gigabit-Ethernet VLAN Tags are also referred to as Q-in-Q tunneling.

You can include this statement at the following hierarchy levels:

- [edit interfaces *interface-name* unit *logical-unit-number* input-vlan-map]
- [edit interfaces *interface-name* unit *logical-unit-number* output-vlan-map]
- [edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number* input-vlan-map]
- [edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number* output-vlan-map]

RELATED DOCUMENTATION

> *input-vlan-map*
>
> *output-vlan-map*
>
> *swap-push*
>
> *unit*
>
> Ethernet Interfaces User Guide for Routing Devices

Rewriting the Inner and Outer VLAN Tags

On Ethernet IQ, IQ2 and IQ2-E interfaces, on MX Series router Gigabit Ethernet, Tri-Rate Ethernet copper, and 10-Gigabit Ethernet interfaces, and on aggregated Ethernet interfaces using Gigabit Ethernet IQ2 and IQ2-E or 10-Gigabit Ethernet PICs on MX Series routers, to replace both the inner and the outer VLAN tags of the incoming frame with a user-specified VLAN tag value, include the swap-swap statement in the input VLAN map or output VLAN map: The stacked and rewriting Gigabit-Ethernet VLAN Tags are also referred to as Q-in-Q tunneling.

```
swap-swap;
```

You can include this statement at the following hierarchy levels:

- [edit interfaces *interface-name* unit *logical-unit-number* input-vlan-map]
- [edit interfaces *interface-name* unit *logical-unit-number* output-vlan-map]
- [edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number* input-vlan-map]
- [edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number* output-vlan-map]

RELATED DOCUMENTATION

Ethernet Interfaces User Guide for Routing Devices

Examples: Stacking and Rewriting Gigabit Ethernet IQ VLAN Tags

Configure a VLAN CCC tunnel in which Ethernet frames enter the tunnel at interface ge-4/0/0 and exit the tunnel at interface ge-4/2/0.

The following examples show how to perform the following tasks:

Push a TPID and VLAN ID Pair on Ingress

```
[edit interfaces]
ge-4/0/0 {
    vlan-tagging;
    encapsulation vlan-ccc;
    gigether-options {
        ethernet-switch-profile {
            tag-protocol-id 0x9909;
        }
    }
    unit 0 {
        encapsulation vlan-ccc;
        vlan-id 512;
        input-vlan-map {
            push;
            tag-protocol-id 0x9909;
            vlan-id 520;
        }
        output-vlan-map pop;
    }
}
ge-4/2/0 {
    vlan-tagging;
    encapsulation vlan-ccc;
    unit 0 {
        encapsulation vlan-ccc;
        vlan-id 515;
        input-vlan-map {
            swap-push;
            vlan-id 520;
            inner-vlan-id 512;
        }
        output-vlan-map {
            pop-swap;
        }
    }
    [edit protocols]
    mpls {
        interface ge-4/0/0.0;
        interface ge-4/2/0.0;
    }
```

```
    connections {
        interface-switch vlan-tag-push {
            interface ge-4/0/0.0;
            interface ge-4/2/0.0;
        }
    }
}
```

Stack Inner and Outer VLAN Tags

```
[edit interfaces]
ge-0/2/0 {
    stacked-vlan-tagging;
    mac 00.01.02.03.04.05;
    gigether-options {
        loopback;
    }
    unit 0 {
        vlan-tags outer 0x8100.200 inner 0x8100.200;
    }
}
```

Swap a VLAN ID on Ingress

```
[edit interfaces]
ge-4/0/0 {
    vlan-tagging;
    encapsulation vlan-ccc;
    gigether-options {
        ethernet-switch-profile {
            tag-protocol-id 0x9100;
        }
    }
    ...
    unit 1 {
        encapsulation vlan-ccc;
        vlan-id 1000;
        input-vlan-map {
            swap;
            tag-protocol-id 0x9100;
            vlan-id 2000;
```

```
        }
    }
}
ge-4/2/0 {
    vlan-tagging;
    encapsulation vlan-ccc;
    ...
    unit 1 {
        encapsulation vlan-ccc;
        vlan-id 2000;
        input-vlan-map {
            swap;
            tag-protocol-id 0x9100;
            vlan-id 1000;
        }
    }
    [edit protocols]
    mpls {
        ...
        interface ge-4/0/0.1;
        interface ge-4/2/0.1;
    }
    connections {
        ...
        interface-switch vlan-tag-swap {
            interface ge-4/2/0.1;
            interface ge-4/0/0.1;
        }
    }
}
```

Swap a VLAN ID on Egress

```
[edit interfaces]
ge-4/0/0 {
    vlan-tagging;
    encapsulation vlan-ccc;
    ...
    unit 1 {
        encapsulation vlan-ccc;
        vlan-id 1000;
    }
```

```
}
ge-4/2/0 {
    vlan-tagging;
    encapsulation vlan-ccc;
    gigether-options {
        ethernet-switch-profile {
            tag-protocol-id 0x8800;
        }
    }
    ...
    unit 1 {
        encapsulation vlan-ccc;
        vlan-id 2000;
        output-vlan-map {
            swap;
            tag-protocol-id 0x8800;
        }
    }
}
[edit protocols]
mpls {
    ...
    interface ge-4/0/0.1;
    interface ge-4/2/0.1;
}
connections {
    ...
    interface-switch vlan-tag-swap {
        interface ge-4/2/0.1;
        interface ge-4/0/0.1;
    }
}
```

Swap a VLAN ID on Both Ingress and Egress

```
[edit interfaces]
ge-4/0/0 {
    vlan-tagging;
    encapsulation vlan-ccc;
    gigether-options {
        ethernet-switch-profile {
            tag-protocol-id [ 0x8800 0x9100 ];
```

```
        }
    }
    ...
    unit 1 {
        encapsulation vlan-ccc;
        vlan-id 1000;
        input-vlan-map {
            swap;
            tag-protocol-id 0x9100;
            vlan-id 2000;
        }
    }
}
ge-4/2/0 {
    vlan-tagging;
    encapsulation vlan-ccc;
    gigether-options {
        ethernet-switch-profile {
            tag-protocol-id [ 0x8800 0x9100 ];
        }
    }
    unit 1 {
        encapsulation vlan-ccc;
        vlan-id 2000;
        output-vlan-map {
            swap;
            tag-protocol-id 0x8800;
        }
    }
}
[edit protocols]
mpls {
    ...
    interface ge-4/0/0.1;
    interface ge-4/2/0.1;
}
connections {
    ...
    interface-switch vlan-tag-swap {
        interface ge-4/2/0.1;
        interface ge-4/0/0.1;
```

```
    }
}
```

Swap the Outer VLAN Tag and Push a New VLAN Tag on Ingress; Pop the Outer VLAN Tag and Swap the Inner VLAN Tag on Egress

```
[edit interfaces]
ge-1/1/0 {
    unit 1 {
        vlan-id 200;
        input-vlan-map {
            swap-push;
            tag-protocol-id 0x9100;
            vlan-id 400;
            inner-tag-protocol-id 0x9100;
            inner-vlan-id 500;
        }
        output-vlan-map {
            pop-swap;
            inner-tag-protocol-id 0x9100;
        }
    }
}
```

Swap a TPID and VLAN ID Pair for Both VLAN Tags on Ingress and on Egress

```
[edit interfaces]
ge-1/1/0 {
    unit 0 {
        vlan-tags {
            inner 0x9100.425;
            outer 0x9200.525;
        }
        input-vlan-map {
            swap-swap;
            tag-protocol-id 0x9100;
            vlan-id 400;
            inner-tag-protocol-id 0x9100;
            inner-vlan-id 500;
        }
        output-vlan-map {
```

```
            swap-swap;
            tag-protocol-id 0x9200;
            inner-tag-protocol-id 0x9100;
        }
    }
}
```

Pop the Outer VLAN Tag and Swap the Inner VLAN Tag on Ingress; Swap the Outer VLAN Tag and Push a New VLAN Tag on Egress

```
[edit interfaces]
ge-1/1/0 {
    unit 0 {
        vlan-tags {
            inner 0x9100.425;
            outer 0x9200.525;
        }
        input-vlan-map {
            pop-swap;
            tag-protocol-id 0x9100;
            vlan-id 400;
        }
        output-vlan-map{
            swap-push;
            tag-protocol-id 0x9200;
            inner-tag-protocol-id 0x9100;
        }
    }
}
```

Pop a TPID and VLAN ID Pair on Ingress; Push a VLAN ID and TPID Pair on Egress

```
[edit interfaces]
ge-1/1/0 {
    unit 0 {
        vlan-tags {
            inner 0x9100.425;
            outer 0x9200.525;
        }
        input-vlan-map{
            pop-pop;
```

```
        }
        output-vlan-map {
            push-push;
            tag-protocol-id 0x9200;
            inner-tag-protocol-id 0x9100;
        }
    }
}
```

Pop an Outer VLAN Tag to Connect an Untagged VPLS Interface to Tagged VPLS Interfaces

```
[edit interfaces]
ge-1/1/0 {
    vlan-tagging;
    encapsulation extended-vlan-vpls;
    unit 0 {
        vlan-id 0;
        input-vlan-map {
            push;
            vlan-id 0;
        }
        output-vlan-map pop;
        family vpls;
    }
}
```

RELATED DOCUMENTATION

- *input-vlan-map*
- *output-vlan-map*
- *inner-tag-protocol-id*
- *inner-vlan-id*
- *pop*
- *pop-pop*
- *pop-swap*
- *push*
- *push-push*
- *swap*

swap-push

swap-swap

unit

Ethernet Interfaces User Guide for Routing Devices

Understanding Transparent Tag Operations and IEEE 802.1p Inheritance

When **swap-by-poppush** is configured on IQ2 interfaces, 10-Gigabit Ethernet LAN/WAN PIC, IQ2-E interfaces, and MX Series interfaces, during a swap operation, the inner VLAN IEEE 802.1p bits are copied to the IEEE bits of the tag being swapped. If swap-by-poppush is not configured on the interface, the VLAN IEEE 802.1p bits of the tag being swapped remains same.

When **swap-by-poppush** is configured but the incoming packet has no inner VLAN tag (transparent tag), the IEEE 802.1p bits are set to zero .

Table 73 on page 419 describes the relationship between the VLAN map operation and the inheritance of IEEE 802.1p from the transparent tag. It assumes the presence of the transparent tag in the incoming packet. If the transparent tag is not present, the IEEE 802.1p value is set to 0.

Table 73: VLAN Map Operation and IEEE 802.1p Inheritance

Rewrite Operation	Untagged *Logical Interface*	Transparent tag IEEE 802.1p Inheritance	Single-tagged Logical Interface	Transparent tag IEEE 802.1p Inheritance	Change in number of tags
push-push	yes	OUTER, INNER	NA	no operation	+2
swap-push	NA	no operation	yes	OUTER, INNER	+1
push	yes	OUTER	yes	*none	+1
swap	NA	NA	yes	OUTER	0

NOTE: *In a **push** operation on a single-tagged logical interface, none of the tags (inner, or outer) inherit the IEEE 802.1p bits from the transparent tag.

The following section shows four different examples of the inheritance of the transparent IEEE 802.1p values into the outer and inner VLAN tags.

Figure 4 on page 420 shows an incoming packet with a transparent tag. A swap-push operation swaps the outer VLAN tag and pushes another VLAN tag. The IEEE 802.1p values are inherited from the transparent tag.

Figure 4: swap-push (transparent tag)

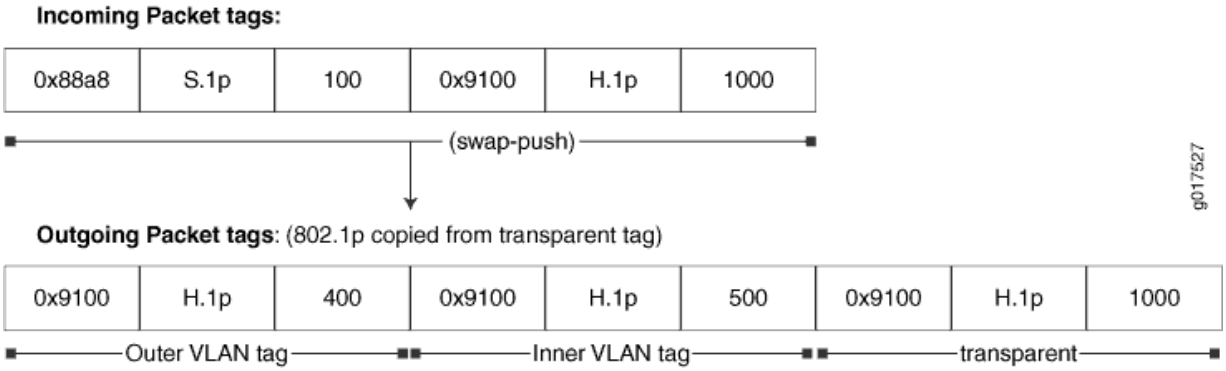


Figure 5 on page 420 shows an incoming packet with no transparent tag. A swap-push operation swaps the outer VLAN tag and pushes another VLAN tag. The IEEE 802.1p value is set to zero, as there is no transparent tag.

Figure 5: swap-push (no transparent tag)

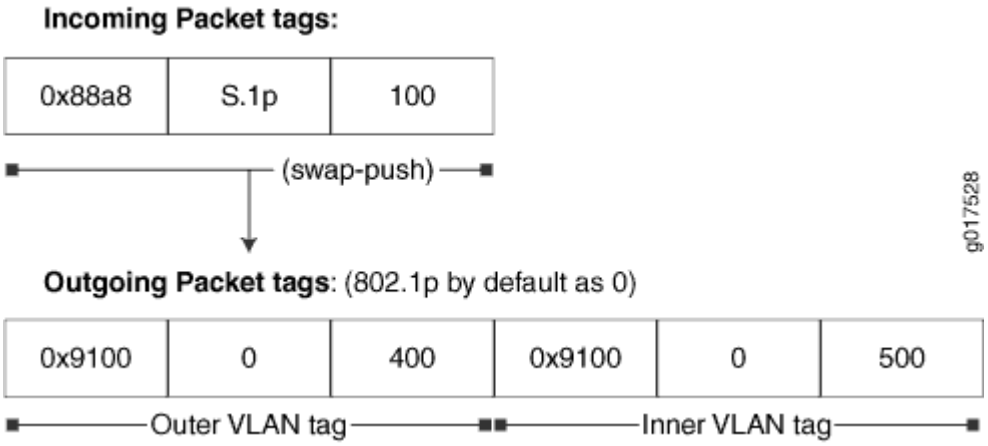


Figure 6 on page 421 shows an incoming packet with a transparent tag. A push operation pushes another VLAN tag. The IEEE 802.1p value is inherited from the transparent tag.

Figure 6: push (transparent tag)

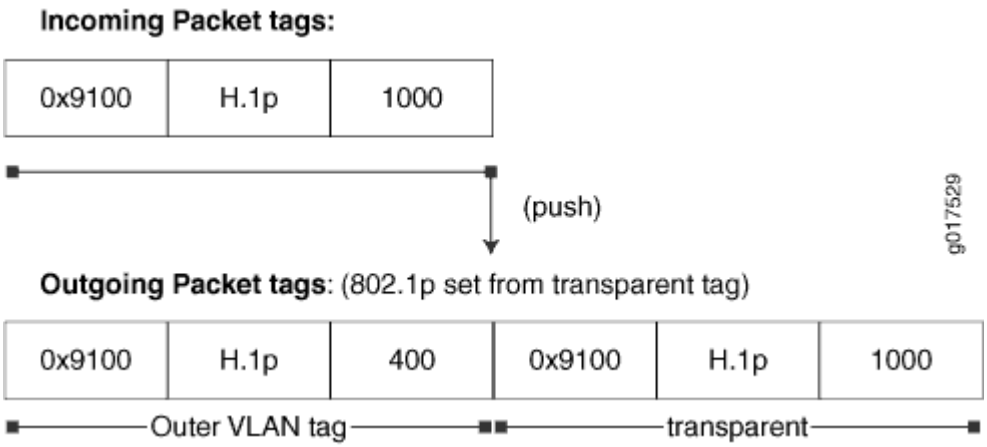
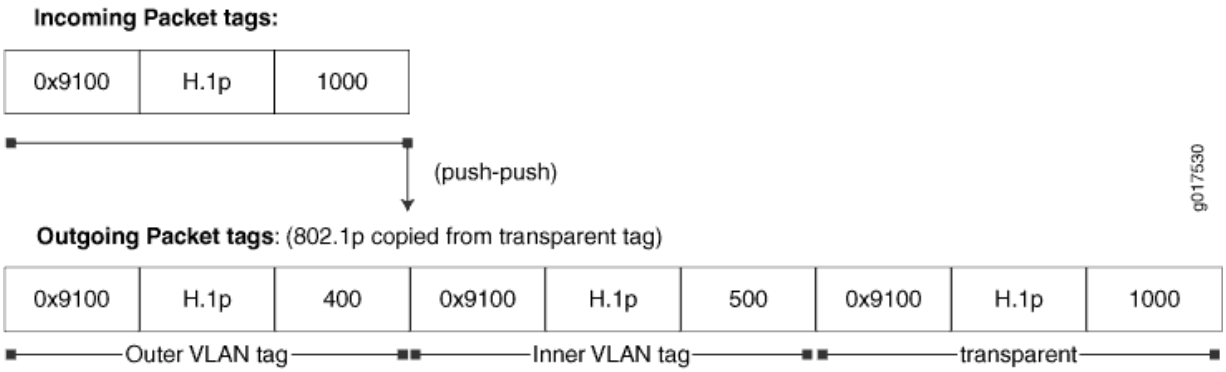


Figure 7 on page 421 shows an incoming packet with a transparent tag. A push-push operation pushes the outer and inner VLAN tags, respectively. The IEEE 802.1p values are inherited from the transparent tag.

Figure 7: push-push (transparent tag)



RELATED DOCUMENTATION

Understanding IEEE 802.1p Inheritance push and swap from a Transparent Tag

Configuring IEEE 802.1p Inheritance push and swap from the Transparent Tag | **422**

Understanding swap-by-poppush | **422**

swap-by-poppush

transparent

Understanding swap-by-poppush

By default, during a swap operation, the IEEE 802.1p bits of the VLAN tag remain unchanged. When the **swap-by-poppush** operation is enabled on a *logical interface*, the swap operation is treated as a **pop** operation followed by **push** operation. The **pop** operation removes the existing tag and the associated IEEE 802.1p bits and the push operation copies the inner VLAN IEEE 802.1p bits to the IEEE bits of the VLAN or VLANs being pushed. As a result, the IEEE 802.1p bits are inherited from the incoming transparent tag.

In effect, **swap-by-poppush** serves as a VLAN operation property and is used along with a **swap** or **swap-push** VLAN rewrite operation, indicating the nature of the swap operation being performed.

RELATED DOCUMENTATION

swap-by-poppush

transparent

Understanding IEEE 802.1p Inheritance push and swap from a Transparent Tag

Configuring IEEE 802.1p Inheritance push and swap from the Transparent Tag | 422

Understanding Transparent Tag Operations and IEEE 802.1p Inheritance | 419

Configuring IEEE 802.1p Inheritance push and swap from the Transparent Tag

To classify incoming packets based on the IEEE 802.1p bits from the transparent tag, include the `transparent` statement at the `[edit class-of-service interfaces` *interface-name* `unit` *logical-unit-number* `classifiers ieee-802.1 vlan-tag]` hierarchy level.

Tagged Interface Example

The following example configuration specifies the classification based on the transparent VLAN tag.

```
edit
class-of-service {
    interfaces {
        ge-3/0/1 {
            unit 0 {
```

```
                classifiers {
                    ieee-802.1 default vlan-tag transparent;
                }
            }
        }
    }
}
```

To configure Junos OS to inherit the IEEE 802.1p bits from the transparent tag, include the swap-by-poppush statement at the [edit interfaces *interface-name* unit *logical-unit-number*] hierarchy level.

The following is a configuration to swap and push VLAN tags and allow inheritance of the IEEE 802.1p value from the transparent VLAN tag in incoming packets.

```
edit
    ge-3/0/0 {
    vlan-tagging;
    encapsulation vlan-ccc;
    unit 0 {
        encapsulation vlan-ccc;
        vlan-id 100;
        swap-by-poppush;
        input-vlan-map {
            swap-push;
            tag-protocol-id 0x9100;
            inner-tag-protocol-id 0x9100;
            vlan-id 500;
            inner-vlan-id 400;
        }
        output-vlan-map {
            pop-swap;
            inner-vlan-id 100;
            inner-tag-protocol-id 0x88a8;
        }
    }
}
```

The swap-by-poppush statement causes a swap operation to be done as a pop followed by a push operation. So for the outer tag, the incoming S-Tag is popped and a new tag is pushed. As a result, the S-Tag inherits the IEEE 802.1p bits from the transparent tag. The inner tag is then pushed, which results in the inner tag inheriting the IEEE 802.1p bits from the transparent tag.

Untagged Interface Example

The following is a configuration to push two VLAN tags and allow inheritance of the IEEE 802.1p value from the transparent VLAN tag in the incoming packet.

```
[edit]
ge-3/0/1 {
    encapsulation ccc;
    unit 0 {
        input-vlan-map {
            push-push;
            tag-protocol-id 0x9100;
            inner-tag-protocol-id 0x9100;
            vlan-id 500;
            inner-vlan-id 400;
        }
        output-vlan-map{
            pop-pop;
        }
    }
}
```

No additional configuration is required to inherit the IEEE 802.1p value, as the `push` operation inherits the IEEE 802.1p values by default.

The following configuration specifies the classification based on the transparent VLAN tag.

```
[edit]
class-of-service {
    interfaces {
        ge-3/0/1 {
            unit 0 {
                classifiers {
                    ieee-802.1 default vlan-tag transparent;
                }
            }
        }
    }
}
```

RELATED DOCUMENTATION

transparent

swap-by-poppush

Understanding IEEE 802.1p Inheritance push and swap from a Transparent Tag

Understanding swap-by-poppush

Understanding Transparent Tag Operations and IEEE 802.1p Inheritance

17

CHAPTER

Configuring Private VLANs

IN THIS CHAPTER

Private VLANs

IN THIS SECTION

Understanding Private VLANs

IN THIS SECTION

VLANs limit broadcasts to specified users. Private VLANs (PVLANs) take this concept a step further by limiting communication within a VLAN. PVLANs accomplish this by restricting traffic flows through their member switch ports (which are called *private ports*) so that these ports communicate only with a specified uplink trunk port or with specified ports within the same VLAN. The uplink trunk port or link aggregation group (LAG) is usually connected to a router, firewall, server, or provider network. Each PVLAN typically contains many private ports that communicate only with a single uplink port, thereby preventing the ports from communicating with each other.

PVLANs provide Layer 2 isolation between ports within a VLAN, splitting a broadcast domain into multiple discrete broadcast subdomains by creating secondary VLANs (*community* VLANs and an *isolated* VLAN) inside a primary VLAN. Ports within the same community VLAN can communicate with each other. Ports within an isolated VLAN can communicate *only* with a single uplink port.

Just like regular VLANs, PVLANs are isolated on Layer 2 and require one of the following options to route Layer 3 traffic among the secondary VLANs:

- A promiscuous port connection with a router
- A routed VLAN interface (RVI)

NOTE: To route Layer 3 traffic among secondary VLANs, a PVLAN needs only one of the options mentioned above. If you use an RVI, you can still implement a promiscuous port connection to a router with the promiscuous port set up to handle only traffic that enters and exits the PVLAN.

PVLANs are useful for restricting the flow of broadcast and unknown unicast traffic and for limiting the communication between known hosts. Service providers use PVLANs to keep their customers isolated from each other. Another typical use for a PVLAN is to provide per-room Internet access in a hotel.

NOTE: You can configure a PVLAN to span switches that support PVLANs.

This topic explains the following concepts regarding PVLANs on EX Series switches:

Benefits of PVLANs

The need to segregate a single VLAN is particularly useful in the following deployment scenarios:

- Server farms—A typical Internet service provider uses a server farm to provide Web hosting for numerous customers. Locating the various servers within a single server farm provides ease of management. Security concerns arise if all servers are in the same VLAN because Layer 2 broadcasts go to all servers in the VLAN.
- Metropolitan Ethernet networks—A metro service provider offers Layer 2 Ethernet access to assorted homes, rental communities, and businesses. The traditional solution of deploying one VLAN per customer is not scalable and is difficult to manage, leading to potential waste of IP addresses. PVLANs provide a more secure and more efficient solution.

Typical Structure and Primary Application of PVLANs

A PVLAN can be configured on a single switch or can be configured to span multiple switches. The types of domains and ports are:

- Primary VLAN—The primary VLAN of the PVLAN is defined with an 802.1Q tag (VLAN ID) for the complete PVLAN. The primary PVLAN can contain multiple secondary VLANs (one isolated VLAN and multiple community VLANs).
- Isolated VLAN/isolated port—A primary VLAN can contain only one isolated VLAN. An interface within an isolated VLAN can forward packets only to a promiscuous port or the Inter-Switch Link (ISL) port. An isolated interface cannot forward packets to another isolated interface; and an isolated interface cannot receive packets from another isolated interface. If a customer device needs to have access *only* to a gateway router, the device must be attached to an isolated trunk port.

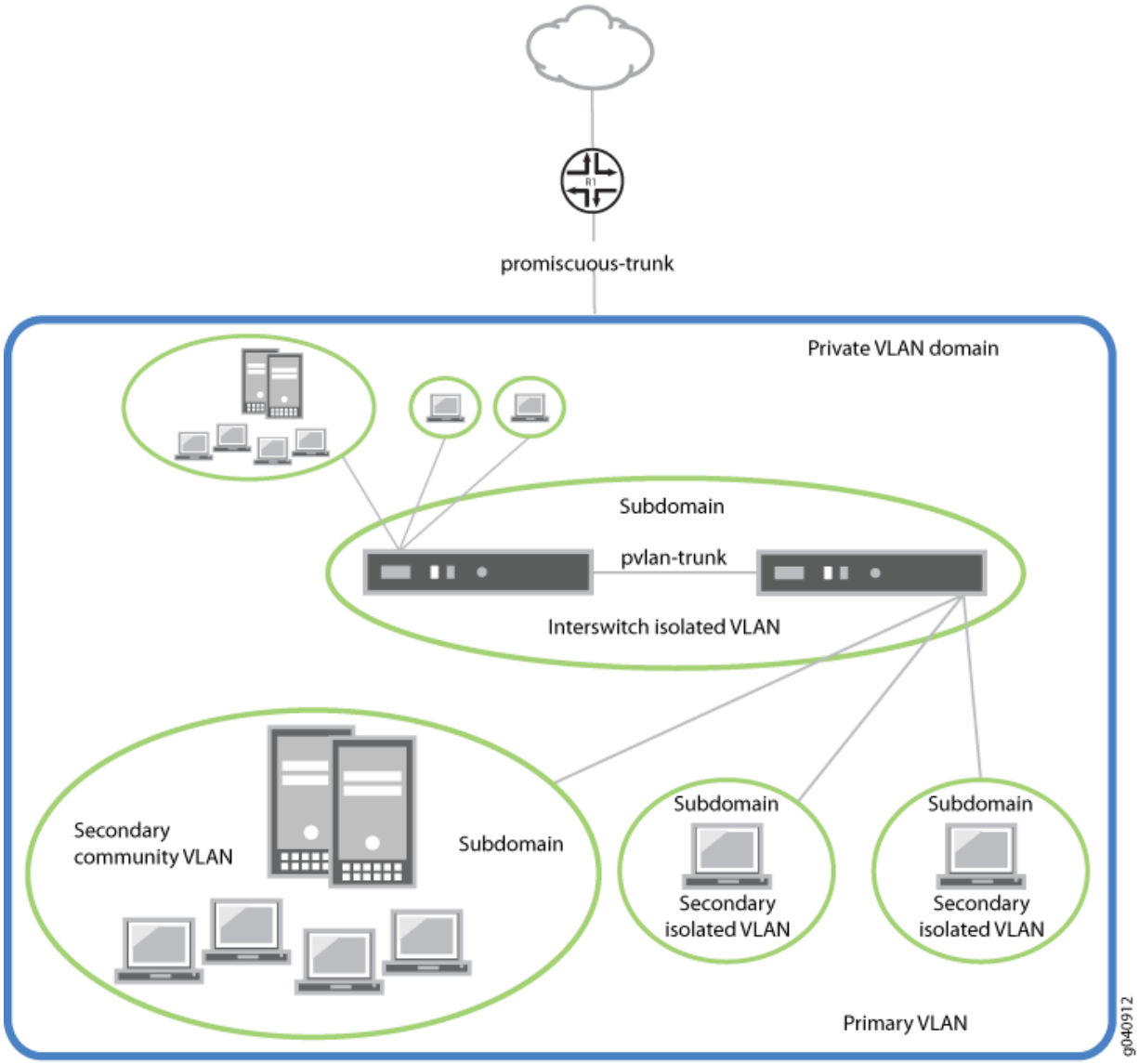
- Community VLAN/community port—You can configure multiple community VLANs within a single PVLAN. An interface within a specific community VLAN can establish Layer 2 communications with any other interface that belongs to the same community VLAN. An interface within a community VLAN can also communicate with a promiscuous port or the ISL port. If you have, for example, two customer devices that you need to isolate from other customer devices but that must be able to communicate with one another, use community ports.

- Promiscuous port—A promiscuous port has Layer 2 communications with all interfaces in the PVLAN, regardless of whether an interface belongs to an isolated VLAN or a community VLAN. A promiscuous port is a member of the primary VLAN but is not included within any secondary subdomain. Layer 3 gateways, DHCP servers, and other trusted devices that need to communicate with endpoint devices are typically connected to a promiscuous port.

- Inter-Switch Link (ISL)—An ISL is a trunk port that connects multiple switches in a PVLAN and contains two or more VLANs. It is required only when a PVLAN spans multiple switches.

The configured PVLAN is the *primary* domain (primary VLAN). Within the PVLAN, you configure *secondary* VLANs, which become subdomains nested within the primary domain. A PVLAN can be configured on a single switch or can be configured to span multiple switches. The PVLAN shown in Figure 8 on page 431 includes two switches, with a primary PVLAN domain and various subdomains.

Figure 8: Subdomains in a PVLAN



As shown in Figure 10 on page 434, a PVLAN has only one primary domain and multiple secondary domains. The types of domains are:

- Primary VLAN—VLAN used to forward frames downstream to isolated and community VLANs. The primary VLAN of the PVLAN is defined with an 802.1Q tag (VLAN ID) for the complete PVLAN. The primary PVLAN can contain multiple secondary VLANs (one isolated VLAN and multiple community VLANs).
- Secondary isolated VLAN—VLAN that receives packets only from the primary VLAN and forwards frames upstream to the primary VLAN. The isolated VLAN is a secondary VLAN nested within the primary VLAN. A primary VLAN can contain only one isolated VLAN. An interface within an isolated VLAN (isolated interface) can forward packets only to a promiscuous port or the PVLAN trunk port.

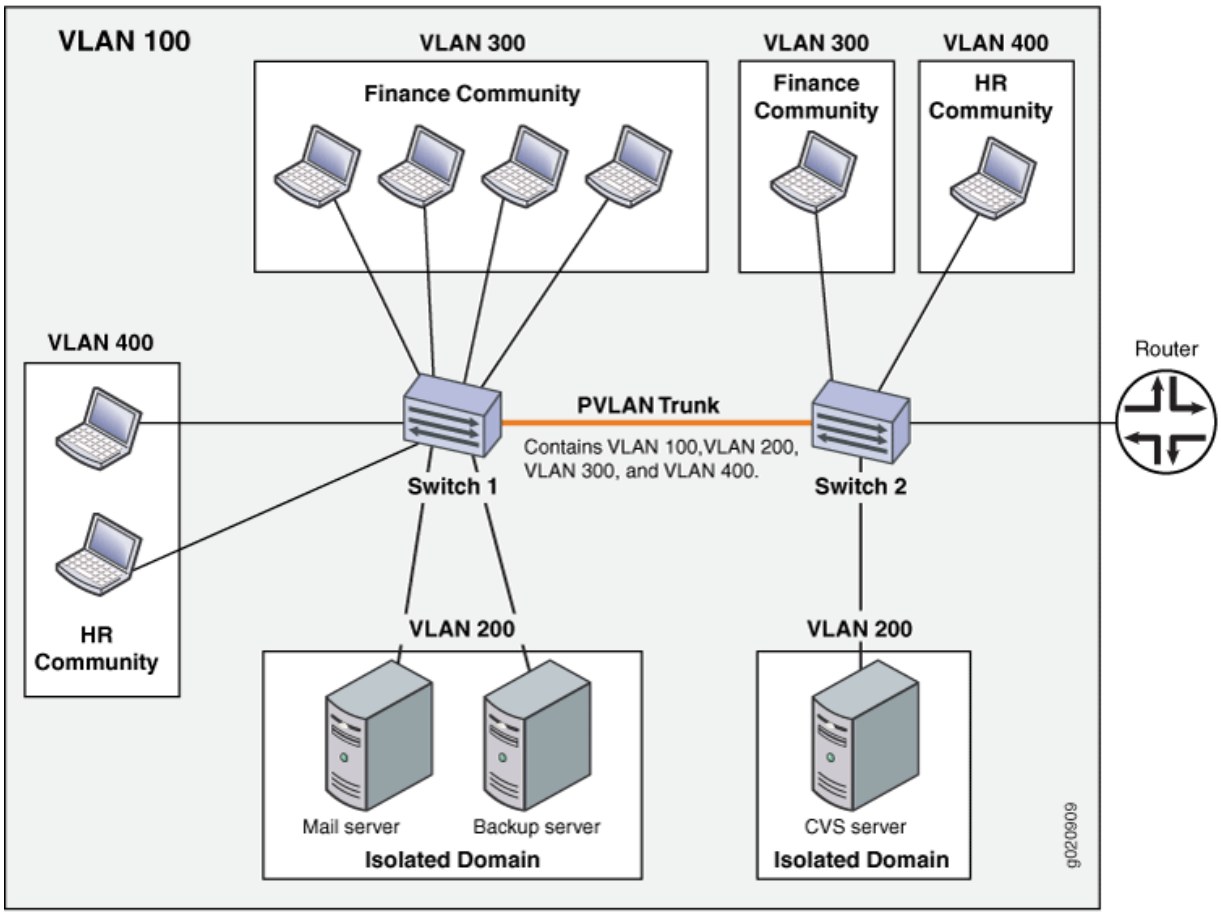
An isolated interface cannot forward packets to another isolated interface; nor can an isolated interface receive packets from another isolated interface. If a customer device needs to have access *only* to a router, the device must be attached to an isolated trunk port.

- Secondary interswitch isolated VLAN—VLAN used to forward isolated VLAN traffic from one switch to another through PVLAN trunk ports. 802.1Q tags are required for interswitch isolated VLANs because IEEE 802.1Q uses an internal tagging mechanism by which a trunking device inserts a 4-byte VLAN frame identification tab into the packet header. An interswitch isolated VLAN is a secondary VLAN nested within the primary VLAN.

- Secondary community VLAN—VLAN used to transport frames among members of a community (a subset of users within the VLAN) and to forward frames upstream to the primary VLAN. A community VLAN is a secondary VLAN nested within the primary VLAN. You can configure multiple community VLANs within a single PVLAN. An interface within a specific community VLAN can establish Layer 2 communications with any other interface that belongs to the same community VLAN. An interface within a community VLAN can also communicate with a promiscuous port or the PVLAN trunk port.

Figure 9 on page 433 shows a PVLAN spanning multiple switches, where the primary VLAN (`100`) contains two community domains (`300` and `400`) and one interswitch isolated domain.

Figure 9: PVLAN Spanning Multiple Switches

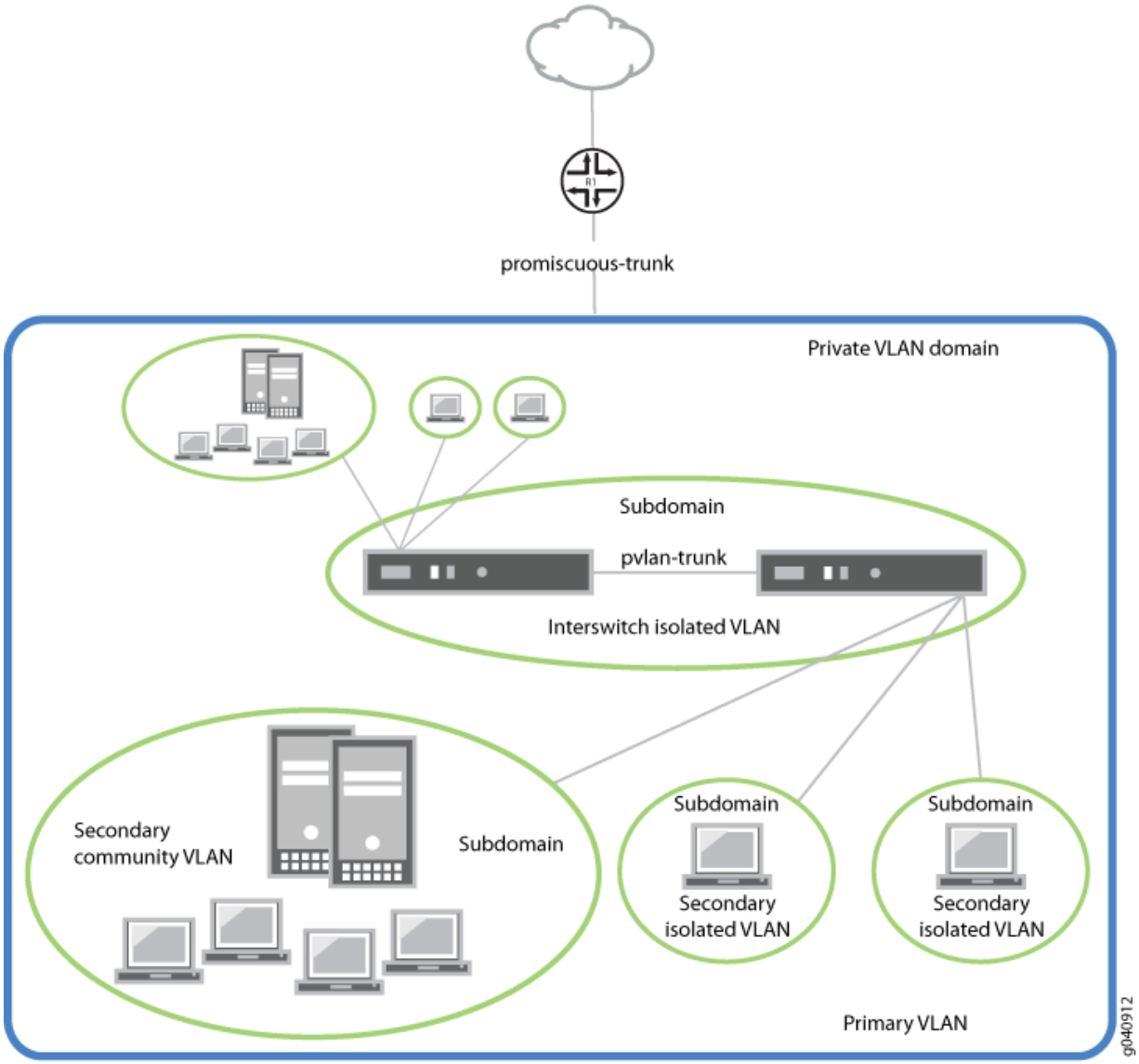


NOTE: Primary and secondary VLANs count against the limit of 4089 VLANs supported on the QFX Series. For example, each VLAN in Figure 9 on page 433 counts against this limit.

Typical Structure and Primary Application of PVLANs on MX Series Routers

The configured PVLAN becomes the primary domain, and secondary VLANs become subdomains that are nested inside the primary domain. A PVLAN can be created on a single router. The PVLAN shown in Figure 10 on page 434 includes one router, with one primary PVLAN domain and multiple secondary subdomains.

Figure 10: Subdomains in a PVLAN With One Router



The types of domains are:

- Primary VLAN—VLAN used to forward frames downstream to isolated and community VLANs.
- Secondary isolated VLAN—VLAN that receives packets only from the primary VLAN and forwards frames upstream to the primary VLAN.
- Secondary interswitch isolated VLAN—VLAN used to forward isolated VLAN traffic from one router to another through PVLAN trunk ports.
- Secondary community VLAN—VLAN used to transport frames among members of a community, which is a subset of users within the VLAN, and to forward frames upstream to the primary VLAN.

NOTE: PVLANs are supported on MX80 routers, on MX240, MX480, and MX960 routers with DPCs in enhanced LAN mode, on MX Series routers with MPC1, MPC2, and Adaptive Services PICs.

Typical Structure and Primary Application of PVLANs on EX Series Switches

NOTE: The primary VLAN of the PVLAN is defined with an 802.1Q tag (VLAN ID) for the complete PVLAN. On EX9200 switches, each secondary VLAN must also be defined with its own separate VLAN ID.

Figure 11 on page 436 shows a PVLAN on a single switch, where the primary VLAN (VLAN `100`) contains two community VLANs (VLAN `300` and VLAN `400`) and one isolated VLAN (VLAN `50`).

Figure 11: Private VLAN on a Single EX Switch

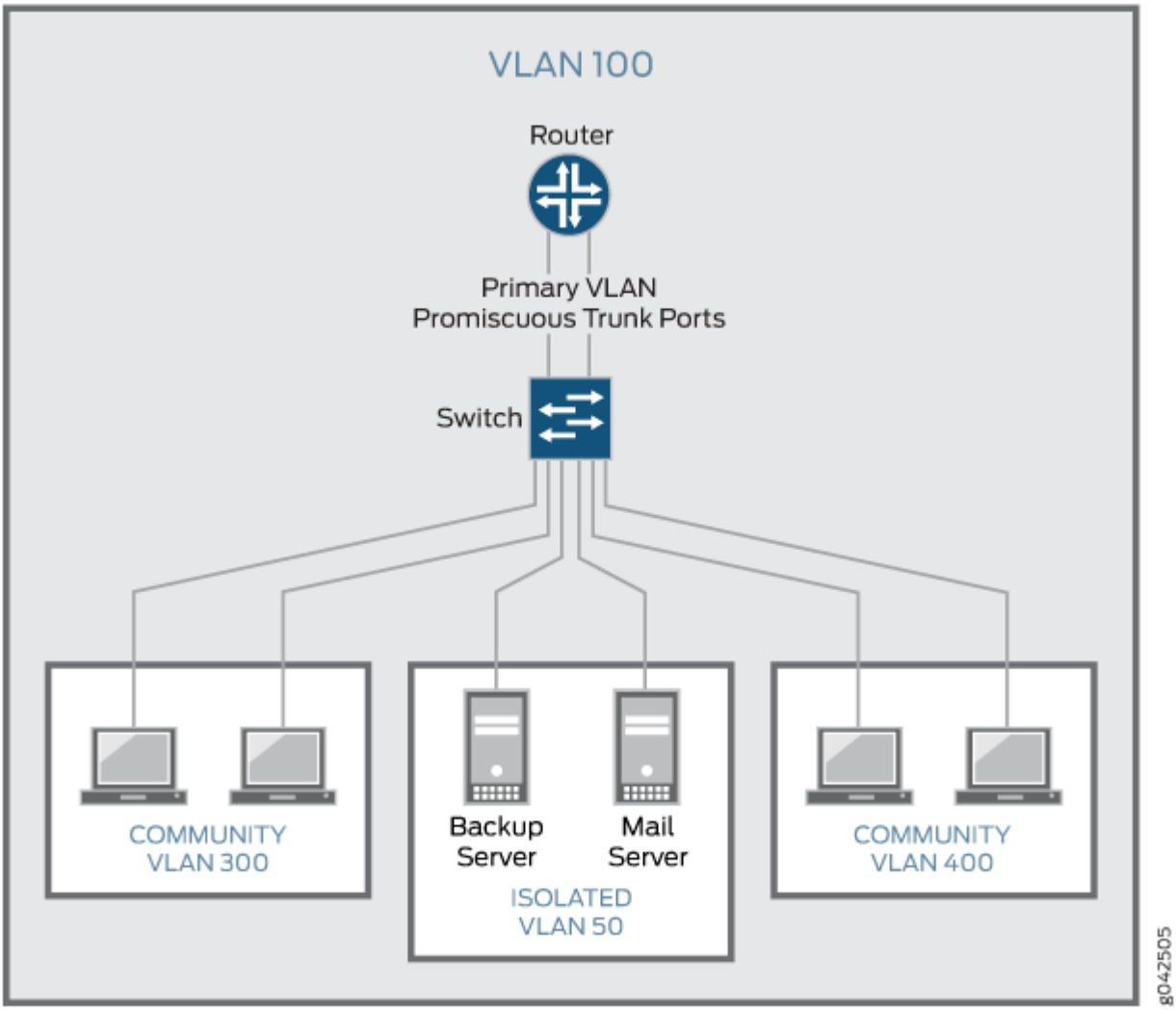
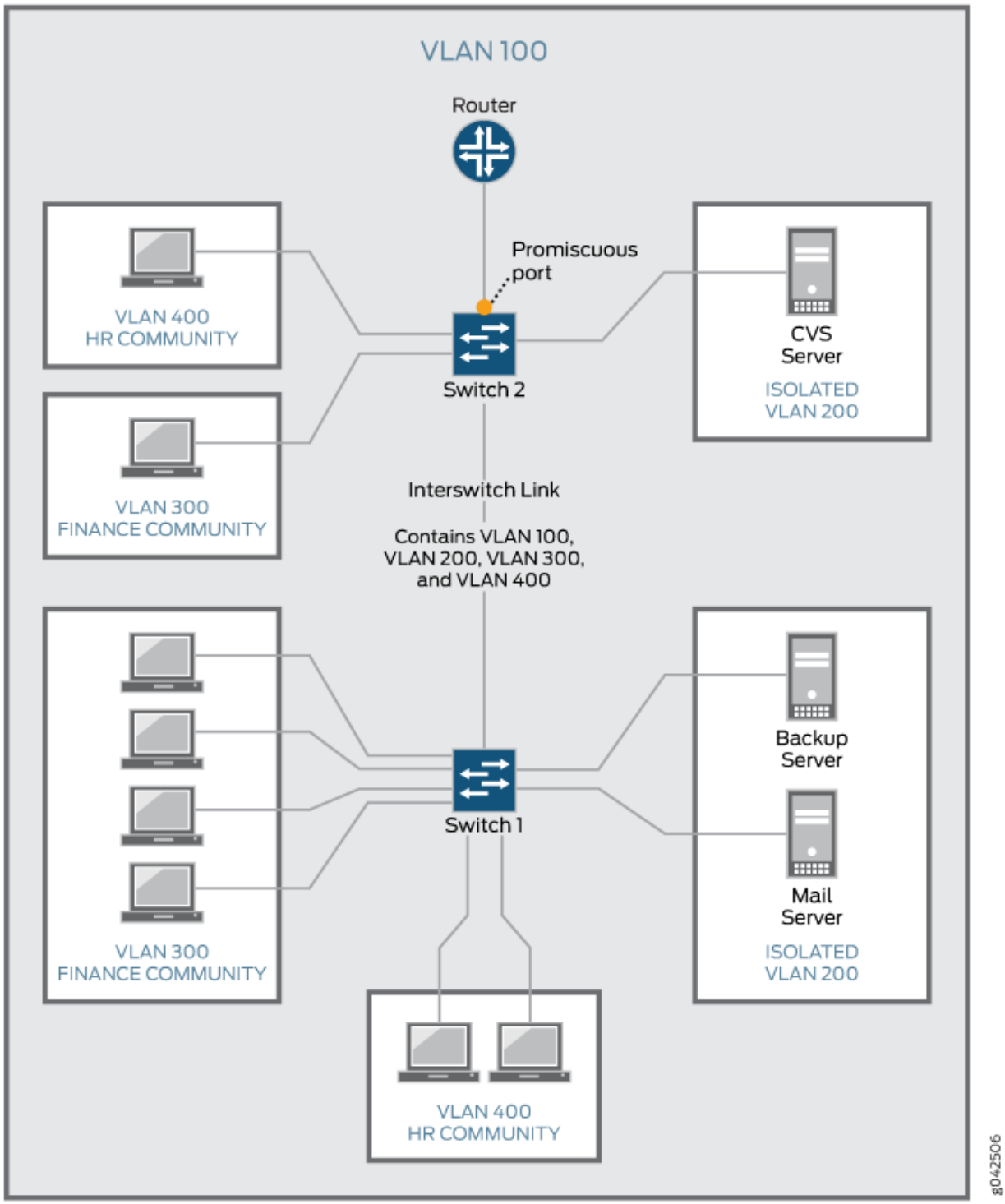


Figure 12 on page 437 shows a PVLAN spanning multiple switches, where the primary VLAN (VLAN 100) contains two community VLANs (VLAN 300 and VLAN 400) and one isolated VLAN (VLAN 200). It also shows that Switches 1 and 2 are connected through an interswitch link (PVLAN trunk link).

Figure 12: PVLAN Spanning Multiple EX Series Switches



Also, the PVLANs shown in Figure 11 on page 436 and Figure 12 on page 437 use a promiscuous port connected to a router as the means to route Layer 3 traffic among the community and isolated VLANs. Instead of using the promiscuous port connected to a router, you can configure an RVI on the switch in Figure 11 on page 436 or one of the switches shown in Figure 12 on page 437 (on some EX switches).

To route Layer 3 traffic between isolated and community VLANs, you must either connect a router to a promiscuous port, as shown in Figure 11 on page 436 and Figure 12 on page 437, or configure an RVI.

If you choose the RVI option, you must configure one RVI for the primary VLAN in the PVLAN domain. This RVI serves the entire PVLAN domain regardless of whether the domain includes one or more switches. After you configure the RVI, Layer 3 packets received by the secondary VLAN interfaces are mapped to and routed by the RVI.

When setting up the RVI, you must also enable proxy Address Resolution Protocol (ARP) so that the RVI can handle ARP requests received by the secondary VLAN interfaces.

For information about configuring PVLANs on a single switch and on multiple switches, see "Creating a Private VLAN on a Single EX Series Switch (CLI Procedure)" on page 485. For information about configuring an RVI, see "Configuring a Routed VLAN Interface in a Private VLAN on an EX Series Switch" on page 759.

Routing Between Isolated and Community VLANs

To route Layer 3 traffic between isolated and community VLANs, you must connect an external router or switch to a trunk port of the primary VLAN. The trunk port of the primary VLAN is a *promiscuous* port; therefore, it can communicate with *all* the ports in the PVLAN.

PVLANs Use 802.1Q Tags to Identify Packets

When packets are marked with a customer-specific 802.1Q tag, that tag identifies ownership of the packets for any switch or router in the network. Sometimes, 802.1Q tags are needed within PVLANs to keep track of packets from different subdomains. Table 74 on page 438 indicates when a VLAN 802.1Q tag is needed on the primary VLAN or on secondary VLANs.

Table 74: When VLANs in a PVLAN Need 802.1Q Tags

	On a Single Switch	On Multiple Switches
Primary VLAN	Specify an 802.1Q tag by setting a VLAN ID.	Specify an 802.1Q tag by setting a VLAN ID.

Table 74: When VLANs in a PVLAN Need 802.1Q Tags *(Continued)*

	On a Single Switch	On Multiple Switches
Secondary VLAN	No tag needed on VLANs.	VLANs need 802.1Q tags: • Specify an 802.1Q tag for each community VLAN by setting a VLAN ID. • Specify the 802.1Q tag for an isolation VLAN ID by setting an isolation ID.

PVLANs Use IP Addresses Efficiently

PVLANs provide IP address conservation and efficient allocation of IP addresses. In a typical network, VLANs usually correspond to a single IP subnet. In PVLANs, the hosts in all secondary VLANs belong to the same IP subnet because the subnet is allocated to the primary VLAN. Hosts within the secondary VLAN are assigned IP addresses based on IP subnets associated with the primary VLAN, and their IP subnet masking information reflects that of the primary VLAN subnet. However, each secondary VLAN is a separate broadcast domain.

PVLAN Port Types and Forwarding Rules

PVLANs can use up to six different port types. The network depicted inFigure 9 on page 433 uses a promiscuous port to transport information to the router, community ports to connect the finance and HR communities to their respective switches, isolated ports to connect the servers, and a PVLAN trunk port to connect the two switches. PVLAN ports have different restrictions:

- Promiscuous trunk port—A promiscuous port has Layer 2 communications with all the interfaces that are in the PVLAN, regardless of whether the interface belongs to an isolated VLAN or a community VLAN. A promiscuous port is a member of the primary VLAN, but is not included within one of the secondary subdomains. Layer 3 gateways, DHCP servers, and other trusted devices that need to communicate with endpoint devices are typically connected to a promiscuous port.
- PVLAN trunk link—The PVLAN trunk link, which is also known as the interswitch link, is required only when a PVLAN is configured to span multiple switches. The PVLAN trunk link connects the multiple switches that compose the PVLAN.
- PVLAN trunk port—A PVLAN trunk port is required in multiswitch PVLAN configurations to span the switches. The PVLAN trunk port is a member of all VLANs within the PVLAN (that is, the primary VLAN, the community VLANs, and the interswitch isolated VLAN), and it carries traffic from the

primary VLAN and all secondary VLANs. It can communicate with all ports other than the isolated ports.

Communication between a PVLAN trunk port and an isolated port is usually unidirectional. A PVLAN trunk port's membership in the interswitch isolated VLAN is egress-only, meaning that an isolated port can forward packets to a PVLAN trunk port, but a PVLAN trunk port does not forward packets to an isolated port (unless the packets ingressed on a promiscuous access port and are therefore being forwarded to all the secondary VLANs in the same primary VLAN as the promiscuous port).

- Secondary VLAN trunk port (not shown)—Secondary trunk ports carry secondary VLAN traffic. For a given private VLAN, a secondary VLAN trunk port can carry traffic for only one secondary VLAN. However, a secondary VLAN trunk port can carry traffic for multiple secondary VLANs as long as each secondary VLAN is a member of a different primary VLAN. For example, a secondary VLAN trunk port can carry traffic for a community VLAN that is part of primary VLAN pvlan100 and also carry traffic for an isolated VLAN that is part of primary VLAN pvlan400.

- Community port—Community ports communicate among themselves and with their promiscuous ports. Community ports serve only a select group of users. These interfaces are separated at Layer 2 from all other interfaces in other communities or isolated ports within their PVLAN.

- Isolated access port—Isolated ports have Layer 2 connectivity only with promiscuous ports and PVLAN trunk ports—an isolated port cannot communicate with another isolated port even if these two ports are members of the same isolated VLAN (or interswitch isolated VLAN) domain. Typically, a server, such as a mail server or a backup server, is connected on an isolated port. In a hotel, each room would typically be connected on an isolated port, meaning that room-to-room communication is not possible, but each room can access the Internet on the promiscuous port.

- Promiscuous access port (not shown)—These ports carry untagged traffic. Traffic that ingresses on a promiscuous access port is forwarded to all secondary VLAN ports on the device. If traffic ingresses into the device on a VLAN-enabled port and egresses on a promiscuous access port, the traffic is untagged on egress. If tagged traffic ingresses on a promiscuous access port, the traffic is discarded.

- Interswitch link port—An interswitch link (ISL) port is a trunk port that connects two routers when a PVLAN spans those routers. The ISL port is a member of all VLANs within the PVLAN (that is, the primary VLAN, the community VLANs, and the isolated VLAN).

  Communication between an ISL port and an isolated port is unidirectional. An ISL port's membership in the interswitch isolated VLAN is egress-only, meaning that incoming traffic on the ISL port is never assigned to the isolated VLAN. An isolated port can forward packets to a PVLAN trunk port, but a PVLAN trunk port cannot forward packets to an isolated port. Table 76 on page 441 summarizes whether Layer 2 connectivity exists between the different types of ports.

Table 75 on page 441 summarizes Layer 2 connectivity between the different types of ports within a PVLAN on EX Series switches that support ELS.

Table 75: PVLAN Ports and Layer 2 Forwarding on EX Series switches that support ELS

From Port Type	To Isolated Ports?	To Promiscuous Ports?	To Community Ports?	To Inter-Switch Link Port?
Isolated	Deny	Permit	Deny	Permit
Promiscuous	Permit	Permit	Permit	Permit
Community 1	Deny	Permit	Permit	Permit

Table 76: PVLAN Ports and Layer 2 Connectivity

Port Type	Promiscuous Trunk	PVLAN Trunk	Secondary Trunk	Community	Isolated Access	Promiscuous access
Promiscuous trunk	Yes	Yes	Yes	Yes	Yes	Yes
PVLAN trunk	Yes	Yes	Yes	Yes—same community only	Yes	Yes
Secondary Trunk	Yes	Yes	No	Yes	No	Yes
Community	Yes	Yes	Yes	Yes—same community only	No	Yes
Isolated access	Yes	Yes—unidirectional only	No	No	No	Yes
Promiscuous access	Yes	Yes	Yes	Yes	Yes	No

Table 77 on page 442 summarizes whether or not Layer 2 connectivity exists between the different types of ports within a PVLAN.

Table 77: PVLAN Ports and Layer 2 Connectivity on EX Series Switches without ELS Support

Port Type To: → From:↓	Promiscuous	Community	Isolated	PVLAN Trunk	RVI
Promiscuous	Yes	Yes	Yes	Yes	Yes
Community	Yes	Yes—same community only	No	Yes	Yes
Isolated	Yes	No	No	Yes **NOTE**: This communication is unidirectional.	Yes
PVLAN trunk	Yes	Yes—same community only	Yes **NOTE**: This communication is unidirectional.	Yes	Yes
RVI	Yes	Yes	Yes	Yes	Yes

As noted in Table 77 on page 442, Layer 2 communication between an isolated port and a PVLAN trunk port is unidirectional. That is, an isolated port can only send packets to a PVLAN trunk port, and a PVLAN trunk port can only receive packets from an isolated port. Conversely, a PVLAN trunk port cannot send packets to an isolated port, and an isolated port cannot receive packets from a PVLAN trunk port.

NOTE: If you enable `no-mac-learning` on a primary VLAN, all isolated VLANs (or the interswitch isolated VLAN) in the PVLAN inherit that setting. However, if you want to disable MAC address learning on any community VLANs, you must configure `no-mac-learning` on each of those VLANs.

Creating a PVLAN

The flowchart shown in Figure 13 on page 443 gives you a general idea of the process for creating PVLANs. If you complete your configuration steps in the order shown, you will not violate these PVLAN

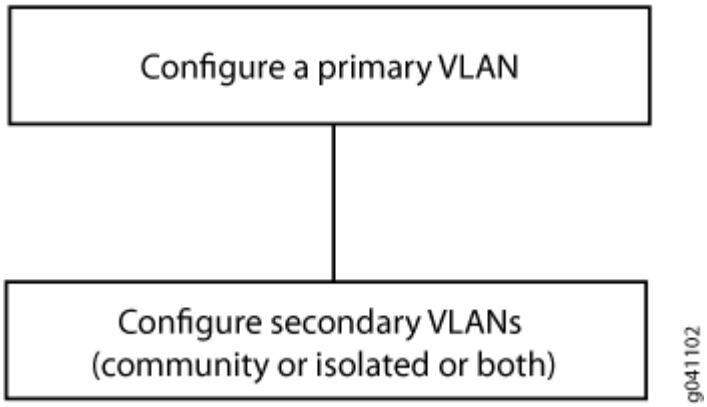
rules. (In the PVLAN rules, configuring the PVLAN trunk port applies only to a PVLAN that spans multiple routers.)

- The primary VLAN must be a tagged VLAN.
- If you are going to configure a community VLAN ID, you must first configure the primary VLAN.
- If you are going to configure an isolation VLAN ID, you must first configure the primary VLAN.

NOTE: Configuring a voice over IP (VoIP) VLAN on PVLAN interfaces is not supported.

Configuring a VLAN on a single router is relatively simple, as shown in Figure 13 on page 443.

Figure 13: Configuring a PVLAN on a Single Switch



Configuring a primary VLAN consists of these steps:

1. Configure the primary VLAN name and 802.1Q tag.
2. Set **no-local-switching** on the primary VLAN.
3. Configure the promiscuous trunk port and access ports.
4. Make the promiscuous trunk and access ports members of the primary VLAN.

Within a primary VLAN, you can configure secondary community VLANs or secondary isolated VLANs or both. Configuring a secondary community VLAN consists of these steps:

1. Configure a VLAN using the usual process.
2. Configure access interfaces for the VLAN.
3. Assign a primary VLAN to the community VLAN,

Isolated VLANs are created internally when the isolated VLAN has access interfaces as members and the option **no-local-switching** is enabled on the primary VLAN.

802.1Q tags are required for interswitch isolated VLANs because IEEE 802.1Q uses an internal tagging mechanism by which a trunking device inserts a 4-byte VLAN frame identification tab into the packet header.

Trunk ports are only needed for multirouter PVLAN configurations—the trunk port carries traffic from the primary VLAN and all secondary VLANs.

Limitations of Private VLANs

The following constraints apply to private VLAN configurations:

- An access interface can belong to only one PVLAN domain, that is, it cannot participate in two different primary VLANs.
- A trunk interface can be a member of two secondary VLANs as long as the secondary VLANs are in two *different* primary VLANs. A trunk interface cannot be a member of two secondary VLANs that are in the *same* primary VLAN.
- A single region of Multiple Spanning Tree Protocol (MSTP) must be configured on all VLANs that are included within the PVLAN.
- VLAN Spanning Tree Protocol (VSTP) is not supported.
- IGMP snooping is not supported with private VLANs.
- Routed VLAN interfaces are not supported on private VLANs
- Routing between secondary VLANs in the same primary VLAN is not supported.
- Some configuration statements cannot be specified on a secondary VLAN. You can configure the following statements at the `[edit vlans vlan-name switch-options]` hierarchy level *only* on the primary PVLAN.
- If you want to change a primary VLAN to be a secondary VLAN, you must first change it to a normal VLAN and commit the change. For example, you would follow this procedure:

  1. Change the primary VLAN to be a normal VLAN.
  2. Commit the configuration.
  3. Change the normal VLAN to be a secondary VLAN.
  4. Commit the configuration.

Follow the same sequence of commits if you want to change a secondary VLAN to be a primary VLAN. That is, make the secondary VLAN a normal VLAN and commit that change and then change the normal VLAN to be a primary VLAN.

The following features are *not* supported on PVLANs on Junos OS switches with support for the ELS configuration style:

- Egress VLAN firewall filters
- Ethernet ring protection (ERP)
- Flexible VLAN tagging
- *global-mac-statistics*
- Multichassis link aggregation groups (MC-LAGs)
- Port mirroring
- Q-in-Q tunneling
- VLAN Spanning Tree Protocol (VSTP)
- Voice over IP (VoIP)

You can configure the following statements at the `[edit vlans vlan-name switch-options]` hierarchy level only on the primary PVLAN:

- *mac-table-size*
- *no-mac-learning*
- *mac-statistics*
- *interface-mac-limit*

RELATED DOCUMENTATION

Understanding Bridging and VLANs on Switches | **140**

Understanding PVLAN Traffic Flows Across Multiple Switches

IN THIS SECTION

This topic illustrates and explains three different traffic flows on a sample multiswitch network configured with a private VLAN (PVLAN). PVLANs restrict traffic flows through their member switch ports (which are called “private ports”) so that they communicate only with a specific uplink trunk port or with specified ports within the same VLAN.

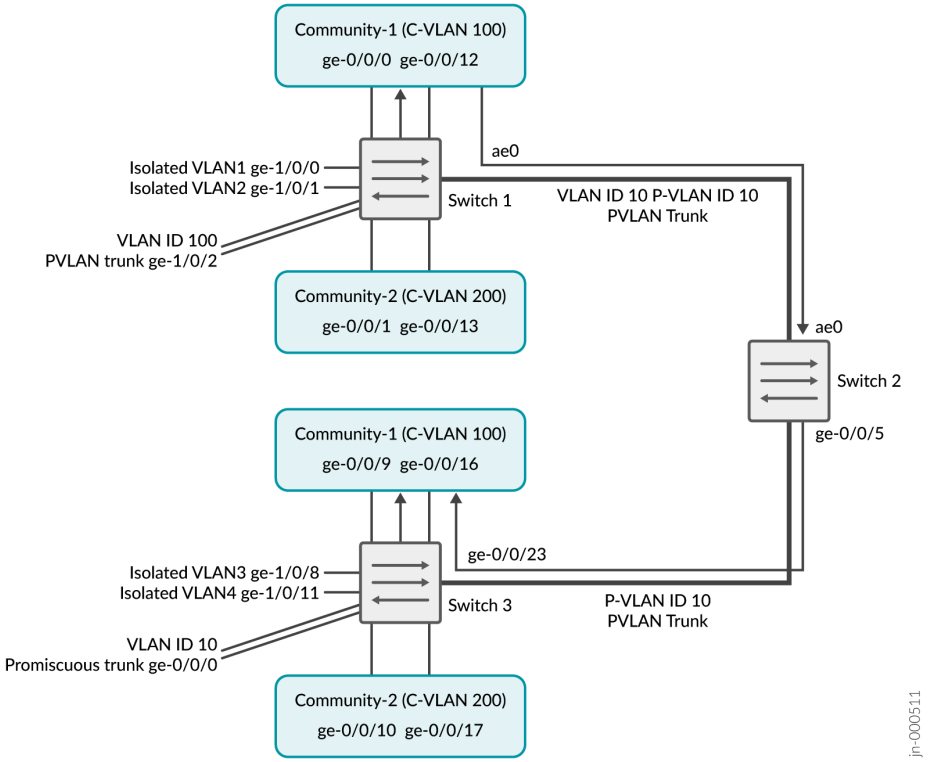
This topic describes:

Community VLAN Sending Untagged Traffic

In this example a member of Community-1 on Switch 1 sends untagged traffic on interface ge-0/0/12. The arrows in Figure 14 on page 447 represent the resulting traffic flow.

NOTE: In this example the community-1 members are assigned C-VLAN ID 100 that is mapped to P-VLAN ID 10.

Figure 14: Community VLAN Sends Untagged Traffic



In this scenario, the following activity takes place on Switch 1:

- Community-1 VLAN on interface ge-0/0/0 and ge-0/0/12: Learning
- pvlan100 on interface ge-0/0/0 and ge-0/0/12: Replication
- Community-1 VLAN on interface ge-0/0/12: Receives untagged traffic
- Community-1 VLAN interface ge-0/0/0: Traffic exits untagged
- PVLAN trunk port: Traffic exits from ge-1/0/2 and from ae0 with tag 10
- Community-2: Interfaces receive no traffic
- Isolated VLANs: Interfaces receive no traffic

In this scenario, this activity takes place on Switch 3:

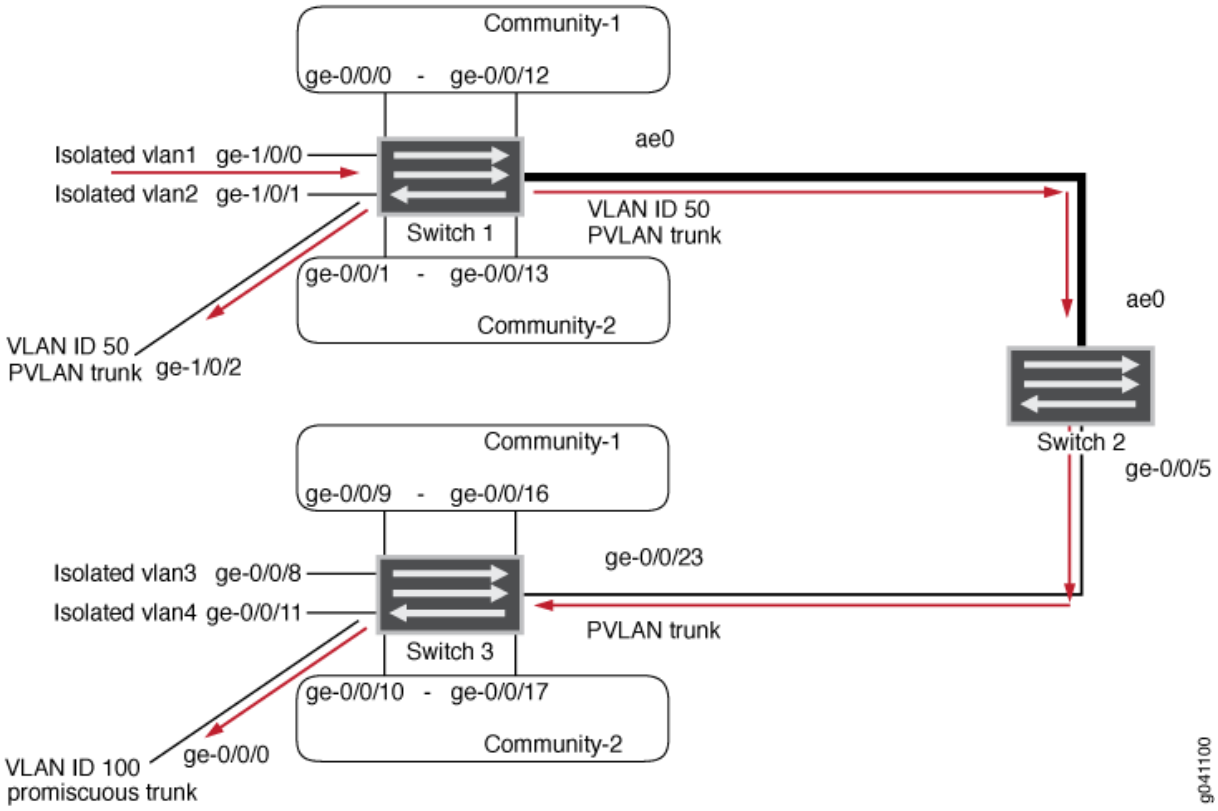
- Community-1 VLAN on interface ge-0/0/23 (PVLAN trunk): Learning
- pvlan100 on interface ge-0/0/23: Replication
- Community-1 VLAN on interfaces ge-0/0/9 and ge-0/0/16: Receive untagged traffic

- Promiscuous trunk port: Traffic exits from ge-0/0/0 with tag 10
- Community-2: Interfaces receive no traffic
- Isolated VLANs: Interfaces receive no traffic

Isolated VLAN Sending Untagged Traffic

In this scenario, isolated VLAN1 on Switch 1 at interface ge-1/0/0 sends untagged traffic. The arrows in Figure 15 on page 448 represent this traffic flow.

Figure 15: Isolated VLAN Sends Untagged Traffic



In this scenario, the following activity takes place on Switch 1:

- Isolated VLAN1 on interface ge-1/0/0: Learning
- pvlan100 on interface ge-1/0/0: Replication
- Traffic exits from pvlan-trunk ge-1/0/2 and ae0 with tag 50
- Community-1 and Community-2: Interfaces receive no traffic

- Isolated VLANs: Interfaces receive no traffic

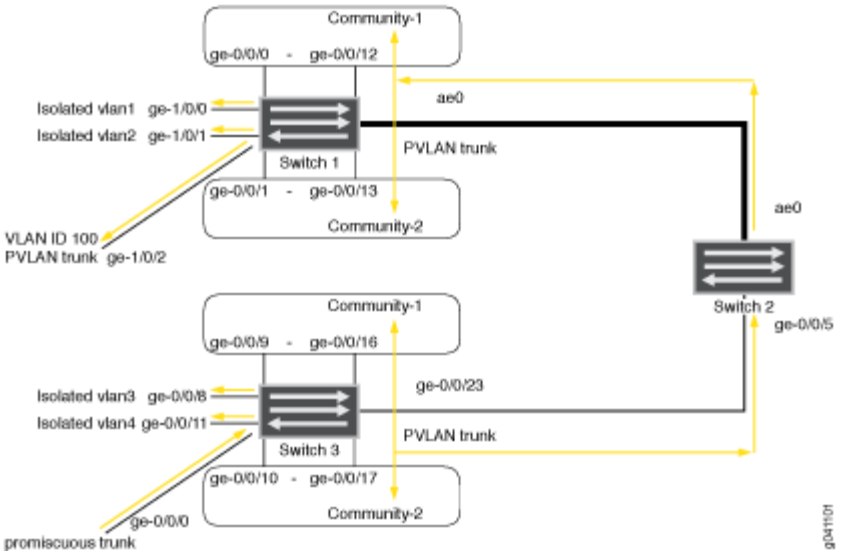
In this scenario, this activity takes place on Switch 3:

- VLAN on interface ge-0/0/23 (PVLAN trunk port): Learning
- pvlan100 on interface ge0/0/23: Replication
- Promiscuous trunk port: Traffic exits from ge-0/0/0 with tag 100
- Community-1 and Community-2: Interfaces receive no traffic
- Isolated VLANs: Receive no traffic

PVLAN Tagged Traffic Sent on a Promiscuous Port

In this scenario, PVLAN tagged traffic is sent on a promiscuous port. The arrows in Figure 16 on page 449 represent this traffic flow.

Figure 16: PVLAN Tagged Traffic Sent on a Promiscuous Port



In this scenario, the following activity takes place on Switch 1:

- pvlan100 VLAN on interface ae0 (PVLAN trunk): Learning
- Community-1, Community-2, and all isolated VLANs on interface ae0: Replication
- VLAN on interface ae0: Replication
- Traffic exits from pvlan-trunk ge-1/0/2 with tag 100

- Community-1 and Community-2: Interfaces receive traffic
- Isolated VLANs: Receive traffic

In this scenario, this activity takes place on Switch 3:

- pvlan100 on interface ge-0/0/0: Learning
- Community-1, Community-2 and all isolated VLANs on interface ge-0/0/0: Replication
- VLAN on interface ge-0/0/0: Replication
- Community-1 and Community-2: Interfaces receive traffic
- Isolated VLANs: Receive traffic

Understanding Secondary VLAN Trunk Ports and Promiscuous Access Ports on PVLANs

IN THIS SECTION

VLANs limit broadcasts to specified users. Private VLANs (PVLANs) take this concept a step further by splitting a VLAN into multiple broadcast subdomains and essentially putting secondary VLANs inside a primary VLAN. PVLANs restrict traffic flows through their member ports so that these ports communicate only with a specified uplink trunk port or with specified ports within the same VLAN. The uplink trunk port is usually connected to a router, firewall, server, or provider network. A PVLAN typically contains many private ports that communicate only with a single uplink, thereby preventing the ports from communicating with each other.

Secondary trunk ports and promiscuous access ports extend the functionality of PVLANs for use in complex deployments, such as:

- Enterprise VMWare Infrastructure environments
- Multitenant cloud services with VM management

- Web hosting services for multiple customers

For example, you can use secondary VLAN trunk ports to connect QFX devices to VMware servers that are configured with private VLANs. You can use promiscuous access ports to connect QFX devices to systems that do not support trunk ports but do need to participate in private VLANs.

This topic explains the following concepts regarding PVLANs on the QFX Series:

PVLAN Port Types

PVLANs can use the following different port types:

- Promiscuous trunk port—A promiscuous port is an upstream trunk port connected to a router, firewall, server, or provider network. A promiscuous trunk port can communicate with all interfaces, including the isolated and community ports within a PVLAN.

- PVLAN trunk port—A PVLAN trunk port is required in multiswitch PVLAN configurations to span the switches. The PVLAN trunk port is a member of all VLANs within the PVLAN (that is, the primary VLAN, the community VLANs, and the interswitch isolated VLAN), and it carries traffic from the primary VLAN and all secondary VLANs. It can communicate with all ports.

  Communication between a PVLAN trunk port and an isolated port is usually unidirectional. A PVLAN trunk port's membership in the interswitch isolated VLAN is egress-only, meaning that an isolated port can forward packets to a PVLAN trunk port, but a PVLAN trunk port does not forward packets to an isolated port (unless the packets ingressed on a promiscuous access port and are therefore being forwarded to all the secondary VLANs in the same primary VLAN as the promiscuous port).

- Secondary VLAN trunk port—Secondary VLAN trunk ports carry secondary VLAN traffic. For a given private (primary) VLAN, a secondary VLAN trunk port can carry traffic for only one secondary VLAN. However, a secondary VLAN trunk port can carry traffic for multiple secondary VLANs as long as each secondary VLAN is a member of a different primary VLAN. For example, a secondary VLAN trunk port can carry traffic for a community VLAN that is part of primary VLAN pvlan100 and also carry traffic for an isolated VLAN that is part of primary VLAN pvlan400.

  NOTE: When traffic egresses from a secondary VLAN trunk port, it normally carries the tag of the primary VLAN that the secondary port is a member of. If you want traffic that egresses from a secondary VLAN trunk port to retain its secondary VLAN tag, use the *extend-secondary-vlan-id* statement.

- Community port—Community ports communicate among themselves and with their promiscuous ports. Community ports serve only a select group of users. These interfaces are separated at Layer 2 from all other interfaces in other communities or isolated ports within their PVLAN.

- Isolated access port—Isolated ports have Layer 2 connectivity only with promiscuous ports and PVLAN trunk ports. An isolated access port cannot communicate with another isolated port even if these two ports are members of the same isolated VLAN.
- Promiscuous access port—These ports carry untagged traffic and can be a member of only one primary VLAN. Traffic that ingresses on a promiscuous access port is forwarded to the ports of the secondary VLANs that are members of the primary VLAN that the promiscuous access port is a member of. In this case, the traffic carries the appropriate secondary VLAN tag when it egresses from the secondary VLAN port if the secondary VLAN port is a trunk port. If traffic ingresses on a secondary VLAN port and egresses on a promiscuous access port, the traffic is untagged on egress. If tagged traffic ingresses on a promiscuous access port, the traffic is discarded.

Secondary VLAN Trunk Port Details

When using a secondary VLAN trunk port, be aware of the following:

- You must configure an isolation VLAN ID for each primary VLAN that the secondary VLAN trunk port will participate in. This is true even if the secondary VLANs that the secondary VLAN trunk port will carry are confined to a single device.
- If you configure a port to be a secondary VLAN trunk port for a given primary VLAN, you can also configure the same physical port to be any of the following:
  - Secondary VLAN trunk port for another primary VLAN
  - PVLAN trunk for another primary VLAN
  - Promiscuous trunk port
  - Access port for a non-private VLAN
- Traffic that ingresses on a secondary VLAN trunk port (with a secondary VLAN tag) and egresses on a PVLAN trunk port retains the secondary VLAN tag on egress.
- Traffic that ingresses on a secondary VLAN trunk port and egresses on a promiscuous trunk port has the appropriate primary VLAN tag on egress.
- Traffic that ingresses on a secondary VLAN trunk port and egresses on a promiscuous access port is untagged on egress.
- Traffic that ingresses on a promiscuous trunk port with a primary VLAN tag and egresses on a secondary VLAN trunk port carries the appropriate secondary VLAN tag on egress. For example, assume that you have configured the following on a switch:
  - Primary VLAN 100
  - Community VLAN 200 as part of the primary VLAN

- Promiscuous trunk port
- Secondary trunk port that carries community VLAN 200

If a packet ingresses on the promiscuous trunk port with primary VLAN tag 100 and egresses on the secondary VLAN trunk port, it carries tag 200 on egress.

Use Cases

IN THIS SECTION

On the same physical interface, you can configure multiple secondary VLAN trunk ports (in different primary VLANs) or combine a secondary VLAN trunk port with other types of VLAN ports. The following use cases provide examples of doing this and show how traffic would flow in each case:

Secondary VLAN Trunks In Two Primary VLANS

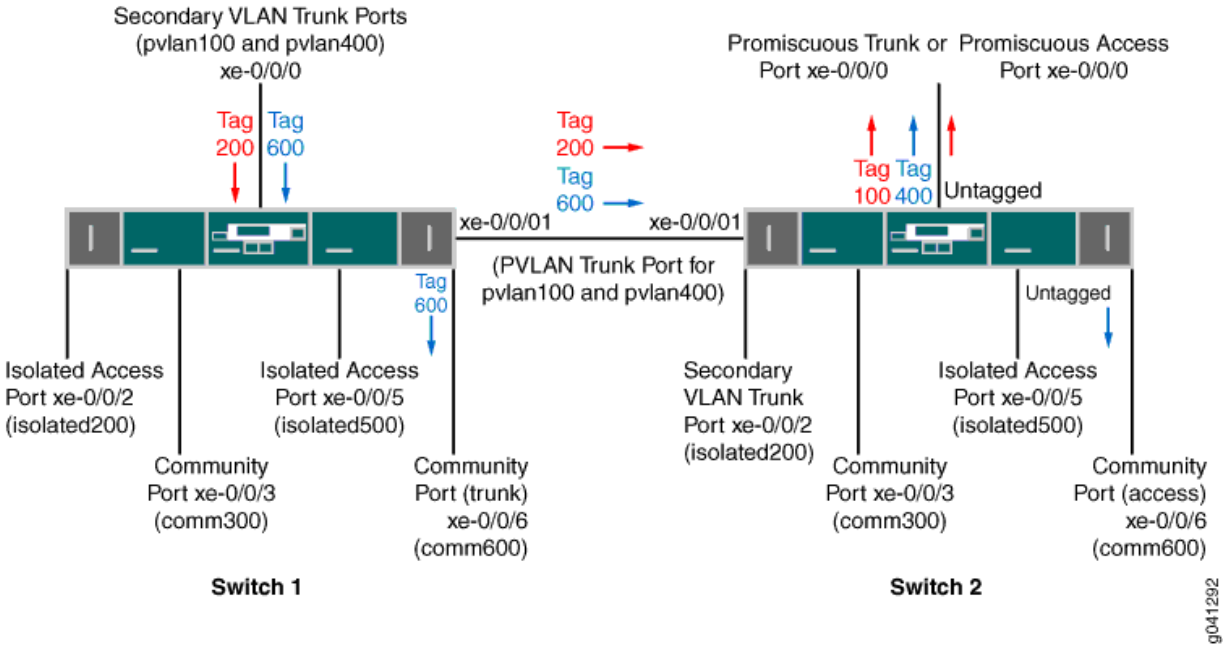
For this use case, assume you have two switches with the following configuration:

- Primary VLAN pvlan100 with tag 100.
  - Isolated VLAN isolated200 with tag 200 is a member of pvlan100.
  - Community VLAN comm300 with tag 300 is a member of pvlan100.
- Primary VLAN pvlan400 with tag 400.
  - Isolated VLAN isolated500 with tag 500 is a member of pvlan400.
  - Community VLAN comm600 with tag 600 is a member of pvlan400.
- Interface xe-0/0/0 on Switch 1 connects to a VMware server (not shown) that is configured with the private VLANs used in this example. This interface is configured with secondary VLAN trunk ports to carry traffic for secondary VLAN comm600 and the isolated VLAN (tag 200) that is a member of pvlan100.

- Interface xe-0/0/0 on Switch 2 is shown configured as a promiscuous trunk port or promiscuous access port. In the latter case, you can assume that it connects to a system (not shown) that does not support trunk ports but is configured with the private VLANs used in this example.
- On Switch 1, xe-0/0/6 is a member of comm600 and is configured as a trunk port.
- On Switch 2, xe-0/0/6 is a member of comm600 and is configured as an access port.

Figure 10 shows this topology and how traffic for isolated200 and comm600 would flow after ingressing on xe-0/0/0 on Switch 1. Note that traffic would flow only where the arrows indicate. For example, there are no arrows for interfaces xe-0/0/2, xe-0/0/3, and xe-0/0/5 on Switch 1 because no packets would egress on those interfaces.

Figure 17: Two Secondary VLAN Trunk Ports on One Interface



Here is the traffic flow for VLAN isolated200:

1. After traffic for isolated200 ingresses on the secondary VLAN trunk port on Switch 1, it egresses on the PVLAN trunk port because the PVLAN trunk port is a member of all the VLANs. The packets keep the secondary VLAN tag (200) when egressing.
2. After traffic for isolated200 ingresses on the secondary VLAN trunk port on Switch 2, it egresses on xe-0/0/0, which is configured as a promiscuous trunk port or promiscuous access port.
   - If xe-0/0/0 on Switch 2 is configured as a promiscuous trunk port, the packets egress on this port with the primary VLAN tag (100).

- If xe-0/0/0 on Switch 2 is configured as a promiscuous access port, the packets egress on this port untagged.

Note that traffic for VLAN isolated200 does not egress on isolated access port xe-0/0/2 on Switch 1 or secondary VLAN trunk port xe-0/0/2 on Switch 2 even though these two ports are members of the same isolated VLAN.

Here is the traffic flow for VLAN comm600:

1. After traffic for comm600 ingresses on the secondary VLAN trunk port on Switch 1, it egresses on the PVLAN trunk port because the PVLAN trunk port is a member of all the VLANs. The packets keep the secondary VLAN tag (600) when egressing.
2. Traffic for comm600 also egresses on community port xe-0/0/6 on Switch 1. The traffic is tagged because the port is configured as a trunk.
3. After traffic for comm600 ingresses on the PVLAN trunk port on Switch 2, it egresses on xe-0/0/0, if this interface is configured as a promiscuous trunk port.

   NOTE: If xe-0/0/0 on Switch 2 is configured as a promiscuous access port, the port can participate in only one primary VLAN. In this case, the promiscuous access port is part of pvlan100, so traffic for comm600 does not egress from it

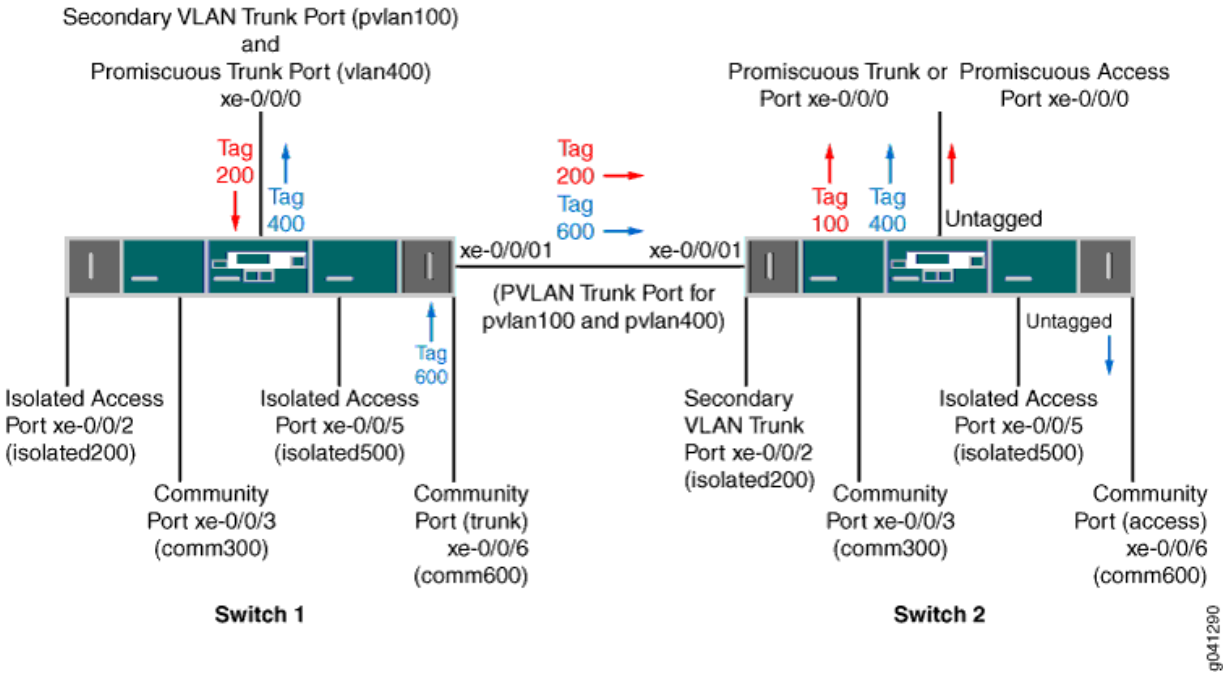
4. Traffic for comm600 also egresses on community port xe-0/0/6 on Switch 2. In this case, the traffic is untagged because the port mode is access.

Secondary VLAN Trunk and Promiscuous Trunk

For this use case, assume you have two switches configured with the same ports and VLANs as in the previous use case, with one exception: In this case, xe-0/0/0 on Switch 1 is configured as a secondary VLAN trunk port for VLAN pvlan100 and is also configured as a promiscuous trunk port for pvlan400.

Figure 11 shows this topology and how traffic for isolated200 (member of pvlan100) and comm600 (member of pvlan400) would flow after ingressing on Switch 1.

Figure 18: Secondary VLAN Trunk and Promiscuous Trunk on One Interface



The traffic flow for VLAN isolated200 is the same as in the previous use case, but the flow for comm600 is different. Here is the traffic flow for VLAN comm600:

1. After traffic for comm600 ingresses on community VLAN port xe-0/0/6 on Switch 1, it egresses on promiscuous trunk port xe-0/0/0 on Switch 1. In this case it carries the primary VLAN tag (400).
2. Traffic for comm600 also egresses on the PVLAN trunk port because the PVLAN trunk port is a member of all the VLANs. The packets keep the secondary VLAN tag (600) when egressing.
3. After traffic for comm600 ingresses on the PVLAN trunk port on Switch 2, it egresses on xe-0/0/0, if this interface is configured as a promiscuous trunk port.

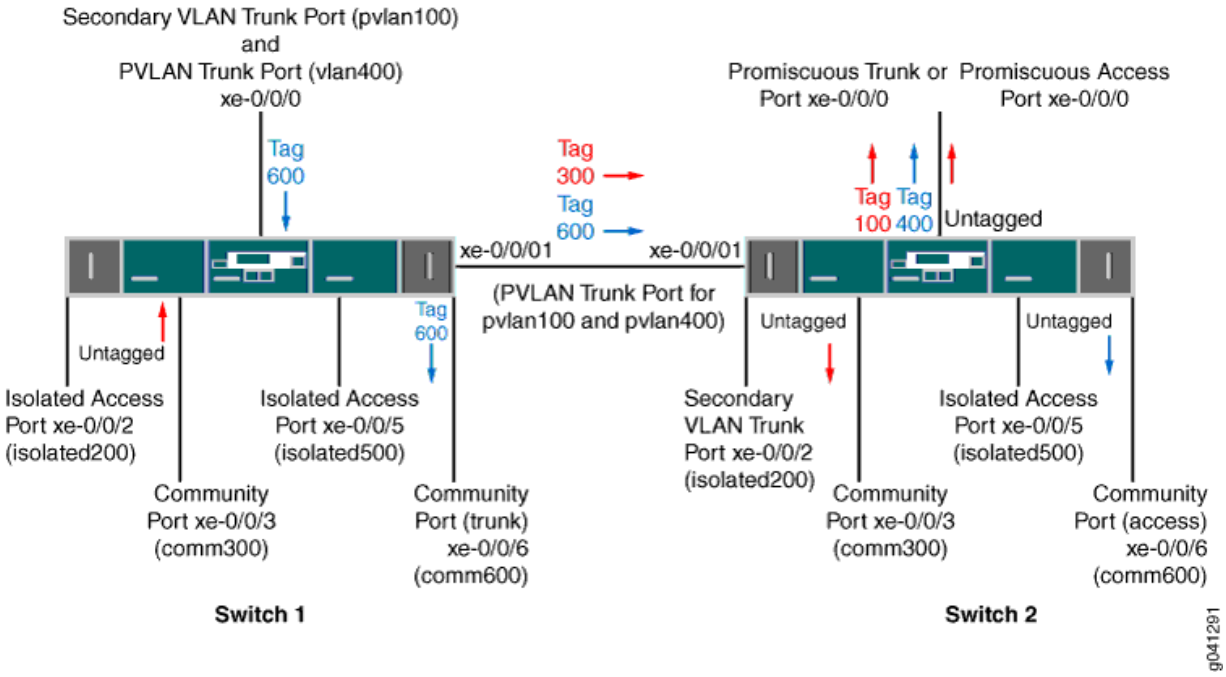
   It does not egress on xe-0/0/0 if this interface is configured as a promiscuous access port because the port can participate only in pvlan100.
4. Traffic for comm600 also egresses on community port xe-0/0/6 on Switch 2.

Secondary VLAN Trunk and PVLAN Trunk

For this use case, assume you have two switches configured with the same ports and VLANs as in the previous use cases except that xe-0/0/0 on Switch 1 is configured as a secondary VLAN trunk port for VLAN pvlan100 and is also configured as a PVLAN trunk port for pvlan400.

Figure 12 shows this topology and how traffic for comm300 (member of pvlan100) and comm600 (member of pvlan400) would flow after ingressing on Switch 1.

Figure 19: Secondary VLAN Trunk and PVLAN Trunk on One Interface



Here is the traffic flow for VLAN comm300:

1. After traffic for comm300 ingresses on community port xe-0/0/3 on Switch 1, it egresses on PVLAN trunk port xe-0/0/1 because that PVLAN trunk port is a member of all the VLANs. The packets keep the secondary VLAN tag (300) when egressing.

   NOTE: Traffic for comm300 does not egress on xe-0/0/0 because the secondary VLAN trunk port on this interface carries isolated200, not comm300.

2. After traffic for comm300 ingresses on the PVLAN trunk port on Switch 2, it egresses on xe-0/0/0, which is configured as a promiscuous trunk port or promiscuous access port.

   - If xe-0/0/0 on Switch 2 is configured as a promiscuous trunk port, the packets egress on this port with the primary VLAN tag (100).
   - If xe-0/0/0 on Switch 2 is configured as a promiscuous access port, the packets egress on this port untagged.

3. Traffic for comm300 also egresses on community port xe-0/0/3 on Switch 2.

Here is the traffic flow for VLAN comm600:

1. After traffic for comm600 ingresses on the PVLAN port xe-0/0/0 on Switch 1, it egresses on the community port xe-0/0/6 on Switch 1. The packets keep the secondary VLAN tag (600) when egressing because xe-0/0/6 is a trunk port.

2. Traffic for comm600 also egresses on PVLAN trunk port xe-0/0/1 because that PVLAN trunk port is a member of all the VLANs. The packets keep the secondary VLAN tag (600) when egressing.

3. After traffic for comm600 ingresses on the PVLAN trunk port on Switch 2, it egresses on xe-0/0/0, if this interface is configured as a promiscuous trunk port.

   It does not egress on xe-0/0/0 if this interface is configured as a promiscuous access port because the port can participate only in pvlan100.

4. Traffic for comm600 also egresses on community port xe-0/0/6 on Switch 2. This traffic is untagged on egress because xe-0/0/6 is an access port.

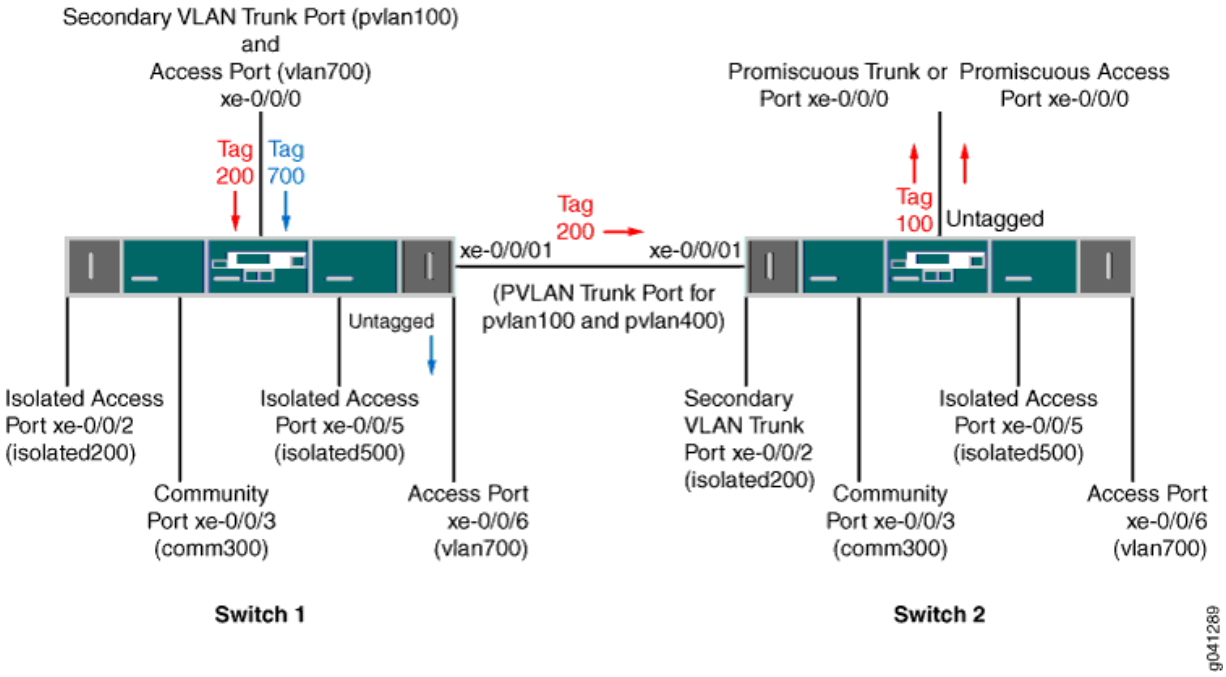
Secondary VLAN Trunk and Non-Private VLAN Interface

For this use case, assume you have two switches configured with the same ports and VLANs as in the previous use cases except for these differences:

- Configuration for xe-0/0/0 on Switch 1:
  - Secondary VLAN trunk port for VLAN pvlan100
  - Access port for vlan700
- Port xe-0/0/6 on both switches is an access port for vlan700.

Figure 13 shows this topology and how traffic for isolated200 (member of pvlan100) and vlan700 would flow after ingressing on Switch 1.

Figure 20: Secondary VLAN Trunk and Non-Private VLAN Port on One Interface



Here is the traffic flow for VLAN isolated200:

1. After traffic for isolated200 ingresses on the secondary VLAN trunk port on Switch 1, it egresses on the PVLAN trunk port. The packets keep the secondary VLAN tag (200) when egressing.
2. After traffic for isolated200 ingresses on the PVLAN trunk port on Switch 2, it egresses on xe-0/0/0, which is configured as a promiscuous trunk port or promiscuous access port.
   - If xe-0/0/0 on Switch 2 is configured as a promiscuous trunk port, the packets egress on this port with the primary VLAN tag (100).
   - If xe-0/0/0 on Switch 2 is configured as a promiscuous access port, the packets egress on this port untagged.

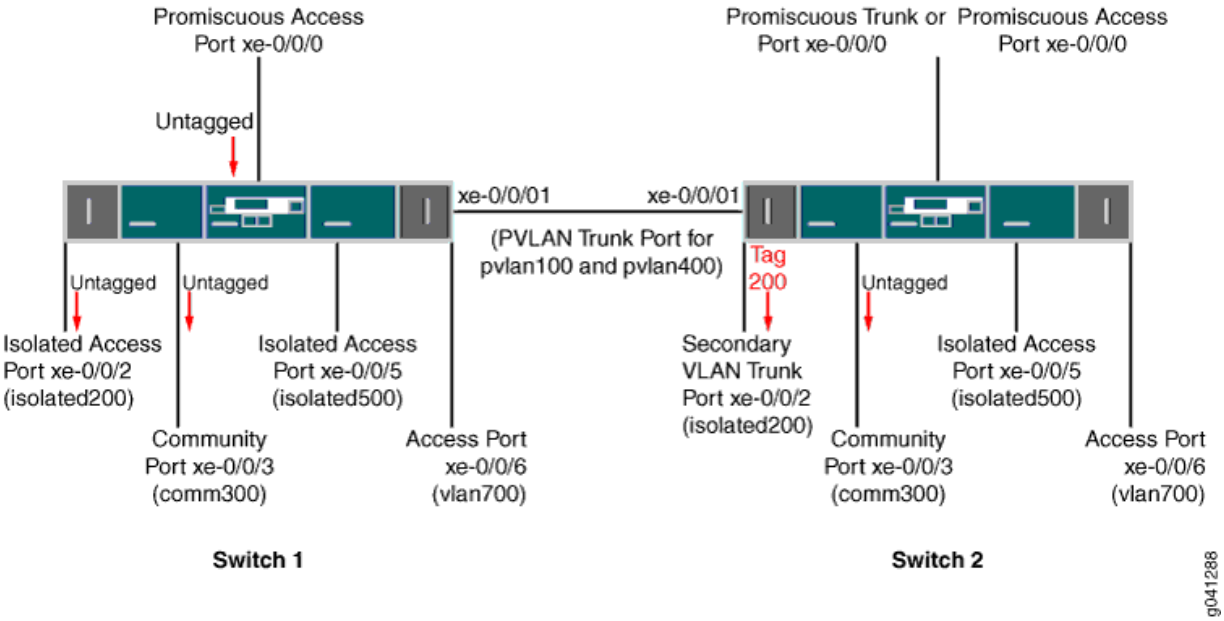
Note that traffic for VLAN isolated200 does not egress on isolated access port xe-0/0/2 on Switch 1 or secondary VLAN trunk port xe-0/0/2 on Switch 2 even though these two ports are members of the same isolated VLAN.

After traffic for vlan700 ingresses on the access port configured on xe-0/0/0 on Switch 1, it egresses on access port xe-0/0/6 because that port is a member of the same VLAN. Traffic for vlan700 is not forwarded to Switch 2 (even though xe-0/0/6 on Switch 2 is a member of vlan700) because the PVLAN trunk on xe-0/0/1 does not carry this VLAN.

Traffic Ingressing on Promiscuous Access Port

For this use case, assume you have two switches configured with the same ports and VLANs as in the previous use case except that xe-0/0/0 on Switch 1 is configured as a promiscuous access port and is a member of pvlan100. Figure 14 shows this topology and how untagged traffic would flow after ingressing through this interface on Switch 1.

Figure 21: Traffic Ingressing on Promiscuous Access Port



As the figure shows, untagged traffic that ingresses on a promiscuous access port is forwarded to all the secondary VLAN ports that are members of the same primary VLAN that the promiscuous access port is a member of. The traffic is untagged when it egresses from access ports and tagged on egress from a trunk port (xe-0/0/2 on Switch 2).

RELATED DOCUMENTATION

- *Understanding Egress Firewall Filters with PVLANs*

Using 802.1X Authentication and Private VLANs Together on the Same Interface

IN THIS SECTION

Understanding Using 802.1X Authentication and PVLANs Together on the Same Interface

You can now configure both 802.1X authentication and private VLANs (PVLANs) on the same interface.

IEEE 802.1X authentication provides network edge security, protecting Ethernet LANs from unauthorized user access by blocking all traffic to and from a supplicant (client) at the interface until the supplicant's credentials are presented and matched on the *authentication server* (a RADIUS server).

Private VLANs (PVLANs) provide Layer 2 isolation between ports within a VLAN, splitting a broadcast domain into multiple discrete broadcast subdomains by creating secondary VLANs. PVLANs are useful for restricting the flow of broadcast and unknown unicast traffic and for limiting the communication between known hosts.

On a switch that is configured with both 802.1X authentication and PVLANs, when a new device is attached to the PVLAN network, the device is authenticated and then is assigned to a secondary VLAN based on the PVLAN configuration or RADIUS profile. The device then obtains an IP address and is given access to the PVLAN network.

NOTE: This document does not provide detailed information about 802.1X authentication or private VLANs. For those details, see the feature documentation that is specific to those individual features. For 802.1X, see User Access and Authentication User Guide. For PVLANs, see Ethernet Switching User Guide.

Configuration Guidelines for Combining 802.1X Authentication with PVLANs

Keep the following guidelines and limitations in mind for configuring these two features on the same interface:

- You cannot configure an 802.1X-enabled interface as a promiscuous interface (an interface that is a member of the primary VLAN by configuration) or as an interswitch-link (ISL) interface.
- Multiple users cannot be authenticated over different VLANs belonging to the same PVLAN domain on a logical interface—for example, if interface ge-0/0/0 is configured as `supplicant multiple` and clients C1 and C2 are authenticated and are added to dynamic VLANs V1 and V2, respectively, then V1 and V2 must belong to different PVLAN domains.
- If the VoIP VLAN and the data VLAN are different, those two VLANs must be in different PVLAN domains.
- When PVLAN membership is changed (that is, an interface is reconfigured in a different PVLAN), clients must be reauthenticated.

Example: Configuring 802.1X Authentication with Private VLANs in One Configuration

IN THIS SECTION

Requirements

- Junos OS Release 18.2R1 or later
- EX2300, EX3400, or EX4300 switch

Before you begin, specify the RADIUS server or servers to be used as the authentication server. See *Specifying RADIUS Server Connections on Switches (CLI Procedure)*.

Overview

The following configuration section shows the access profile configuration, the 802.1X authentication configuration, and finally the VLANs (including PVLANs) configuration.

Configuring 802.1X Authentication with Private VLANs in One Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

```
[edit]
set access radius-server 10.20.9.199 port 1812
set access radius-server 10.20.9.199 secret "$9$Lqa7dsaZjP5F245Fn/0OX7-V24JGDkmf"
set access profile dot1x-auth authentication-order radius
set access profile authp authentication-order radius
set access profile authp radius authentication-server 10.204.96.165
set switch-options voip interface ge-0/0/8.0 vlan voip
set interfaces ge-0/0/8 unit 0 family ethernet-switching interface-mode access
set interfaces ge-0/0/8 unit 0 family ethernet-switching vlan members data
set protocols dot1x authenticator authentication-profile-name authp
set protocols dot1x authenticator interface ge-0/0/8.0 supplicant multiple
set protocols dot1x authenticator interface ge-0/0/8.0 mac-radius
set vlans community vlan-id 20
set vlans community private-vlan community
set vlans community-one vlan-id 30
set vlans community-one private-vlan community
set vlans isolated vlan-id 200
set vlans isolated private-vlan isolated
set vlans pvlan vlan-id 2000
set vlans pvlan isolated-vlan isolated
set vlans pvlan community-vlans [community community-one]
set vlans data vlan-id 43
set vlans voip vlan-id 33
```

Step-by-Step Procedure

To configure 802.1X authentication and PVLANs in one configuration:

1. Configure the access profile:

```
[edit access]
set radius-server 10.20.9.199 port 1812
set radius-server 10.20.9.199 secret "$9$Lqa7dsaZjP5F245Fn/0OX7-V24JGDkmf"
set profile dot1x-auth authentication-order radius
set profile authp authentication-order radius
set profile authp radius authentication-server 10.204.96.165
[edit switch-options]
set voip interface ge-0/0/8.0 vlan voip
```

NOTE: The configured VoIP VLAN cannot be a PVLAN (primary, community, or isolated).

2. Configure the 802.1X settings:

```
[edit interfaces]
set ge-0/0/8 unit 0 family ethernet-switching interface-mode access
set ge-0/0/8 unit 0 family ethernet-switching vlan members data
[edit protocols]
set dot1x authenticator authentication-profile-name authp
set dot1x authenticator interface ge-0/0/8.0 supplicant multiple
set dot1x authenticator interface ge-0/0/8.0 mac-radius
```

NOTE: The configured data VLAN could also be a community VLAN or an isolated VLAN.

3. Configure the VLANs (including the PVLANs):

```
[edit vlans]
set community vlan-id 20
set community private-vlan community
set community-one vlan-id 30
set community-one private-vlan community
set isolated vlan-id 200
set isolated private-vlan isolated
set pvlan vlan-id 2000
```

```
set pvlan isolated-vlan isolated
set pvlan community-vlans [community community-one]
set data vlan-id 43
set voip vlan-id 33
```

Results

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

```
user@switch# show access
radius-server {
    10.20.9.199 {
        port 1812;
        secret "$9$Lqa7dsaZjP5F245Fn/0OX7-V24JGDkmf"; ## SECRET-DATA
    }
}
profile dot1x-auth {
    authentication-order radius;
}
profile authp {
    authentication-order radius;
    radius {
        authentication-server 10.204.96.165;
    }
}
user@switch# show interfaces
ge-0/0/8 {
    unit 0 {
        family ethernet-switching {
            interface-mode access;
            vlan {
                members data;
            }
        }
    }
}
user@switch# show protocols
dot1x {
    authenticator {
```

```
        authentication-profile-name authp;
        interface {
            ge-0/0/8.0 {
                supplicant multiple;
                mac-radius;
            }
        }
    }
}
user@switch# show switch-options
voip {
    interface ge-0/0/8.0 {
        vlan voip;
    }
}
user@switch# show vlans
community {
    vlan-id 20;
    private-vlan community;
}
community-one {
    vlan-id 30;
    private-vlan community;
}
data {
    vlan-id 43;
}
isolated {
    vlan-id 200;
    private-vlan isolated;
}
pvlan {
    vlan-id 2000;
    isolated-vlan isolated;
    community-vlans [community community-one];
}
voip {
    vlan-id 33;
}
```

Verification

IN THIS SECTION

Verify That Client MAC Addresses Are Learned on the Primary VLAN

Purpose

Show that a client MAC address has been learned on the primary VLAN.

Action

```
user@switch> show ethernet-switching table
MAC flags (S - static MAC, D - dynamic MAC, L - locally learned, P - Persistent static, C -
Control MAC, SE - statistics enabled, NM - non configured MAC, R - remote PE MAC, O - ovsdb MAC)
Ethernet switching table : 1 entries, 1 learned
Routing instance : default-switch
    Vlan        MAC               MAC    Age  Logical      NH     RTR
    name        address           flags       interface    Index  ID
    pvlan       00:30:48:8C:66:BD D      -    ge-0/0/8.0   0      0
```

Verify That the Primary VLAN Is an Authenticated VLAN

Purpose

Show that the primary VLAN is shown as an authenticated VLAN.

Action

```
user@switch> show dot1x interface ge-0/0/8.0 detail
ge-0/0/8.0
  Role: Authenticator
  Administrative state: Auto
```

```
Supplicant mode: Multiple
Number of retries: 3
Quiet period: 60 seconds
Transmit period: 30 seconds
Mac Radius: Enabled
Mac Radius Strict: Disabled
Reauthentication: Enabled Reauthentication interval: 40 seconds
Supplicant timeout: 30 seconds
Server timeout: 30 seconds
Maximum EAPOL requests: 1
Guest VLAN member: <not configured>
Number of connected supplicants: 1
  Supplicant: user5, 00:30:48:8C:66:BD
    Operational state: Authenticated
    Authentication method: Radius
    Authenticated VLAN: pvlan
    Reauthentication due in 17 seconds
```

Putting Access Port Security on Private VLANs

IN THIS SECTION

Understanding Access Port Security on PVLANs

You can now enable access port security features, such as DHCP snooping, on private VLANs (PVLANs).

For security reasons, it is often useful to restrict the flow of broadcast and unknown unicast traffic and to even limit the communication between known hosts. The PVLAN feature allows you to split a broadcast domain into multiple isolated broadcast subdomains, essentially putting a VLAN inside a VLAN.

Ethernet LANs are vulnerable to attacks such as address spoofing (forging) and Layer 2 denial of service (DoS) on network devices. The following access port security features help protect your device against

losses of information and productivity that such attacks can cause, and you can now configure these security features on a PVLAN:

- DHCP snooping—Filters and blocks ingress DHCP server messages on untrusted ports. DHCP snooping builds and maintains a database of DHCP lease information, which is called the DHCP snooping database.
- DHCPv6 snooping—DHCP snooping for IPv6.
- DHCP option 82—Also known as the DHCP Relay Agent Information option. Helps protect the switch against attacks such as spoofing of IP addresses and MAC addresses and DHCP IP address starvation. Option 82 provides information about the network location of a DHCP client. The DHCP server uses this information to implement IP addresses or other parameters for the client.
- DHCPv6 options:
  - Option 37—Remote ID option for DHCPv6; inserts information about the network location of the remote host into DHCPv6 packets.
  - Option 18—Circuit ID option for DHCPv6; inserts information about the client port into DHCPv6 packets.
  - Option 16—Vendor ID option for DHCPv6; inserts information about the vendor of the client hardware into DHCPv6 packets.
- Dynamic ARP inspection (DAI)—Prevents Address Resolution Protocol (ARP) spoofing attacks. ARP requests and replies are compared against entries in the DHCP snooping database, and filtering decisions are made on the basis of the results of those comparisons.
- IP source guard—Mitigates the effects of IP address spoofing attacks on the Ethernet LAN; validates the source IP address in the packet sent from an untrusted access interface against the DHCP snooping database. If the packet cannot be validated, it is discarded.
- IPv6 source guard—IP source guard for IPv6.
- IPv6 neighbor discovery inspection—Prevents IPv6 address spoofing attacks; compares neighbor discovery requests and replies against entries in the DHCPv6 snooping database, and filtering decisions are made on the basis of the results of those comparisons.

NOTE: This document does not provide detailed information about access port security features or PVLANs. For those details, see the feature documentation that is specific to those individual features. For access port security, see Security Services Administration Guide. For PVLANs, see Ethernet Switching User Guide.

Configuration Guidelines for Putting Access Port Security Features on PVLANs

Keep the following guidelines and limitations in mind for configuring access port security features on PVLANs:

- You must apply the *same* access port security features on both the primary vlan and all its secondary VLANs.
- A PVLAN can have only one integrated routing and bridging (IRB) interface, and the IRB interface must be on the primary VLAN.
- Limitations on access port security configurations on PVLANs are the same as those for access port security features configurations that are not in PVLANs. See the access port security documentation at Security Services Administration Guide.

Example: Configuring Access Port Security on a PVLAN

IN THIS SECTION

Requirements

- Junos OS Release 18.2R1 or later
- EX4300 switch

Overview

The following configuration section shows:

- Configuration of a private VLAN, with the primary VLAN (`vlan-pri`) and its three secondary VLANs—community VLANs (`vlan-hr` and `vlan-finance`) and isolated VLAN (`vlan-iso`).
- Configuration of the interfaces that are used to send communications between the interfaces on those VLANs.
- Configuration of access security features on the primary and secondary VLANs that make up the PVLAN.

Figure 22: Topology

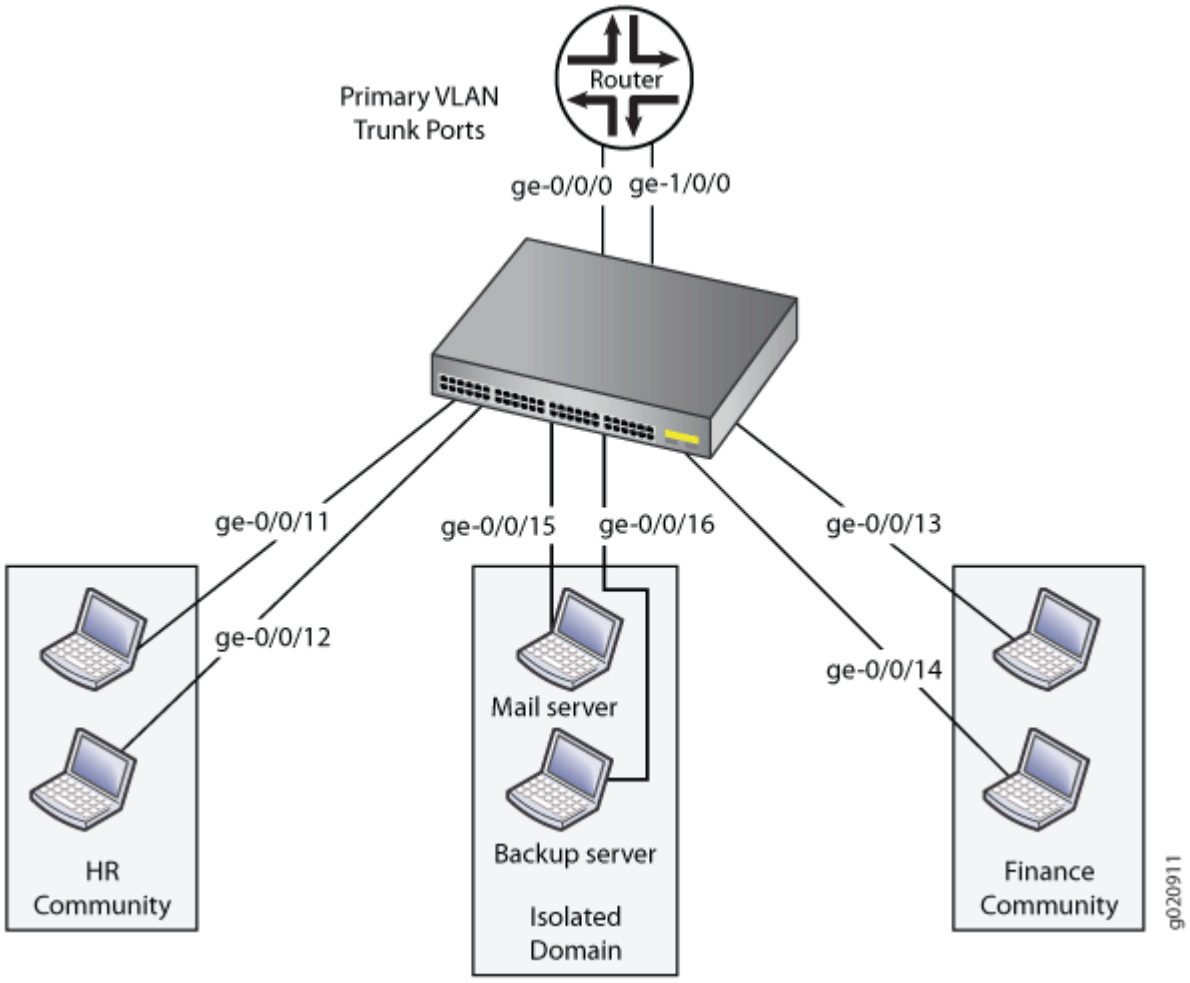


Table 5 lists the settings for the example topology.

Table 78: Components of the Topology for Configuring a PVLAN with Access Port Security Features

Interface	Description
ge-0/0/0.0	Primary VLAN (vlan1-pri) trunk interface
ge-0/0/11.0	User 1, HR Community (vlan-hr)
ge-0/0/12.0	User 2, HR Community (vlan-hr)

Table 78: Components of the Topology for Configuring a PVLAN with Access Port Security Features *(Continued)*

Interface	Description
ge-0/0/13.0	User 3, Finance Community (vlan-finance)
ge-0/0/14.0	User 4, Finance Community (vlan-finance)
ge-0/0/15.0	Mail server, Isolated (vlan-iso)
ge-0/0/16.0	Backup server, Isolated (vlan-iso)
ge-1/0/0.0	Primary VLAN (vlan-pri) trunk interface

Configuring Access Port Security on a PVLAN

IN THIS SECTION

- Procedure | 472

Procedure

CLI Quick Configuration

```
set vlans vlan-pri vlan-id 100
set vlans vlan-hr private-vlan community vlan-id 200
set vlans vlan-finance private-vlan community vlan-id 300
set vlans vlan-iso private-vlan isolated vlan-id 400
set vlans vlan-pri community-vlan vlan-hr
set vlans vlan-pri community-vlan vlan-finance
set vlans vlan-pri isolated-vlan vlan-iso
set interfaces ge-0/0/11 unit 0 family ethernet-switching interface-mode access vlan members
vlan-hr
set interfaces ge-0/0/12 unit 0 family ethernet-switching interface-mode trunk vlan members vlan-
```

```
hr
set interfaces ge-0/0/13 unit 0 family ethernet-switching interface-mode access vlan members
vlan-finance
set interfaces ge-0/0/14 unit 0 family ethernet-switching interface-mode trunk vlan members vlan-
finance
set interfaces ge-0/0/15 unit 0 family ethernet-switching interface-mode access vlan members
vlan-iso
set interfaces ge-0/0/16 unit 0 family ethernet-switching interface-mode access vlan members
vlan-iso
set interfaces ge-0/0/0 unit 0 family ethernet-switching interface-mode trunk vlan members vlan-
pri
set interfaces ge-1/0/0 unit 0 family ethernet-switching interface-mode trunk vlan members vlan-
pri
set vlans vlan-pri forwarding-options dhcp-security arp-inspection
set vlans vlan-pri forwarding-options dhcp-security ip-source-guard
set vlans vlan-pri forwarding-options dhcp-security ipv6-source-guard
set vlans vlan-pri forwarding-options dhcp-security neighbor-discovery-inspection
set vlans vlan-pri forwarding-options dhcp-security option-82
set vlans vlan-pri forwarding-options dhcp-security dhcpv6-options option-16
set vlans vlan-pri forwarding-options dhcp-security dhcpv6-options light-weight-dhcpv6-relay
set vlans vlan-hr forwarding-options dhcp-security arp-inspection
set vlans vlan-hr forwarding-options dhcp-security ip-source-guard
set vlans vlan-hr forwarding-options dhcp-security ipv6-source-guard
set vlans vlan-hr forwarding-options dhcp-security neighbor-discovery-inspection
set vlans vlan-hr forwarding-options dhcp-security option-82
set vlans vlan-hr forwarding-options dhcp-security dhcpv6-options option-16
set vlans vlan-hr forwarding-options dhcp-security dhcpv6-options light-weight-dhcpv6-relay
set vlans vlan-finance forwarding-options dhcp-security arp-inspection
set vlans vlan-finance forwarding-options dhcp-security ip-source-guard
set vlans vlan-finance forwarding-options dhcp-security ipv6-source-guard
set vlans vlan-finance forwarding-options dhcp-security neighbor-discovery-inspection
set vlans vlan-finance forwarding-options dhcp-security option-82
set vlans vlan-finance forwarding-options dhcp-security dhcpv6-options option-16
set vlans vlan-finance forwarding-options dhcp-security dhcpv6-options light-weight-dhcpv6-relay
set vlans vlan-iso forwarding-options dhcp-security arp-inspection
set vlans vlan-iso forwarding-options dhcp-security ip-source-guard
set vlans vlan-iso forwarding-options dhcp-security ipv6-source-guard
set vlans vlan-iso forwarding-options dhcp-security neighbor-discovery-inspection
set vlans vlan-iso forwarding-options dhcp-security option-82
set vlans vlan-iso forwarding-options dhcp-security dhcpv6-options option-16
set vlans vlan-iso forwarding-options dhcp-security dhcpv6-options light-weight-dhcpv6-relay
```

Step-by-Step Procedure

To configure a private VLAN (PVLAN) and then configure access port security features on that PVLAN:

1. Configure the PVLAN—Create the primary VLAN and its secondary VLANs and assign VLAN IDs to them. Associate interfaces with the VLANs. (For details on configuring VLANs, see Configuring VLANs for EX Series Switches with ELS Support (CLI Procedure).)

   a.
   ```
   [edit vlans]
   user@switch# set vlan-pri vlan-id 100
   user@switch# set vlan-hr private-vlan community vlan-id 200
   user@switch# set vlan-finance private-vlan community vlan-id 300
   user@switch# set vlan-iso private-vlan isolated vlan-id 400
   user@switch# set vlan-pri community-vlan vlan-hr
   user@switch# set vlan-pri community-vlan vlan-finance
   user@switch# set vlan-pri isolated-vlan vlan-iso
   ```

   b.
   ```
   [edit interfaces]
   user@switch# set ge-0/0/11 unit 0 family ethernet-switching interface-mode access vlan
   members vlan-hr
   user@switch# set ge-0/0/12 unit 0 family ethernet-switching interface-mode trunk vlan
   members vlan-hr
   user@switch# set ge-0/0/13 unit 0 family ethernet-switching interface-mode access vlan
   members vlan-finance
   user@switch# set ge-0/0/14 unit 0 family ethernet-switching interface-mode trunk vlan
   members vlan-finance
   user@switch# set ge-0/0/15 unit 0 family ethernet-switching interface-mode access vlan
   members vlan-iso
   user@switch# set ge-0/0/16 unit 0 family ethernet-switching interface-mode access vlan
   members vlan-iso
   user@switch# set ge-0/0/0 unit 0 family ethernet-switching interface-mode trunk vlan
   members vlan-pri
   user@switch# set ge-1/0/0 unit 0 family ethernet-switching interface-mode trunk vlan
   members vlan-pri
   ```

2. Configure access port security features on the primary VLAN and all its secondary VLANs:

NOTE: When you configure ARP inspection, IP source guard, IPv6 source guard, neighbor discovery inspection, DHCP option 82, or DHCPv6 options, then DHCP snooping and DHCPv6 snooping are automatically configured.

```
[edit vlans]
user@switch# set vlan-pri forwarding-options dhcp-security arp-inspection
user@switch# set vlan-pri forwarding-options dhcp-security ip-source-guard
user@switch# set vlan-pri forwarding-options dhcp-security ipv6-source-guard
user@switch# set vlan-pri forwarding-options dhcp-security neighbor-discovery-inspection
user@switch# set vlan-pri forwarding-options dhcp-security option-82
user@switch# set vlan-pri forwarding-options dhcp-security dhcpv6-options option-16
user@switch# set vlan-pri forwarding-options dhcp-security dhcpv6-options light-weight-dhcpv6-
relay
user@switch# set vlan-hr forwarding-options dhcp-security arp-inspection
user@switch# set vlan-hr forwarding-options dhcp-security ip-source-guard
user@switch# set vlan-hr forwarding-options dhcp-security ipv6-source-guard
user@switch# set vlan-hr forwarding-options dhcp-security neighbor-discovery-inspection
user@switch# set vlan-hr forwarding-options dhcp-security option-82
user@switch# set vlan-hr forwarding-options dhcp-security dhcpv6-options option-16
user@switch# set vlan-hr forwarding-options dhcp-security dhcpv6-options light-weight-dhcpv6-
relay
user@switch# set vlan-finance forwarding-options dhcp-security arp-inspection
user@switch# set vlan-finance forwarding-options dhcp-security ip-source-guard
user@switch# set vlan-finance forwarding-options dhcp-security ipv6-source-guard
user@switch# set vlan-finance forwarding-options dhcp-security neighbor-discovery-inspection
user@switch# set vlan-finance forwarding-options dhcp-security option-82
user@switch# set vlan-finance forwarding-options dhcp-security dhcpv6-options option-16
user@switch# set vlan-finance forwarding-options dhcp-security dhcpv6-options light-weight-
dhcpv6-relay
user@switch# set vlan-iso forwarding-options dhcp-security arp-inspection
user@switch# set vlan-iso forwarding-options dhcp-security ip-source-guard
user@switch# set vlan-iso forwarding-options dhcp-security ipv6-source-guard
user@switch# set vlan-iso forwarding-options dhcp-security neighbor-discovery-inspection
user@switch# set vlan-iso forwarding-options dhcp-security option-82
user@switch# set vlan-iso forwarding-options dhcp-security dhcpv6-options option-16
user@switch# set vlan-iso forwarding-options dhcp-security dhcpv6-options light-weight-dhcpv6-
relay
```

Results

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

```
[edit]
user@switch#  show interfaces
ge-0/0/0 {
    unit 0 {
        family ethernet-switching {
            interface-mode trunk;
            vlan {
                members vlan-pri;
            }
        }
    }
}
ge-1/0/0 {
    unit 0 {
        family ethernet-switching {
            interface-mode trunk;
            vlan {
                members vlan-pri;
            }
        }
    }
}
ge-0/0/11 {
    unit 0 {
        family ethernet-switching {
            interface-mode access;
            vlan {
                members vlan-hr;
            }
        }
    }
}
ge-0/0/12 {
    unit 0 {
        family ethernet-switching {
            interface-mode access;
```

```
            vlan {
                members vlan-hr;
            }
        }
    }
}
ge-0/0/13 {
    unit 0 {
        family ethernet-switching {
            interface-mode access;
            vlan {
                members vlan-hr;
            }
        }
    }
}
ge-0/0/14 {
    unit 0 {
        family ethernet-switching {
            interface-mode access;
            vlan {
                members vlan-hr;
            }
        }
    }
}
ge-0/0/15 {
    unit 0 {
        family ethernet-switching {
            interface-mode access;
            vlan {
                members vlan-iso;
            }
        }
    }
}
ge-0/0/16 {
    unit 0 {
        family ethernet-switching {
            interface-mode access;
            vlan {
                members vlan-iso;
            }
```

```
        }
    }
}
user@switch#  show vlans
vlan-finance {
    vlan-id 300;
    private-vlan community;
    interface {
        ge-0/0/13.0;
        ge-0/0/14.0;
    }
    forwarding-options {
        dhcp-security {
            arp-inspection;
                ip-source-guard;
                neighbor-discovery-inspection;
                ipv6-source-guard;
                option-82;
                dhcpv6-options light-weight-dhcpv6-relay;
            dhcpv6-options option-16;
        }
    }
}
vlan-hr {
    vlan-id 200;
    private-vlan community;
    interface {
        ge-0/0/11.0;
        ge-0/0/12.0;
    }
    forwarding-options {
        dhcp-security {
            arp-inspection;
                ip-source-guard;
                neighbor-discovery-inspection;
                ipv6-source-guard;
                option-82;
                dhcpv6-options light-weight-dhcpv6-relay;
            dhcpv6-options option-16;
        }
    }
}
vlan-iso {
```

```
    vlan-id 400;
    private-vlan isolated;
    interface {
        ge-0/0/15.0;
        ge-0/0/16.0;
    }
    forwarding-options {
        dhcp-security {
            arp-inspection;
                ip-source-guard;
                neighbor-discovery-inspection;
                ipv6-source-guard;
                option-82;
                dhcpv6-options light-weight-dhcpv6-relay;
            dhcpv6-options option-16;
        }
    }
}
vlan-pri {
    vlan-id 100;
        community-vlan vlan-finance;
        community-vlan vlan-hr;
    isolated-vlan vlan-iso;
    interface {
        ge-0/0/0.0;
        ge-1/0/0.0;
    }
    forwarding-options {
        dhcp-security {
            arp-inspection;
                ip-source-guard;
                neighbor-discovery-inspection;
                ipv6-source-guard;
                option-82;
                dhcpv6-options light-weight-dhcpv6-relay;
            dhcpv6-options option-16;
        }
    }
}
```

Verification

IN THIS SECTION

- Verify That Access Security Features Are Working as Expected | 480

Verify That Access Security Features Are Working as Expected

Purpose

Verify that the access port security features that you configured on your PVLAN are working as expected.

Action

Use the `show dhcp-security` and the `clear dhcp-security` CLI commands to verify that the features are working as expected. See details about those commands in Security Services Administration Guide.

Creating a Private VLAN on a Single Switch with ELS Support (CLI Procedure)

NOTE: This task uses Junos OS for switches with support for the Enhanced Layer 2 Software (ELS) configuration style. If your EX Series switch runs software that does not support ELS, see "Creating a Private VLAN on a Single EX Series Switch (CLI Procedure)" on page 485. For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15.

NOTE: Private VLANs are not supported on QFX5100 switches and QFX10002 switches running Junos OS Release 15.1X53.

For security reasons, it is often useful to restrict the flow of broadcast and unknown unicast traffic or limit the communication between known hosts. Private VLANs (PVLANs) enable you to split a broadcast domain (primary VLAN) into multiple isolated broadcast subdomains (secondary VLANs), essentially putting a VLAN inside a VLAN. This procedure describes how to create a PVLAN on a single switch.

NOTE: You must specify a VLAN ID for each secondary VLAN even if the PVLAN is configured on a single switch.

You do not need to preconfigure the primary VLAN. This topic shows the primary VLAN being configured as part of this PVLAN configuration procedure.

For a list of guidelines on configuring PVLANs, see "Understanding Private VLANs" on page 428.

To configure a private VLAN on a single switch:

1. Set the VLAN ID for the primary VLAN:

```
[edit vlans]
user@switch# set primary-vlan-name vlan-id vlan-id-number
```

2. Configure at least one interface within the primary VLAN so that it communicates with all the subdomains of the PVLAN. This interface functions as a *promiscuous* port. It can be either a trunk port or an access port.

```
[edit interfaces]
user@switch# set interface-name unit 0 family ethernet-switching
user@switch# set interface-name unit 0 family ethernet-switching vlan members primary-vlan-name
```

3. Configure another promiscuous interface of the primary VLAN as a trunk port to connect the PVLAN to the external router or switch:

```
[edit interfaces]
user@switch# set interface-name unit 0 family ethernet-switching interface-mode trunk
user@switch# set interface-name unit 0 family ethernet-switching vlan members primary-vlan-name
```

4. Create an isolated VLAN by selecting the `isolated` option for `private-vlan`, and setting a VLAN ID for the isolated VLAN:

```
[edit vlans]
user@switch# set isolated-vlan-name private-vlan isolated vlan-id isolated-vlan-id
```

NOTE: You can create only one isolated VLAN within a private VLAN. Setting the VLAN name for the isolated VLAN is optional. Configuring the VLAN ID is required.

5. Create a community VLAN by selecting the `community` option for `private-vlan`, and setting a VLAN ID for this community VLAN:

```
[edit vlans]
user@switch# set community-vlan-name private-vlan community vlan-id community-vlan-id
```

NOTE: To create additional community VLANs, repeat this step and specify a different name for the community VLAN. Setting the VLAN name for the community VLAN is optional. Configuring the VLAN ID is required.

6. Associate the isolated VLAN with the primary VLAN:

```
[edit vlans]
user@switch# set primary-vlan-name vlan-id primary-vlan-id isolated-vlan isolated-vlan-name
```

7. Associate each community VLAN with the primary VLAN:

```
[edit vlans]
user@switch# set primary-vlan-name vlan-id primary-vlan-id community-vlan community-vlan-name
```

8. If you have not already done so, configure at least one interface of the isolated VLAN.

```
[edit interfaces]
user@switch# set interface-name unit logical-unit-number family ethernet-switching interface-
mode access vlan members isolated-vlan-name
```

9. If you have not already done so, configure at least one interface of the community VLAN.

```
[edit interfaces]
user@switch# set interface-name unit logical-unit-number family ethernet-switching interface-
mode access vlan members community-vlan-name
```

NOTE: Repeat the same step on other community VLANs that you want to include in the PVLAN.

Creating a Private VLAN on a Single QFX Switch without ELS Support

NOTE: If your switch runs software that supports ELS, see "Creating a Private VLAN on a Single Switch with ELS Support (CLI Procedure)" on page 494.

For security reasons, it is often useful to restrict the flow of broadcast and unknown unicast traffic and to even limit the communication between known hosts. The private VLAN (PVLAN) feature allows you to split a broadcast domain into multiple isolated broadcast subdomains, essentially putting a secondary VLAN inside a primary VLAN. This topic describes how to configure a PVLAN on a single switch.

Before you begin, configure names for all secondary VLANs that will be part of the primary VLAN. (You do not need to preconfigure the primary VLAN—it is configured as part of this procedure.) You do not need to create VLAN IDs (tags) for the secondary VLANs. It does not impair functioning if you tag the secondary VLANS, but tags are not used when secondary VLANs are configured on a single switch.

Keep these rules in mind when configuring a PVLAN:

- The primary VLAN must be a tagged VLAN.
- If you are going to configure a community VLAN, you must first configure the primary VLAN and the PVLAN trunk port. You must also configure the primary VLAN to be private using the *pvlan* statement.
- If you are going to configure an isolated VLAN, you must first configure the primary VLAN and the PVLAN trunk port.

If you complete your configuration steps in the order shown, you will not violate these PVLAN rules. To configure a private VLAN on a single switch:

1. Set the name and VLAN ID (802.1Q tag) for the primary VLAN:

```
[edit vlans]
user@switch# set primary-vlan-name vlan-id vlan-id-number
```

2. Configure the VLAN to be private:

```
[edit vlans]
user@switch# set primary-vlan-name pvlan
```

3. Configure the trunk interfaces for the primary VLAN:

```
[edit interfaces]
user@switch# set interface-name unit 0 family ethernet-switching port-mode trunk
user@switch# set interface-name unit 0 family ethernet-switching vlan members primary-vlan-name
```

4. Add the trunk interfaces to the primary VLAN:

```
[edit vlans]
user@switch# set primary-vlan-name interface interface-name
```

5. Configure the access interfaces for the community (secondary) VLANs:

```
[edit interfaces]
user@switch# set interface-name unit 0 family ethernet-switching port-mode access
```

6. Add the access interfaces to the community VLANs:

```
[edit vlans]
user@switch# set community-vlan-name interface interface-name
```

7. For each community VLAN, set the primary VLAN:

```
[edit vlans]
user@switch# set community-vlan-name primary-vlan primary-vlan-name
```

8. Configure isolated ports:

```
[edit vlans]
user@switch# set primary-vlan-name interface interface-name isolated
```

Creating a Private VLAN on a Single EX Series Switch without ELS Support (CLI Procedure)

NOTE: If your switch runs software that supports ELS, see "Creating a Private VLAN on a Single Switch with ELS Support (CLI Procedure)" on page 494.

For security reasons, it is often useful to restrict the flow of broadcast and unknown unicast traffic and to even limit the communication between known hosts. The private VLAN (PVLAN) feature on EX Series switches enables you to split a broadcast domain, also known as a primary VLAN, into multiple isolated broadcast subdomains, also known as secondary VLANs. Splitting the primary VLAN into secondary VLANs essentially nests a VLAN inside another VLAN. This topic describes how to configure a PVLAN on a single switch.

Before you begin, configure names for all secondary VLANs that will be part of the primary VLAN. (Unlike the secondary VLANs, you do not need to preconfigure the primary VLAN—this procedure provides the complete configuration of the primary VLAN.) Although tags are not needed when a secondary VLAN is configured on a single switch, configuring a secondary VLAN as tagged does not adversely affect its functionality. For instructions on configuring the secondary VLANs, see *Configuring VLANs for EX Series Switches*.

Keep these rules in mind when configuring a PVLAN on a single switch:

- The primary VLAN must be a tagged VLAN.
- Configuring a VoIP VLAN on PVLAN interfaces is not supported.

To configure a private VLAN on a single switch:

1. Set the VLAN ID for the primary VLAN:

```
[edit vlans]
user@switch# set primary-vlan-name vlan-id vlan-id-number
```

2. Set the interfaces and port modes:

```
[edit interfaces]
user@switch# set interface-name unit 0 family ethernet-switching port-mode mode
user@switch# set interface-name unit 0 family ethernet-switching vlan members (all | vlan-id
| vlan-number)
```

3. Configure the access ports in the primary VLAN to not forward packets to one another:

```
[edit vlans]
user@switch# set vlan-id vlan-id-number no-local-switching
```

4. For each community VLAN, configure access interfaces:

```
[edit vlans]
user@switch# set community-vlan-name interface-mac-limit interface-name
```

5. For each community VLAN, set the primary VLAN:

```
[edit vlans]
user@switch# set community-vlan-name primary-vlan primary-vlan-name
```

Isolated VLANs are not configured as part of this process. Instead, they are created internally if **no-local-switching** is enabled on the primary VLAN and the isolated VLAN has access interfaces as members.

To optionally enable routing between isolated and community VLANs by using a routed VLAN interface (RVI) instead of a promiscuous port connected to a router, see "Configuring a Routed VLAN Interface in a Private VLAN on an EX Series Switch" on page 759.

NOTE: Only an EX8200 switch or EX8200 Virtual Chassis support the use of an RVI to route Layer 3 traffic between isolated and community VLANs in a PVLAN domain.

Creating a Private VLAN Spanning Multiple QFX Series Switches without ELS Support

NOTE: If your switch runs software that supports ELS, see "Creating a Private VLAN Spanning Multiple EX Series Switches with ELS Support (CLI Procedure)" on page 488.

For security reasons, it is often useful to restrict the flow of broadcast and unknown unicast traffic and to even limit the communication between known hosts. The private VLAN (PVLAN) feature allows you to split a broadcast domain into multiple isolated broadcast subdomains, essentially putting a secondary VLAN inside a primary VLAN. This topic describes how to configure a PVLAN to span multiple switches.

Before you begin, configure names for all secondary VLANs that will be part of the primary VLAN. (You do not need to preconfigure the primary VLAN—it is configured as part of this procedure.) You do not need to create VLAN IDs (tags) for the secondary VLANs. It does not impair functioning if you tag the secondary VLANS, but tags are not used when secondary VLANs are configured on a single switch.

The following rules apply to creating PVLANs:

- The primary VLAN must be a tagged VLAN.
- If you are going to configure a community VLAN, you must first configure the primary VLAN and the PVLAN trunk port. You must also configure the primary VLAN to be private using the *pvlan* statement.
- If you are going to configure an isolated VLAN, you must first configure the primary VLAN and the PVLAN trunk port.

If you complete your configuration steps in the order shown, you will not violate these PVLAN rules. To configure a private VLAN to span multiple switches:

1. Set the name and VLAN ID (802.1Q tag) for the primary VLAN:

```
[edit vlans]
user@switch# set primary-vlan-name vlan-id vlan-id-number
```

2. Configure the VLAN to be private:

```
 [edit vlans]
user@switch# set primary-vlan-name pvlan
```

3. Configure the trunk interfaces for the primary VLAN:

```
[edit interfaces]
user@switch# set interface-name unit 0 family ethernet-switching port-mode trunk
user@switch# set interface-name unit 0 family ethernet-switching vlan members primary-vlan-
name
```

4. Add the trunk interfaces to the primary VLAN:

```
 [edit vlans]
user@switch# set primary-vlan-name interface interface-name
```

5. Configure the access interfaces for the community (secondary) VLANs:

```
[edit interfaces]
user@switch# set interface-name unit 0 family ethernet-switching port-mode access
```

6. Add the access interfaces to the community VLANs:

```
[edit vlans]
user@switch# set community-vlan-name interface interface-name
```

7. For each community VLAN, set the primary VLAN:

```
[edit vlans]
user@switch# set community-vlan-name primary-vlan primary-vlan-name
```

8. Configure an isolated VLAN ID to create an interswitch isolated domain that spans the switches:

```
[edit vlans]
user@switch# set primary-vlan-name isolation-vlan-id number
```

9. Configure isolated ports:

```
[edit vlans]
user@switch# set primary-vlan-name interface interface-name isolated
```

Creating a Private VLAN Spanning Multiple EX Series Switches with ELS Support (CLI Procedure)

NOTE: This task uses Junos OS for EX Series switches with support for the Enhanced Layer 2 Software (ELS) configuration style If your switch runs software that does not support ELS, see "Creating a Private VLAN Spanning Multiple EX Series Switches (CLI Procedure)" on page 491. For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15.

NOTE: Private VLANs are not supported on QFX5100 switches and QFX10002 switches running Junos OS Release 15.1X53.

For security reasons, it is often useful to restrict the flow of broadcast and unknown unicast traffic or limit the communication between known hosts. Private VLANs (PVLANs) enable you to split a broadcast domain (primary VLAN) into multiple isolated broadcast subdomains (secondary VLANs), essentially putting a VLAN inside a VLAN. This procedure describes how to configure a PVLAN to span multiple switches.

For a list of guidelines on configuring PVLANs, see "Understanding Private VLANs" on page 428.

To configure a PVLAN to span multiple switches, perform the following procedure on all the switches that will participate in the PVLAN::

1. Create the primary VLAN by setting the unique VLAN name and specify an 802.1Q tag for the VLAN:

```
[edit vlans]
user@switch# set primary-vlan-name vlan-id number
```

2. On the switch that will connect to a router, configure a promiscuous interface as a trunk port to connect the PVLAN to the router:

```
[edit interfaces]
user@switch# set interface-name unit 0 family ethernet-switching interface-mode trunk
user@switch# set interface-name unit 0 family ethernet-switching vlan members primary-vlan-name
```

3. On all the switches, configure a trunk interface as the Inter-Switch Link (ISL) that will be used to connect the switches to each other:

```
[edit interfaces]
user@switch# set interface-name unit 0 family ethernet-switching interface-mode trunk inter-switch-link
user@switch# set interface-name unit 0 family ethernet-switching vlan members name-of-primary-vlan
```

4. Create an isolated VLAN within the primary VLAN by selecting the `isolated` option for `private-vlan`, and setting a VLAN ID for the isolated VLAN:

```
[edit vlans]
user@switch# set isolated-vlan-name private-vlan isolated vlan-id isolated-vlan-id
```

> **NOTE**: You can create only one isolated VLAN within a private VLAN. The isolated VLAN can contain member interfaces from the multiple switches that compose the PVLAN.
> Setting the VLAN name for the isolated VLAN is optional. Configuring the VLAN ID is required.

5. Create a community VLAN within the primary VLAN by selecting the `community` option for `private-vlan`, and setting a VLAN ID for this community VLAN:

```
[edit vlans]
user@switch# set community-vlan-name private-vlan community vlan-id community-vlan-id
```

> **NOTE**: To create additional community VLANs, repeat this step and specify a different name for the community VLAN. Setting the VLAN name for the community VLAN is optional. Configuring the VLAN ID is required.

6. Associate the isolated VLAN with the primary VLAN:

```
[edit vlans primary-vlan-name vlan-id primary-vlan-id]
user@switch# set isolated-vlan isolated-vlan-name
```

7. Associate each community VLAN with the primary VLAN:

```
[edit vlans primary-vlan-name vlan-id primary-vlan-id]
user@switch# set community-vlan community-vlan-name
```

8. If you have not already done so, configure at least one access interface to be a member of the isolated VLAN.

```
[edit interface]
user@switch# set interface-name unit logical-unit-number family ethernet-switching interface-
mode access vlan members isolated-vlan-name
```

9. If you have not already done so, configure at least one access interface to be a member of the community VLAN.

```
[edit interface]
user@switch# set interface-name unit logical-unit-number family ethernet-switching interface-
mode access vlan members community-vlan-name
```

NOTE: Repeat this step for the other community VLANs that you are including in the PVLAN.

Creating a Private VLAN Spanning Multiple EX Series Switches without ELS Support(CLI Procedure)

NOTE: If your switch runs software that supports ELS, see "Creating a Private VLAN Spanning Multiple EX Series Switches with ELS Support (CLI Procedure)" on page 488.

For security reasons, it is often useful to restrict the flow of broadcast and unknown unicast traffic and to even limit the communication between known hosts. The private VLAN (PVLAN) feature on EX Series switches enables an administrator to split a broadcast domain, also known as a primary VLAN, into multiple isolated broadcast subdomains, also known as secondary VLANs. Splitting the primary VLAN into secondary VLANs essentially nests a VLAN inside another VLAN. This topic describes how to configure a PVLAN to span multiple switches.

Before you begin, configure names for all secondary VLANs that will be part of the primary VLAN. (Unlike the secondary VLANs, you do not need to preconfigure the primary VLAN—this procedure provides the complete configuration of the primary VLAN.) For instructions on configuring the secondary VLANs, see *Configuring VLANs for EX Series Switches*.

The following rules apply to creating PVLANs:

- The primary VLAN must be a tagged VLAN.
- You must configure the primary VLAN and the PVLAN trunk port before configuring the secondary VLANs.
- Configuring a VoIP VLAN on PVLAN interfaces is not supported.
- If the Multiple VLAN Registration Protocol (MVRP) is configured on the PVLAN trunk port, the configuration of secondary VLANs and the PVLAN trunk port must be committed with the same commit operation.

To configure a private VLAN to span multiple switches:

1. Configure a name and an 802.1Q tag for the primary VLAN:.

```
[edit vlans]
user@switch# set primary-vlan-name vlan-id number
```

2. Set the primary VLAN to have no local switching:

```
 [edit vlans]
user@switch# set primary-vlan-name no-local-switching
```

3. Set the PVLAN trunk interface that will connect the primary VLAN to the neighboring switch:

```
[edit vlans]
user@switch# set primary-vlan-name interface interface-name pvlan-trunk
```

4. Configure a name and 802.1Q tag for a community VLAN that spans the switches:

```
[edit vlans]
user@switch# set community-vlan-name vlan-id number
```

5. Add access interfaces to the community VLAN:

```
 [edit vlans]
user@switch# set community-vlan-name interface interface-name
```

6. Specify the primary VLAN of the specified community VLAN:

```
[edit vlans]
user@switch# set community-vlan-name primary-vlan primary-vlan-name
```

7. Add the isolated interface to the specified primary VLAN:

```
[edit vlans]
user@switch# set primary-vlan-name interface interface-name
```

> **NOTE**: To configure an isolated interface, include it as one of the members of the primary VLAN, but do not configure it as belonging to one of the community VLANs.

8. Set the 802.1Q tag of the interswitch isolated VLAN:

```
[edit vlans]
user@switch# set primary-vlan-name isolation-id number
```

   802.1Q tags are required for interswitch isolated VLANs because IEEE 802.1Q uses an internal tagging mechanism by which a trunking device inserts a 4-byte VLAN frame identification tab into the packet header.

To optionally enable routing between isolated and community VLANs by using a routed VLAN interface (RVI) instead of a promiscuous port connected to a router, see "Configuring a Routed VLAN Interface in a Private VLAN on an EX Series Switch" on page 759.

> **NOTE**: Only an EX8200 switch or EX8200 Virtual Chassis support the use of an RVI to route Layer 3 traffic between isolated and community VLANs in a PVLAN domain.

Example: Configuring a Private VLAN on a Single Switch with ELS Support

IN THIS SECTION

NOTE: This example uses Junos OS for switches with support for the Enhanced Layer 2 Software (ELS) configuration style. If your EX switch runs software that does not support ELS, see "Example: Configuring a Private VLAN on a Single EX Series Switch" on page 507. For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15.

NOTE: Private VLANs are not supported on QFX5100 switches and QFX10002 switches running Junos OS Release 15.1X53.

For security reasons, it is often useful to restrict the flow of broadcast and unknown unicast traffic or limit the communication between known hosts. Private VLANs (PVLANs) enable you to split a broadcast domain (primary VLAN) into multiple isolated broadcast subdomains (secondary VLANs), essentially putting a VLAN inside a VLAN.

This example describes how to create a PVLAN on a single switch:

Requirements

This example uses the following hardware and software components:

- One Junos OS switch
- Junos OS Release 14.1X53-D10 or later for EX Series switches

  Junos OS Release 14.1X53-D15 or later for QFX Series switches

Overview and Topology

You can isolate groups of subscribers for improved security and efficiency. This configuration example uses a simple topology to illustrate how to create a PVLAN with one primary VLAN and three secondary VLANs (one isolated VLAN, and two community VLANs).

Table 79 on page 495 lists the interfaces of the topology used in the example.

Table 79: Interfaces of the Topology for Configuring a PVLAN

Interface	Description
ge-0/0/0 ge-1/0/0	Promiscuous member ports
ge-0/0/11, ge-0/0/12	HR community VLAN member ports
ge-0/0/13, ge-0/0/14	Finance community VLAN member ports
ge-0/0/15, ge-0/0/16	Isolated member ports

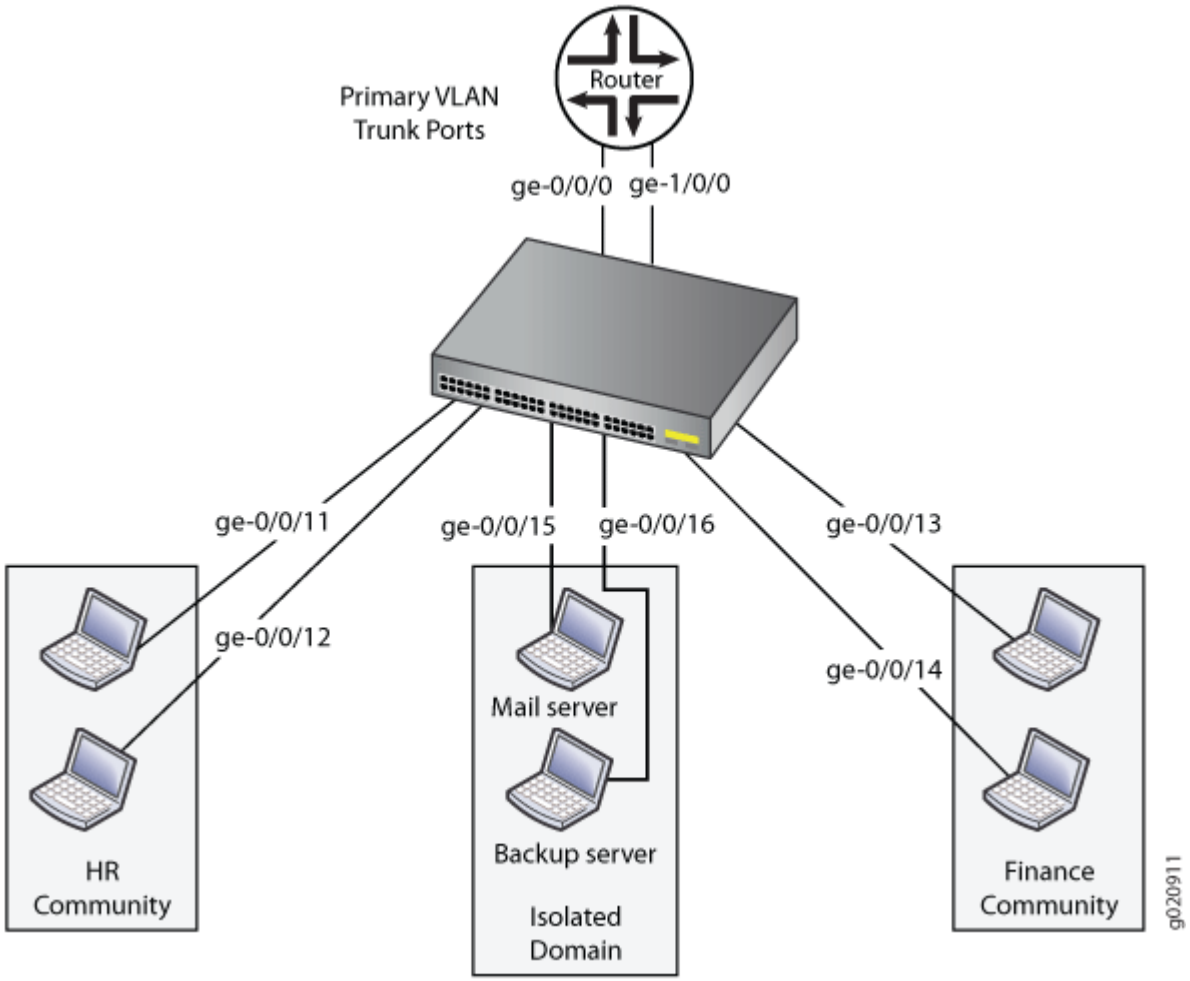
Table 80 on page 495 lists the VLAN IDs of the topology used in the example.

Table 80: VLAN IDs in the Topology for Configuring a PVLAN

VLAN ID	Description
100	Primary VLAN
200	HR community VLAN
300	Finance community VLAN
400	Isolated VLAN

Figure 23 on page 496 shows the topology for this example.

Figure 23: Topology of a Private VLAN on a Single EX Series Switch



Configuration

IN THIS SECTION

You can use an existing VLAN as the basis for your private PVLAN and create subdomains within it. This example creates a primary VLAN—using the VLAN name **vlan-pri**—as part of the procedure.

To configure a PVLAN, perform these tasks:

CLI Quick Configuration

To quickly create and configure a PVLAN, copy the following commands and paste them into the switch terminal window:

```
[edit]
set vlans vlan-pri vlan-id 100
set vlans vlan-iso private-vlan isolated vlan-id 400
set vlans vlan-hr private-vlan community vlan-id 200
set vlans vlan-finance private-vlan community vlan-id 300
set vlans vlan-pri vlan-id 100 isolated-vlan vlan-iso community-vlan vlan-hr community-vlan vlan-finance
set interface ge-0/0/11 unit 0 family ethernet-switching interface-mode access vlan members vlan-hr
set interface ge-0/0/12 unit 0 family ethernet-switching interface-mode trunk vlan members vlan-hr
set interface ge-0/0/13 unit 0 family ethernet-switching interface-mode access vlan members vlan-finance
set interface ge-0/0/14 unit 0 family ethernet-switching interface-mode trunk vlan members vlan-finance
set interface ge-0/0/15 unit 0 family ethernet-switching interface-mode access vlan members vlan-iso
set interface ge-0/0/16 unit 0 family ethernet-switching interface-mode access vlan members vlan-iso
set interface ge-0/0/0 unit 0 family ethernet-switching interface-mode trunk vlan members vlan-pri
set interface ge-1/0/0 unit 0 family ethernet-switching interface-mode trunk vlan members vlan-pri
```

Procedure

Step-by-Step Procedure

To configure the PVLAN:

1. Create the primary VLAN (in this example, the name is **vlan-pri**) of the private VLAN:

```
[edit vlans]
 user@switch# set vlan-pri vlan-id 100
```

2. Create an isolated VLAN and assign it a VLAN ID:

```
[edit vlans]
 user@switch# set vlan-iso private-vlan isolated vlan-id 400
```

3. Create the HR community VLAN and assign it a VLAN ID:

```
[edit vlans]
user@switch# set vlan-hr private-vlan community vlan-id 200
```

4. Create the finance community VLAN and assign it a VLAN ID:

```
[edit vlans]
user@switch# set vlan-finance private-vlan community vlan-id 300
```

5. Associate the secondary VLANs with the primary VLAN:

```
[edit vlans]
user@switch# set vlan-pri vlan-id 100 isolated-vlan vlan-iso community-vlan vlan-hr community-
vlan vlan-finance
```

6. Set the interfaces to the appropriate interface modes:

```
[edit interfaces]
user@switch# set ge-0/0/11 unit 0 family ethernet-switching interface-mode access vlan
members vlan-hr
user@switch# set ge-0/0/12 unit 0 family ethernet-switching interface-mode access vlan
members vlan-hr
user@switch# set ge-0/0/13 unit 0 family ethernet-switching interface-mode access vlan
members vlan-finance
user@switch# set ge-0/0/14 unit 0 family ethernet-switching interface-mode trunk vlan members
vlan-finance
user@switch# set ge-0/0/15 unit 0 family ethernet-switching interface-mode access vlan
members vlan-iso
user@switch# set ge-0/0/16 unit 0 family ethernet-switching interface-mode trunk vlan members
vlan-iso
```

7. Configure a promiscuous trunk interface of the primary VLAN. This interface is used by the primary VLAN to communicate with the secondary VLANs.

```
user@switch# set ge-0/0/0 unit 0 family ethernet-switching interface-mode trunk vlan members
vlan-pri
```

8. Configure another trunk interface (it is also a promiscuous interface) of the primary VLAN, connecting the PVLAN to the router.

```
user@switch# set ge-1/0/0 unit 0 family ethernet-switching interface-mode trunk  vlan members
vlan-pri
```

Example: Configuring a Private VLAN on a Single QFX Series Switch

IN THIS SECTION

For security reasons, it is often useful to restrict the flow of broadcast and unknown unicast traffic and even to limit the communication between known hosts. The private VLAN (PVLAN) feature allows an administrator to split a broadcast domain into multiple isolated broadcast subdomains, essentially putting a VLAN inside a VLAN.

This example describes how to create a PVLAN on a single switch:

Requirements

This example uses the following hardware and software components:

- One QFX3500 device
- Junos OS Release 12.1 or later for the QFX Series

Before you begin configuring a PVLAN, make sure you have created and configured the necessary VLANs. See "Configuring VLANs on QFX Series Switches without Enhanced Layer 2 Support" on page 152.

Overview and Topology

In a large office with multiple buildings and VLANs, you might need to isolate some workgroups or other endpoints for security reasons or to partition the broadcast domain. This configuration example shows a simple topology to illustrate how to create a PVLAN with one primary VLAN and two community VLANs, one for HR and one for finance, as well as two isolated ports—one for the mail server and the other for the backup server.

Table 81 on page 500 lists the settings for the sample topology.

Table 81: Components of the Topology for Configuring a PVLAN

Interface	Description
ge-0/0/0.0	Primary VLAN (pvlan100) trunk interface
ge-0/0/11.0	User 1, HR Community (hr-comm)
ge-0/0/12.0	User 2, HR Community (hr-comm)
ge-0/0/13.0	User 3, Finance Community (finance-comm)
ge-0/0/14.0	User 4, Finance Community (finance-comm)
ge-0/0/15.0	Mail server, Isolated (isolated)
ge-0/0/16.0	Backup server, Isolated (isolated)
ge-1/0/0.0	Primary VLAN (pvlan100) trunk interface

Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly create and configure a PVLAN, copy the following commands and paste them into the switch terminal window:

```
[edit]
set vlans pvlan100 vlan-id 100
set interfaces ge-0/0/0 unit 0 family ethernet-switching port-mode trunk
set interfaces ge-0/0/0 unit 0 family ethernet-switching vlan members pvlan
set interfaces ge-1/0/0 unit 0 family ethernet-switching port-mode trunk
set interfaces ge-1/0/0 unit 0 family ethernet-switching vlan members pvlan
set interfaces ge-0/0/11 unit 0 family ethernet-switching port-mode access
set interfaces ge-0/0/12 unit 0 family ethernet-switching port-mode access
set interfaces ge-0/0/13 unit 0 family ethernet-switching port-mode access
set interfaces ge-0/0/14 unit 0 family ethernet-switching port-mode access
set interfaces ge-0/0/15 unit 0 family ethernet-switching port-mode access
set interfaces ge-0/0/16 unit 0 family ethernet-switching port-mode access
set vlans pvlan100 pvlan
set vlans pvlan100 interface ge-0/0/0.0
set vlans pvlan100 interface ge-1/0/0.0
set vlans hr-comm interface ge-0/0/11.0
set vlans hr-comm interface ge-0/0/12.0
set vlans finance-comm interface ge-0/0/13.0
set vlans finance-comm interface ge-0/0/14.0
set vlans hr-comm primary-vlan pvlan100
set vlans finance-comm primary-vlan pvlan100
set pvlan100 interface ge-0/0/15.0 isolated
set pvlan100 interface ge-0/0/16.0 isolated
```

Procedure

Step-by-Step Procedure

To configure the PVLAN:

1. Set the VLAN ID for the primary VLAN:

```
[edit vlans]
user@switch# set pvlan vlan-id 100
```

2. Set the interfaces and port modes:

```
[edit interfaces]
user@switch# set ge-0/0/0 unit 0 family ethernet-switching port-mode trunk
user@switch# set ge-0/0/0 unit 0 family ethernet-switching vlan members pvlan
user@switch# set ge-1/0/0 unit 0 family ethernet-switching port-mode trunk
user@switch# set ge-1/0/0 unit 0 family ethernet-switching vlan members pvlan
user@switch# set ge-0/0/11 unit 0 family ethernet-switching port-mode access
user@switch# set ge-0/0/12 unit 0 family ethernet-switching port-mode access
user@switch# set ge-0/0/13 unit 0 family ethernet-switching port-mode access
user@switch# set ge-0/0/14 unit 0 family ethernet-switching port-mode access
user@switch# set ge-0/0/15 unit 0 family ethernet-switching port-mode access
user@switch# set ge-0/0/16 unit 0 family ethernet-switching port-mode access
```

3. Set the primary VLAN to have no local switching:

NOTE: The primary VLAN must be a tagged VLAN.

```
[edit vlans]
user@switch# set pvlan100 pvlan
```

4. Add the trunk interfaces to the primary VLAN:

```
[edit vlans]
user@switch# set pvlan100 interface ge-0/0/0.0
user@switch# set pvlan100 interface ge-1/0/0.0
```

5. For each secondary VLAN, configure access interfaces:

> **NOTE**: We recommend that the secondary VLANs be untagged VLANs. It does not impair functioning if you tag the secondary VLANS. However, the tags are not used when a secondary VLAN is configured on a single switch.

```
[edit vlans]
user@switch# set hr-comm interface ge-0/0/11.0
user@switch# set hr-comm interface ge-0/0/12.0
user@switch# set finance-comm interface ge-0/0/13.0
user@switch# set finance-comm interface ge-0/0/14.0
```

6. For each community VLAN, set the primary VLAN:

```
[edit vlans]
user@switch# set hr-comm primary-vlan pvlan100
user@switch# set finance-comm primary-vlan pvlan100
```

7. Configure the isolated interfaces in the primary VLAN:

```
[edit vlans]
user@switch# set pvlan100 interface ge-0/0/15.0 isolated
user@switch# set pvlan100 interface ge-0/0/16.0 isolated
```

Results

Check the results of the configuration:

```
[edit]
user@switch#  show
interfaces {
    ge-0/0/0 {
        unit 0 {
            family ethernet-switching {
                port-mode trunk;
                vlan {
                    members pvlan100;
```

```
                }
            }
        }
    }
    ge-1/0/0 {
        unit 0 {
            family ethernet-switching;
        }
    }
    ge-0/0/11 {
        unit 0 {
            family ethernet-switching {
                port-mode access;
            }
        }
    }
    ge-0/0/12 {
        unit 0 {
            family ethernet-switching {
                port-mode access;
            }
        }
    }
    ge-0/0/13 {
        unit 0 {
            family ethernet-switching {
                port-mode access;
            }
        }
    }
    ge-0/0/14 {
        unit 0 {
            family ethernet-switching {
                port-mode access;
            }
        }
    }
vlans {
    finance-comm {
    interface {
        ge-0/0/13.0;
        ge-0/0/14.0;
    }
```

```
    primary-vlan pvlan100;
    }
    hr-comm {
        interface {
            ge-0/0/11.0;
            ge-0/0/12.0;
        }
        primary-vlan pvlan100;
    }
    pvlan100 {
        vlan-id 100;
        interface {
            ge-0/0/15.0;
            ge-0/0/16.0;
            ge-0/0/0.0;
            ge-1/0/0.0;
        }
        pvlan;
    }
}
```

Verification

IN THIS SECTION

To confirm that the configuration is working properly, perform these tasks:

Verifying That the Private VLAN and Secondary VLANs Were Created

Purpose

Verify that the primary VLAN and secondary VLANs were properly created on the switch.

Action

Use the `show vlans` command:

```
user@switch> show vlans pvlan100 extensive
VLAN: pvlan100, Created at: Tue Sep 16 17:59:47 2008
802.1Q Tag: 100, Internal index: 18, Admin State: Enabled, Origin: Static
Private VLAN Mode: Primary
Protocol: Port Mode
Number of interfaces: Tagged 2 (Active = 0), Untagged  6 (Active = 0)
      ge-0/0/0.0, tagged, trunk
      ge-0/0/11.0, untagged, access
      ge-0/0/12.0, untagged, access
      ge-0/0/13.0, untagged, access
      ge-0/0/14.0, untagged, access
      ge-0/0/15.0, untagged, access
      ge-0/0/16.0, untagged, access
      ge-1/0/0.0, tagged, trunk
Secondary VLANs: Isolated 2, Community  2
  Isolated VLANs :
      __pvlan_pvlan_ge-0/0/15.0__
      __pvlan_pvlan_ge-0/0/16.0__
  Community VLANs :
      finance-comm
      hr-comm
user@switch> show vlans hr-comm extensive
VLAN: hr-comm, Created at: Tue Sep 16 17:59:47 2008
Internal index: 22, Admin State: Enabled, Origin: Static
Private VLAN Mode: Community, Primary VLAN: pvlan100
Protocol: Port Mode
Number of interfaces: Tagged 2 (Active = 0), Untagged  2 (Active = 0)
      ge-0/0/0.0, tagged, trunk
      ge-0/0/11.0, untagged, access
      ge-0/0/12.0, untagged, access
      ge-1/0/0.0, tagged, trunk
user@switch> show vlans finance-comm extensive
VLAN: finance-comm, Created at: Tue Sep 16 17:59:47 2008
Internal index: 21, Admin State: Enabled, Origin: Static
Private VLAN Mode: Community, Primary VLAN: pvlan100
Protocol: Port Mode
Number of interfaces: Tagged 2 (Active = 0), Untagged  2 (Active = 0)
      ge-0/0/0.0, tagged, trunk
```

```
      ge-0/0/13.0, untagged, access
      ge-0/0/14.0, untagged, access
      ge-1/0/0.0, tagged, trunk
user@switch> show vlans __pvlan_pvlan_ge-0/0/15.0__ extensive
VLAN: __pvlan_pvlan_ge-0/0/15.0__, Created at: Tue Sep 16 17:59:47 2008
Internal index: 19, Admin State: Enabled, Origin: Static
Private VLAN Mode: Isolated, Primary VLAN: pvlan100
Protocol: Port Mode
Number of interfaces: Tagged 2 (Active = 0), Untagged  1 (Active = 0)
      ge-0/0/0.0, tagged, trunk
      ge-0/0/15.0, untagged, access
      ge-1/0/0.0, tagged, trunk
user@switch> show vlans __pvlan_pvlan_ge-0/0/16.0__ extensive
VLAN: __pvlan_pvlan_ge-0/0/16.0__, Created at: Tue Sep 16 17:59:47 2008
Internal index: 20, Admin State: Enabled, Origin: Static
Private VLAN Mode: Isolated, Primary VLAN: pvlan100
Protocol: Port Mode
Number of interfaces: Tagged 2 (Active = 0), Untagged  1 (Active = 0)
      ge-0/0/0.0, tagged, trunk
      ge-0/0/16.0, untagged, access
      ge-1/0/0.0, tagged, trunk
```

Meaning

The output shows that the primary VLAN was created and identifies the interfaces and secondary VLANs associated with it.

Example: Configuring a Private VLAN on a Single EX Series Switch

IN THIS SECTION

For security reasons, it is often useful to restrict the flow of broadcast and unknown unicast traffic and to even limit the communication between known hosts. The private VLAN (PVLAN) feature on EX Series switches allows an administrator to split a broadcast domain into multiple isolated broadcast subdomains, essentially putting a VLAN inside a VLAN.

This example describes how to create a PVLAN on a single EX Series switch:

NOTE: Configuring a voice over IP (VoIP) VLAN on PVLAN interfaces is not supported.

Requirements

This example uses the following hardware and software components:

- One EX Series switch
- Junos OS Release 9.3 or later for EX Series switches

Before you begin configuring a PVLAN, make sure you have created and configured the necessary VLANs. See *Configuring VLANs for EX Series Switches*.

Overview and Topology

In a large office with multiple buildings and VLANs, you might need to isolate some workgroups or other endpoints for security reasons or to partition the broadcast domain. This configuration example shows a simple topology to illustrate how to create a PVLAN with one primary VLAN and two community VLANs, one for HR and one for finance, as well as two isolated ports—one for the mail server and the other for the backup server.

Table 82 on page 508 lists the settings for the example topology.

Table 82: Components of the Topology for Configuring a PVLAN

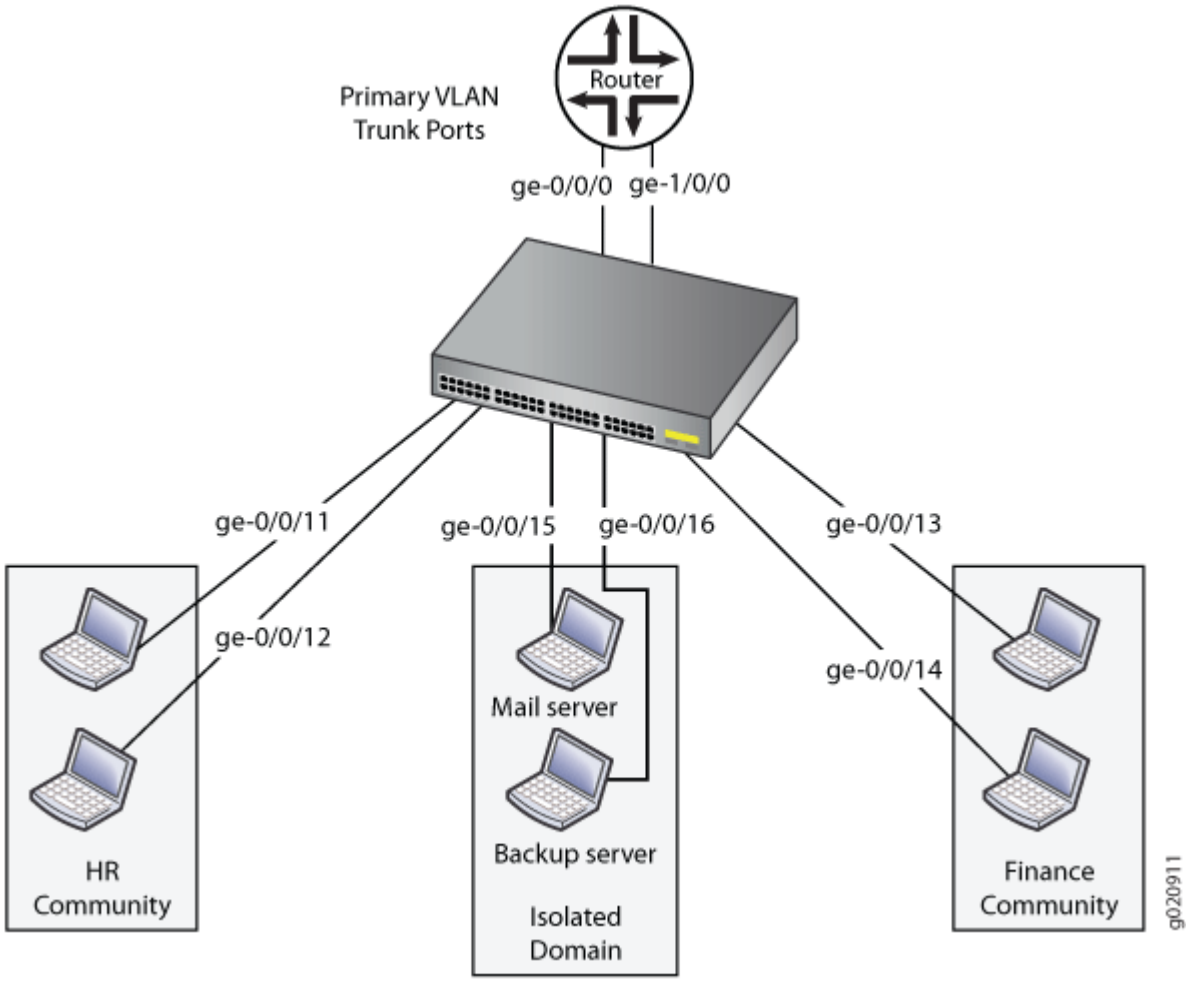
Interface	Description
ge-0/0/0.0	Primary VLAN (**vlan1**) trunk interface
ge-0/0/11.0	User 1, HR Community (**hr-comm**)
ge-0/0/12.0	User 2, HR Community (**hr-comm**)
ge-0/0/13.0	User 3, Finance Community (**finance-comm**)

Table 82: Components of the Topology for Configuring a PVLAN *(Continued)*

Interface	Description
ge-0/0/14.0	User 4, Finance Community (**finance-comm**)
ge-0/0/15.0	Mail server, Isolated (**isolated**)
ge-0/0/16.0	Backup server, Isolated (**isolated**)
ge-1/0/0.0	Primary VLAN (**pvlan**) trunk interface

Figure 24 on page 510 shows the topology for this example.

Figure 24: Topology of a Private VLAN on a Single EX Series Switch



Configuration

IN THIS SECTION

To configure a PVLAN, perform these tasks:

CLI Quick Configuration

To quickly create and configure a PVLAN, copy the following commands and paste them into the switch terminal window:

```
[edit]
set vlans vlan1 vlan-id 1000
set interfaces ge-0/0/0 unit 0 family ethernet-switching port-mode trunk
set interfaces ge-0/0/0 unit 0 family ethernet-switching vlan members vlan1
set interfaces ge-1/0/0 unit 0 family ethernet-switching port-mode trunk
set interfaces ge-1/0/0 unit 0 family ethernet-switching vlan members vlan1
set interfaces ge-0/0/11 unit 0 family ethernet-switching port-mode access
set interfaces ge-0/0/12 unit 0 family ethernet-switching port-mode access
set interfaces ge-0/0/13 unit 0 family ethernet-switching port-mode access
set interfaces ge-0/0/14 unit 0 family ethernet-switching port-mode access
set interfaces ge-0/0/15 unit 0 family ethernet-switching port-mode access
set interfaces ge-0/0/16 unit 0 family ethernet-switching port-mode access
set vlans vlan1 no-local-switching
set vlans vlan1 interface ge-0/0/0.0
set vlans vlan1 interface ge-1/0/0.0
set vlans hr-comm vlan-id 400
set vlans hr-comm interface ge-0/0/11.0
set vlans hr-comm interface ge-0/0/12.0
set vlans finance-comm vlan-id 300
set vlans finance-comm interface ge-0/0/13.0
set vlans finance-comm interface ge-0/0/14.0
set vlans hr-comm primary-vlan vlan1
set vlans finance-comm primary-vlan vlan1
set vlans vlan1 interface ge-0/0/15.0
set vlans vlan1 interface ge-0/0/16.0
```

Procedure

Step-by-Step Procedure

To configure the PVLAN:

1. Set the VLAN ID for the primary VLAN:

```
[edit vlans]
user@switch# set vlan1 vlan-id 1000
```

2. Set the interfaces and port modes:

```
[edit interfaces]
user@switch# set ge-0/0/0 unit 0 family ethernet-switching port-mode trunk
user@switch# set ge-0/0/0 unit 0 family ethernet-switching vlan members pvlan
user@switch# set ge-1/0/0 unit 0 family ethernet-switching port-mode trunk
user@switch# set ge-1/0/0 unit 0 family ethernet-switching vlan members vlan1
user@switch# set ge-0/0/11 unit 0 family ethernet-switching port-mode access
user@switch# set ge-0/0/12 unit 0 family ethernet-switching port-mode access
user@switch# set ge-0/0/13 unit 0 family ethernet-switching port-mode access
user@switch# set ge-0/0/14 unit 0 family ethernet-switching port-mode access
user@switch# set ge-0/0/15 unit 0 family ethernet-switching port-mode access
user@switch# set ge-0/0/16 unit 0 family ethernet-switching port-mode access
```

3. Set the primary VLAN to have no local switching:

NOTE: The primary VLAN must be a tagged VLAN.

```
[edit vlans]
user@switch# set vlan1 no-local-switching
```

4. Add the trunk interfaces to the primary VLAN:

```
[edit vlans]
user@switch# set vlan1 interface ge-0/0/0.0
user@switch# set vlan1 interface ge-1/0/0.0
```

5. For each secondary VLAN, configure the VLAN IDs and the access interfaces:

NOTE: We recommend that the secondary VLANs be untagged VLANs. It does not impair functioning if you tag the secondary VLANS. However, the tags are not used when a secondary VLAN is configured on a single switch.

```
 [edit vlans]
user@switch# set hr-comm vlan-id 400
user@switch# set hr-comm interface ge-0/0/11.0
 user@switch# set hr-comm interface ge-0/0/12.0
 user@switch# set finance-comm vlan-id 300
 user@switch# set finance-comm interface ge-0/0/13.0
 user@switch# set finance-comm interface ge-0/0/14.0
```

6. For each community VLAN, set the primary VLAN:

```
 [edit vlans]
user@switch# set hr-comm primary-vlan vlan1
 user@switch# set finance-comm primary-vlan vlan1
```

7. Add each isolated interface to the primary VLAN:

```
 [edit vlans]
user@switch# set vlan1 interface ge-0/0/15.0
 user@switch# set vlan1 interface ge-0/0/16.0
```

Results

Check the results of the configuration:

```
[edit]
user@switch#  show
interfaces {
    ge-0/0/0 {
        unit 0 {
            family ethernet-switching {
                port-mode trunk;
                vlan {
                    members vlan1;
```

```
                }
            }
        }
    }
    ge-1/0/0 {
        unit 0 {
            family ethernet-switching {
                port-mode trunk;
                vlan {
                    members vlan1;
                }
            }
        }
    }
    ge-0/0/11 {
        unit 0 {
            family ethernet-switching {
                port-mode access;
            }
        }
    }
    ge-0/0/12 {
        unit 0 {
            family ethernet-switching {
                port-mode access;
            }
        }
    }
    ge-0/0/13 {
        unit 0 {
            family ethernet-switching {
                port-mode access;
            }
        }
    }
    ge-0/0/14 {
        unit 0 {
            family ethernet-switching {
                port-mode access;
            }
        }
    }
vlans {
```

```
    finance-comm {
        vlan-id 300;
        interface {
            ge-0/0/13.0;
            ge-0/0/14.0;
        }
        primary-vlan vlan1;
    }
    hr-comm {
        vlan-id 400;
        interface {
            ge-0/0/11.0;
            ge-0/0/12.0;
        }
        primary-vlan vlan1;
    }
    vlan1 {
        vlan-id 1000;
        interface {
            ge-0/0/15.0;
            ge-0/0/16.0;
            ge-0/0/0.0;
            ge-1/0/0.0;
        }
        no-local-switching;
    }
}
```

Verification

IN THIS SECTION

To confirm that the configuration is working properly, perform these tasks:

Verifying That the Private VLAN and Secondary VLANs Were Created

Purpose

Verify that the primary VLAN and secondary VLANs were properly created on the switch.

Action

Use the `show vlans` command:

```
user@switch> show vlans vlan1 extensive
VLAN: vlan1, Created at: Tue Sep 16 17:59:47 2008
802.1Q Tag: 1000, Internal index: 18, Admin State: Enabled, Origin: Static
Private VLAN Mode: Primary
Protocol: Port Mode
Number of interfaces: Tagged 2 (Active = 0), Untagged  6 (Active = 0)
      ge-0/0/0.0, tagged, trunk
      ge-0/0/11.0, untagged, access
      ge-0/0/12.0, untagged, access
      ge-0/0/13.0, untagged, access
      ge-0/0/14.0, untagged, access
      ge-0/0/15.0, untagged, access
      ge-0/0/16.0, untagged, access
      ge-1/0/0.0, tagged, trunk
Secondary VLANs: Isolated 2, Community  2
  Isolated VLANs :
      __vlan1_vlan1_ge-0/0/15.0__
      __vlan1_vlan1_ge-0/0/16.0__
  Community VLANs :
      finance-comm
      hr-comm
user@switch> show vlans hr-comm extensive
VLAN: hr-comm, Created at: Tue Sep 16 17:59:47 2008
802.1Q Tag: 400,Internal index: 22, Admin State: Enabled, Origin: Static
Private VLAN Mode: Community, Primary VLAN: vlan1
Protocol: Port Mode
Number of interfaces: Tagged 2 (Active = 0), Untagged  2 (Active = 0)
      ge-0/0/0.0, tagged, trunk
      ge-0/0/11.0, untagged, access
      ge-0/0/12.0, untagged, access
      ge-1/0/0.0, tagged, trunk
user@switch> show vlans finance-comm extensive
```

```
VLAN: finance-comm, Created at: Tue Sep 16 17:59:47 2008
802.1Q Tag: 300,Internal index: 21, Admin State: Enabled, Origin: Static
Private VLAN Mode: Community, Primary VLAN: vlan1
Protocol: Port Mode
Number of interfaces: Tagged 2 (Active = 0), Untagged  2 (Active = 0)
      ge-0/0/0.0, tagged, trunk
      ge-0/0/13.0, untagged, access
      ge-0/0/14.0, untagged, access
      ge-1/0/0.0, tagged, trunk
user@switch> show vlans __vlan1_vlan1_ge-0/0/15.0__ extensive
VLAN: __vlan1_vlan1_ge-0/0/15.0__, Created at: Tue Sep 16 17:59:47 2008
Internal index: 19, Admin State: Enabled, Origin: Static
Private VLAN Mode: Isolated, Primary VLAN: vlan1
Protocol: Port Mode
Number of interfaces: Tagged 2 (Active = 0), Untagged  1 (Active = 0)
      ge-0/0/0.0, tagged, trunk
      ge-0/0/15.0, untagged, access
      ge-1/0/0.0, tagged, trunk
user@switch> show vlans __vlan1_vlan1_ge-0/0/16.0__ extensive
VLAN: __vlan1_vlan1_ge-0/0/16.0__, Created at: Tue Sep 16 17:59:47 2008
Internal index: 20, Admin State: Enabled, Origin: Static
Private VLAN Mode: Isolated, Primary VLAN: vlan1
Protocol: Port Mode
Number of interfaces: Tagged 2 (Active = 0), Untagged  1 (Active = 0)
      ge-0/0/0.0, tagged, trunk
      ge-0/0/16.0, untagged, access
      ge-1/0/0.0, tagged, trunk
```

Meaning

The output shows that the primary VLAN was created and identifies the interfaces and secondary VLANs associated with it.

Example: Configuring a Private VLAN Spanning Multiple QFX Switches

IN THIS SECTION

For security reasons, it is often useful to restrict the flow of broadcast and unknown unicast traffic and even to limit the communication between known hosts. The private VLAN (PVLAN) feature allows an administrator to split a broadcast domain into multiple isolated broadcast subdomains, essentially putting a VLAN inside a VLAN. A PVLAN can span multiple switches.

This example describes how to create a PVLAN spanning multiple switches. The example creates one primary PVLAN containing multiple secondary VLANs:

Requirements

This example uses the following hardware and software components:

- Three QFX3500 devices
- Junos OS Release 12.1 or later for the QFX Series

Before you begin configuring a PVLAN, make sure you have created and configured the necessary VLANs. See "Configuring VLANs on QFX Series Switches without Enhanced Layer 2 Support" on page 152.

Overview and Topology

IN THIS SECTION

In a large office with multiple buildings and VLANs, you might need to isolate some workgroups or other endpoints for security reasons or to partition the broadcast domain. This configuration example shows how to create a PVLAN spanning multiple QFX devices, with one primary VLAN containing two community VLANs (one for HR and one for Finance), and an interswitch isolated VLAN (for the mail server, the backup server, and the CVS server). The PVLAN comprises three switches, two access switches and one distribution switch. The PVLAN is connected to a router through a promiscuous port, which is configured on the distribution switch.

NOTE: The isolated ports on Switch 1 and on Switch 2 do not have Layer 2 connectivity with one another even though they are included within the same domain. See "Understanding Private VLANs" on page 428.

Figure 25 on page 520 shows the topology for this example—two access switches connecting to a distribution switch, which has a connection (through a promiscuous port) to the router.

Figure 25: PVLAN Topology Spanning Multiple Switches

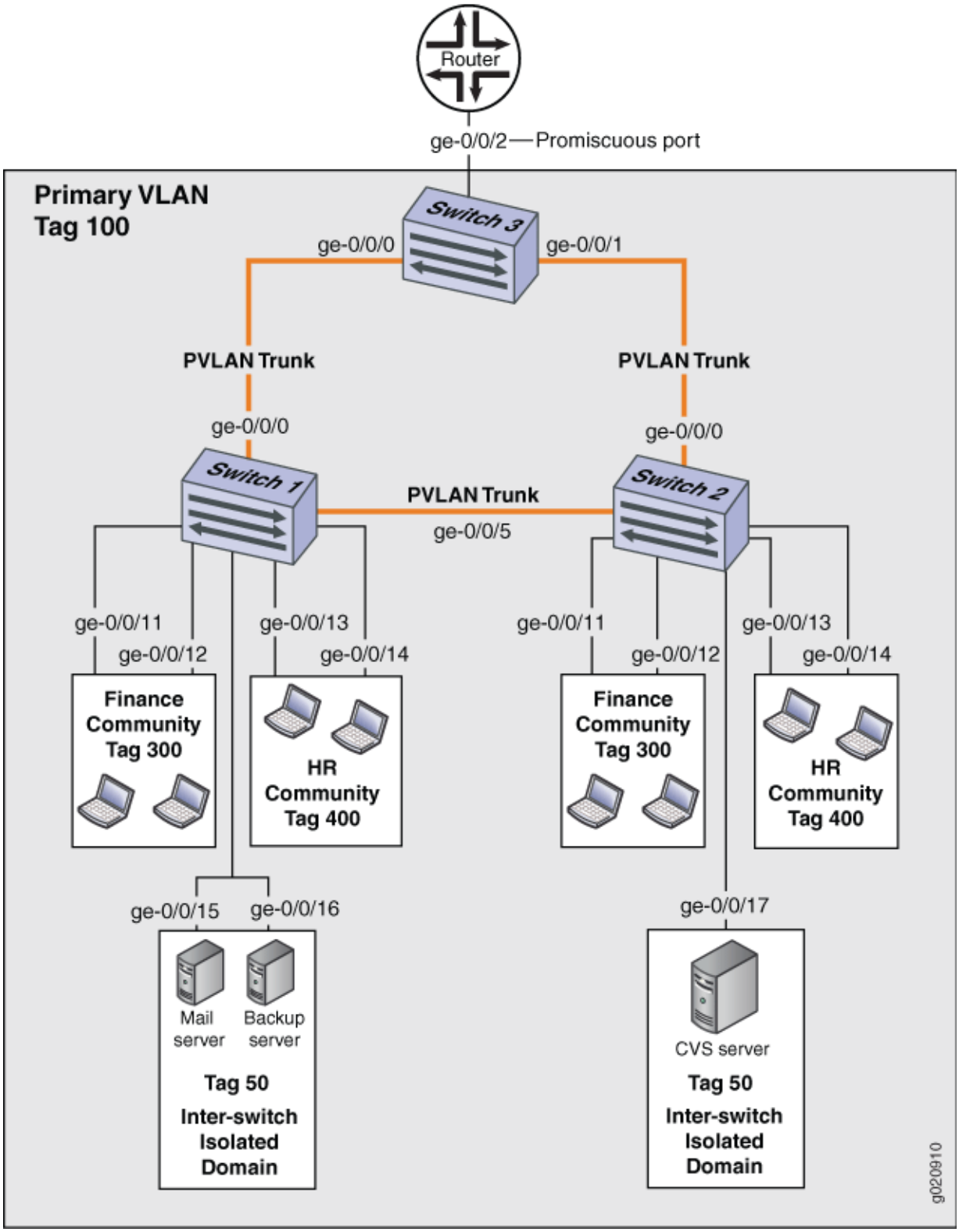


Table 83 on page 521, Table 84 on page 521, and Table 85 on page 522 list the settings for the example topology.

Table 83: Components of Switch 1 in the Topology for Configuring a PVLAN Spanning Multiple Devices

Property	Settings
VLAN names and tag IDs	**primary-vlan**, tag **100** **isolation-vlan-id**, tag **50** **finance-comm**, tag **300** **hr-comm**, tag **400**
PVLAN trunk interfaces	**ge-0/0/0.0**, connects Switch 1 to Switch 3 **ge-0/0/5.0**, connects Switch 1 to Switch 2
Isolated Interfaces in primary VLAN	**ge-0/0/15.0**, mail server **ge-0/0/16.0**, backup server
Interfaces in VLAN **finance-com**	**ge-0/0/11.0** **ge-0/0/12.0**
Interfaces in VLAN **hr-comm**	**ge-0/0/13.0** **ge-0/0/14.0**

Table 84: Components of Switch 2 in the Topology for Configuring a PVLAN Spanning Multiple Devices

Property	Settings
VLAN names and tag IDs	**primary-vlan**, tag **100** **isolation-vlan-id**, tag **50** **finance-comm**, tag **300** **hr-comm**, tag **400**
PVLAN trunk interfaces	**ge-0/0/0.0**, connects Switch 2 to Switch 3 **ge-0/0/5.0**, connects Switch 2 to Switch 1

Table 84: Components of Switch 2 in the Topology for Configuring a PVLAN Spanning Multiple Devices *(Continued)*

Property	Settings
Isolated Interface in primary VLAN	**ge-0/0/17.0**, CVS server
Interfaces in VLAN **finance-com**	**ge-0/0/11.0** **ge-0/0/12.0**
Interfaces in VLAN **hr-comm**	**ge-0/0/13.0** **ge-0/0/14.0**

Table 85: Components of Switch 3 in the Topology for Configuring a PVLAN Spanning Multiple Devices

Property	Settings
VLAN names and tag IDs	**primary-vlan**, tag **100** **isolation-vlan-id**, tag **50** **finance-comm**, tag **300** **hr-comm**, tag **400**
PVLAN trunk interfaces	**ge-0/0/0.0**, connects Switch 3 to Switch 1 **ge-0/0/1.0**, connects Switch 3 to Switch 2
Promiscuous port	**ge-0/0/2**, connects the PVLAN to the router **NOTE**: You must configure the trunk port that connects the PVLAN to another switch or router outside the PVLAN as a member of the PVLAN, which implicitly configures it as a promiscuous port.

Topology

Configuring a PVLAN on Switch 1

IN THIS SECTION

When configuring a PVLAN on multiple switches, these rules apply:

- The primary VLAN must be a tagged VLAN. We recommend that you configure the primary VLAN first.
- If you are going to configure a community VLAN ID, you must first configure the primary VLAN and the PVLAN trunk port. You must also configure the primary VLAN to be private using the *pvlan* statement.
- If you are going to configure an isolation VLAN ID, you must first configure the primary VLAN and the PVLAN trunk port.

CLI Quick Configuration

To quickly create and configure a PVLAN spanning multiple switches, copy the following commands and paste them into the terminal window of Switch 1:

```
[edit]
set vlans finance-comm vlan-id 300
set vlans finance-comm interface ge-0/0/11.0
set vlans finance-comm interface ge-0/0/12.0
set vlans finance-comm primary-vlan pvlan100
set vlans hr-comm vlan-id 400
set vlans hr-comm interface ge-0/0/13.0
set vlans hr-comm interface ge-0/0/14.0
set vlans hr-comm primary-vlan pvlan100
set vlans pvlan100 vlan-id 100
set vlans pvlan100  interface ge-0/0/15.0
set vlans pvlan100  interface ge-0/0/16.0
set vlans pvlan100 interface ge-0/0/0.0 pvlan-trunk
set vlans pvlan100 interface ge-0/0/5.0 pvlan-trunk
```

```
set vlans pvlan100 pvlan
set vlans pvlan100 pvlan isolation-vlan-id 50
set pvlan100 interface ge-0/0/15.0 isolated
set pvlan100 interface ge-0/0/16.0 isolated
```

Procedure

Step-by-Step Procedure

1. Set the VLAN ID for the primary VLAN:

```
[edit vlans]
user@switch# set pvlan100 vlan-id 100
```

2. Set the PVLAN trunk interfaces to connect this VLAN across neighboring switches:

```
[edit vlans]
user@switch# set pvlan100 interface ge-0/0/0.0 pvlan-trunk
user@switch# set pvlan100 interface ge-0/0/5.0 pvlan-trunk
```

3. Set the primary VLAN to be private and have no local switching:

```
 [edit vlans]
user@switch# set pvlan100 pvlan
```

4. Set the VLAN ID for the **finance-comm** community VLAN that spans the switches:

```
[edit vlans]
user@switch# set finance-comm vlan-id 300
```

5. Configure access interfaces for the **finance-comm** VLAN:

```
[edit vlans]
user@switch# set finance-comm interface ge-0/0/11.0
```

```
user@switch# set finance-comm interface ge-0/0/12.0
```

6. Set the primary VLAN of this secondary community VLAN, **finance-comm** :

```
[edit vlans]
user@switch# set vlans finance-comm primary-vlan pvlan100
```

7. Set the VLAN ID for the HR community VLAN that spans the switches.

```
[edit vlans]
user@switch# set hr-comm vlan-id 400
```

8. Configure access interfaces for the **hr-comm** VLAN:

```
[edit vlans]
user@switch# set hr-comm interface ge-0/0/13.0
user@switch# set hr-comm interface ge-0/0/14.0
```

9. Set the primary VLAN of this secondary community VLAN, **hr-comm**:

```
[edit vlans]
user@switch# set vlans hr-comm primary-vlan pvlan100
```

10. Set the interswitch isolated ID to create an interswitch isolated domain that spans the switches:

```
[edit vlans]
user@switch# set pvlan100 pvlan isolation-vlan-id 50
```

11. Configure the isolated interfaces in the primary VLAN:

```
[edit vlans]
user@switch# set pvlan100 interface ge-0/0/15.0 isolated
user@switch# set pvlan100 interface ge-0/0/16.0 isolated
```

NOTE: When you configure an isolated port, include it as a member of the primary VLAN, but do not configure it as a member of any community VLAN.

Results

Check the results of the configuration:

```
[edit]
user@switch# show
vlans {
    finance-comm {
        vlan-id 300;
        interface {
            ge-0/0/11.0;
            ge-0/0/12.0;
        }
        primary-vlan pvlan100;
    }
    hr-comm {
        vlan-id 400;
        interface {
            ge-0/0/13.0;
            ge-0/0/14.0;
        }
        primary-vlan pvlan100;
    }
    pvlan100 {
        vlan-id 100;
        interface {
            ge-0/0/15.0;
            ge-0/0/16.0;
            ge-0/0/0.0 {
                pvlan-trunk;
```

```
            }
            ge-0/0/5.0 {
                pvlan-trunk;
            }
        }
        pvlan;
        isolation-vlan-id 50;
    }
}
```

Configuring a PVLAN on Switch 2

IN THIS SECTION

CLI Quick Configuration

To quickly create and configure a private VLAN spanning multiple switches, copy the following commands and paste them into the terminal window of Switch 2:

NOTE: The configuration of Switch 2 is the same as the configuration of Switch 1 except for the interface in the interswitch isolated domain. For Switch 2, the interface is **ge-0/0/17.0**.

```
[edit]
set vlans finance-comm vlan-id 300
set vlans finance-comm interface ge-0/0/11.0
set vlans finance-comm interface ge-0/0/12.0
set vlans finance-comm primary-vlan pvlan100
set vlans hr-comm vlan-id 400
set vlans hr-comm interface ge-0/0/13.0
set vlans hr-comm interface ge-0/0/14.0
set vlans hr-comm primary-vlan pvlan100
set vlans pvlan100 vlan-id 100
```

```
set vlans pvlan100  interface ge-0/0/17.0
set vlans pvlan100 interface ge-0/0/0.0 pvlan-trunk
set vlans pvlan100 interface ge-0/0/5.0 pvlan-trunk
set vlans pvlan100 pvlan
set vlans pvlan100 pvlan isolation-vlan-id 50
set pvlan100 interface ge-0/0/17.0 isolated
```

Procedure

Step-by-Step Procedure

To configure a PVLAN on Switch 2 that will span multiple switches:

1. Set the VLAN ID for the **finance-comm** community VLAN that spans the switches:

```
[edit vlans]
user@switch# set finance-comm vlan-id 300
```

2. Configure access interfaces for the **finance-comm** VLAN:

```
 [edit vlans]
user@switch# set finance-comm interface ge-0/0/11.0
```

```
 user@switch# set finance-comm interface ge-0/0/12.0
```

3. Set the primary VLAN of this secondary community VLAN, **finance-comm**:

```
[edit vlans]
user@switch# set vlans finance-comm primary-vlan pvlan100
```

4. Set the VLAN ID for the HR community VLAN that spans the switches.

```
[edit vlans]
user@switch# set hr-comm vlan-id 400
```

5. Configure access interfaces for the **hr-comm** VLAN:

```
 [edit vlans]
user@switch# set hr-comm interface ge-0/0/13.0
 user@switch# set hr-comm interface ge-0/0/14.0
```

6. Set the primary VLAN of this secondary community VLAN, **hr-comm**:

```
[edit vlans]
user@switch# set vlans hr-comm primary-vlan pvlan100
```

7. Set the VLAN ID for the primary VLAN:

```
[edit vlans]
user@switch# set pvlan100 vlan-id 100
```

8. Set the PVLAN trunk interfaces that will connect this VLAN across neighboring switches:

```
[edit vlans]
user@switch# set pvlan100 interface ge-0/0/0.0 pvlan-trunk
user@switch# set  pvlan100 interface ge-0/0/5.0 pvlan-trunk
```

9. Set the primary VLAN to be private and have no local switching:

```
 [edit vlans]
user@switch# set pvlan100 pvlan
```

10. Set the interswitch isolated ID to create an interswitch isolated domain that spans the switches:

```
[edit vlans]
user@switch# set pvlan100 pvlan isolation-vlan-id 50
```

NOTE: To configure an isolated port, include it as one of the members of the primary VLAN, but do not configure it as belonging to one of the community VLANs.

11. Configure the isolated interface in the primary VLAN:

```
 [edit vlans]
user@switch# set pvlan100 interface ge-0/0/17.0 isolated
```

Results

Check the results of the configuration:

```
[edit]
user@switch# show
vlans {
    finance-comm {
        vlan-id 300;
        interface {
            ge-0/0/11.0;
            ge-0/0/12.0;
        }
        primary-vlan pvlan100;
    }
    hr-comm {
        vlan-id 400;
        interface {
            ge-0/0/13.0;
            ge-0/0/14.0;
        }
        primary-vlan pvlan100;
    }
    pvlan100 {
        vlan-id 100;
        interface {
            ge-0/0/15.0;
            ge-0/0/16.0;
            ge-0/0/0.0 {
                pvlan-trunk;
            }
            ge-0/0/5.0 {
                pvlan-trunk;
            }
            ge-0/0/17.0;
        }
```

```
        pvlan;
        isolation-vlan-id 50;
    }
}
```

Configuring a PVLAN on Switch 3

IN THIS SECTION

CLI Quick Configuration

To quickly configure Switch 3 to function as the distribution switch of this PVLAN, copy the following commands and paste them into the terminal window of Switch 3:

NOTE: Interface ge-0/0/2.0 is a trunk port connecting the PVLAN to a router.

```
[edit]
set vlans finance-comm vlan-id 300
set vlans finance-comm primary-vlan pvlan100
set vlans hr-comm vlan-id 400
set vlans hr-comm primary-vlan pvlan100
set vlans pvlan100 vlan-id 100
set vlans pvlan100 interface ge-0/0/0.0 pvlan-trunk
set vlans pvlan100 interface ge-0/0/1.0 pvlan-trunk
set vlans pvlan100 pvlan
set vlans pvlan100 pvlan isolation-vlan-id 50
```

Procedure

Step-by-Step Procedure

To configure Switch 3 to function as the distribution switch for this PVLAN, use the following procedure:

1. Set the VLAN ID for the **finance-comm** community VLAN that spans the switches:

```
[edit vlans]
user@switch# finance-comm vlan-id 300
```

2. Set the primary VLAN of this secondary community VLAN, **finance-comm**:

```
[edit vlans]
user@switch# set vlans finance-comm primary-vlan pvlan100
```

3. Set the VLAN ID for the HR community VLAN that spans the switches:

```
[edit vlans]
user@switch# set hr-comm vlan-id 400
```

4. Set the primary VLAN of this secondary community VLAN, **hr-comm**:

```
[edit vlans]
user@switch# set vlans hr-comm primary-vlan pvlan100
```

5. Set the VLAN ID for the primary VLAN:

```
[edit vlans]
user@switch# set pvlan100 vlan-id 100
```

6. Set the PVLAN trunk interfaces that will connect this VLAN across neighboring switches:

```
[edit vlans]
user@switch# set pvlan100 interface ge-0/0/0.0 pvlan-trunk
user@switch# set pvlan100 interface ge-0/0/5.0 pvlan-trunk
```

7. Set the primary VLAN to be private and have no local switching:

```
 [edit vlans]
user@switch# set pvlan100 pvlan
```

8. Set the interswitch isolated ID to create an interswitch isolated domain that spans the switches:

```
[edit vlans]
user@switch# set pvlan100 pvlan isolation-vlan-id 50
```

NOTE: To configure an isolated port, include it as one of the members of the primary VLAN, but do not configure it as belonging to one of the community VLANs.

Results

Check the results of the configuration:

```
[edit]
user@switch# show
vlans {
    finance-comm {
        vlan-id 300;
        primary-vlan pvlan100;
    }
    hr-comm {
        vlan-id 400;
        primary-vlan pvlan100;
    }
    pvlan100 {
        vlan-id 100;
        interface {
            ge-0/0/0.0 {
                pvlan-trunk;
            }
            ge-0/0/1.0 {
                pvlan-trunk;
            }
            ge-0/0/2.0;
        }
        pvlan;
        isolation-vlan-id 50;
    }
}
```

Verification

IN THIS SECTION

To confirm that the configuration is working properly, perform these tasks:

Verifying That the Primary VLAN and Secondary VLANs Were Created on Switch 1

Purpose

Verify that the PVLAN configuration spanning multiple switches is working properly on Switch 1:

Action

Use the `show vlans extensive` command:

```
user@switch> show vlans extensive
VLAN: __pvlan_pvlan100_ge-0/0/15.0__, Created at: Thu Sep 16 23:15:27 2010
Internal index: 5, Admin State: Enabled, Origin: Static
Private VLAN Mode: Isolated, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  1 (Active = 1)
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/5.0*, tagged, trunk, pvlan-trunk
      ge-0/0/15.0*, untagged, access

VLAN: __pvlan_pvlan100_ge-0/0/16.0__, Created at: Thu Sep 16 23:15:27 2010
Internal index: 6, Admin State: Enabled, Origin: Static
Private VLAN Mode: Isolated, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  1 (Active = 1)
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/5.0*, tagged, trunk, pvlan-trunk
      ge-0/0/16.0*, untagged, access
```

```
VLAN: __pvlan_pvlan100_isiv__, Created at: Thu Sep 16 23:15:27 2010
802.1Q Tag: 50, Internal index: 7, Admin State: Enabled, Origin: Static
Private VLAN Mode: Inter-switch-isolated, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  0 (Active = 0)
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/5.0*, tagged, trunk, pvlan-trunk

VLAN: default, Created at: Thu Sep 16 03:03:18 2010
Internal index: 2, Admin State: Enabled, Origin: Static
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 0 (Active = 0), Untagged  0 (Active = 0)

VLAN: finance-comm, Created at: Thu Sep 16 23:15:27 2010
802.1Q Tag: 300, Internal index: 8, Admin State: Enabled, Origin: Static
Private VLAN Mode: Community, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  2 (Active = 2)
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/5.0*, tagged, trunk, pvlan-trunk
      ge-0/0/11.0*, untagged, access
      ge-0/0/12.0*, untagged, access

VLAN: hr-comm, Created at: Thu Sep 16 23:15:27 2010
802.1Q Tag: 400, Internal index: 9, Admin State: Enabled, Origin: Static
Private VLAN Mode: Community, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  2 (Active = 2)
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/5.0*, tagged, trunk, pvlan-trunk
      ge-0/0/13.0*, untagged, access
      ge-0/0/14.0*, untagged, access

VLAN: pvlan100, Created at: Thu Sep 16 23:15:27 2010
802.1Q Tag: 100, Internal index: 4, Admin State: Enabled, Origin: Static
Private VLAN Mode: Primary
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  6 (Active = 6)
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/5.0*, tagged, trunk, pvlan-trunk
      ge-0/0/11.0*, untagged, access
      ge-0/0/12.0*, untagged, access
```

```
      ge-0/0/13.0*, untagged, access
      ge-0/0/14.0*, untagged, access
      ge-0/0/15.0*, untagged, access
      ge-0/0/16.0*, untagged, access
Secondary VLANs: Isolated 2, Community  2, Inter-switch-isolated  1
  Isolated VLANs :
      __pvlan_pvlan100_ge-0/0/15.0__
      __pvlan_pvlan100_ge-0/0/16.0__
  Community VLANs :
      finance-comm
      hr-comm
  Inter-switch-isolated VLAN :
      __pvlan_pvlan100_isiv__
```

Meaning

The output shows that a PVLAN was created on Switch 1 and shows that it includes two isolated VLANs, two community VLANs, and an interswitch isolated VLAN. The presence of the pvlan-trunk and Inter-switch-isolated fields indicates that this PVLAN is spanning more than one switch.

Verifying That the Primary VLAN and Secondary VLANs Were Created on Switch 2

Purpose

Verify that the PVLAN configuration spanning multiple switches is working properly on Switch 2:

Action

Use the `show vlans extensive` command:

```
user@switch> show vlans extensive
VLAN: __pvlan_pvlan100_ge-0/0/17.0__, Created at: Thu Sep 16 23:19:22 2010
Internal index: 5, Admin State: Enabled, Origin: Static
Private VLAN Mode: Isolated, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  1 (Active = 1)
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/5.0*, tagged, trunk, pvlan-trunk
      ge-0/0/17.0*, untagged, access

VLAN: __pvlan_pvlan100_isiv__, Created at: Thu Sep 16 23:19:22 2010
```

```
802.1Q Tag: 50, Internal index: 6, Admin State: Enabled, Origin: Static
Private VLAN Mode: Inter-switch-isolated, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  0 (Active = 0)
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/5.0*, tagged, trunk, pvlan-trunk

VLAN: default, Created at: Thu Sep 16 03:03:18 2010
Internal index: 2, Admin State: Enabled, Origin: Static
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 0 (Active = 0), Untagged  0 (Active = 0)

VLAN: finance-comm, Created at: Thu Sep 16 23:19:22 2010
802.1Q Tag: 300, Internal index: 7, Admin State: Enabled, Origin: Static
Private VLAN Mode: Community, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  2 (Active = 2)
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/5.0*, tagged, trunk, pvlan-trunk
      ge-0/0/11.0*, untagged, access
      ge-0/0/12.0*, untagged, access

VLAN: hr-comm, Created at: Thu Sep 16 23:19:22 2010
802.1Q Tag: 400, Internal index: 8, Admin State: Enabled, Origin: Static
Private VLAN Mode: Community, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  2 (Active = 2)
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/5.0*, tagged, trunk, pvlan-trunk
      ge-0/0/13.0*, untagged, access
      ge-0/0/14.0*, untagged, access

VLAN: pvlan100, Created at: Thu Sep 16 23:19:22 2010
802.1Q Tag: 100, Internal index: 4, Admin State: Enabled, Origin: Static
Private VLAN Mode: Primary
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  5 (Active = 5)
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/5.0*, tagged, trunk, pvlan-trunk
      ge-0/0/11.0*, untagged, access
      ge-0/0/12.0*, untagged, access
      ge-0/0/13.0*, untagged, access
      ge-0/0/14.0*, untagged, access
```

```
      ge-0/0/17.0*, untagged, access
Secondary VLANs: Isolated 1, Community  2, Inter-switch-isolated  1
  Isolated VLANs :
      __pvlan_pvlan100_ge-0/0/17.0__
  Community VLANs :
      finance-comm
      hr-comm
  Inter-switch-isolated VLAN :
      __pvlan_pvlan100_isiv__
```

Meaning

The output shows that a PVLAN was created on Switch 2 and shows that it includes one isolated VLAN, two community VLANs, and an interswitch isolated VLAN. The presence of the pvlan-trunk and Inter-switch-isolated fields indicates that this PVLAN is spanning more than one switch. When you compare this output to the output of Switch 1, you can see that both switches belong to the same PVLAN (**pvlan100**).

Verifying That the Primary VLAN and Secondary VLANs Were Created on Switch 3

Purpose

Verify that the PVLAN configuration spanning multiple switches is working properly on Switch 3:

Action

Use the `show vlans extensive` command:

```
user@switch> show vlans extensive
VLAN: __pvlan_pvlan100_isiv__, Created at: Thu Sep 16 23:22:40 2010
802.1Q Tag: 50, Internal index: 5, Admin State: Enabled, Origin: Static
Private VLAN Mode: Inter-switch-isolated, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  0 (Active = 0)
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/1.0*, tagged, trunk, pvlan-trunk

VLAN: default, Created at: Thu Sep 16 03:03:18 2010
Internal index: 2, Admin State: Enabled, Origin: Static
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 0 (Active = 0), Untagged  0 (Active = 0)
```

```
VLAN: finance-comm, Created at: Thu Sep 16 23:22:40 2010
802.1Q Tag: 300, Internal index: 6, Admin State: Enabled, Origin: Static
Private VLAN Mode: Community, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  0 (Active = 0)
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/1.0*, tagged, trunk, pvlan-trunk

VLAN: hr-comm, Created at: Thu Sep 16 23:22:40 2010
802.1Q Tag: 400, Internal index: 7, Admin State: Enabled, Origin: Static
Private VLAN Mode: Community, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  0 (Active = 0)
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/1.0*, tagged, trunk, pvlan-trunk

VLAN: pvlan100, Created at: Thu Sep 16 23:22:40 2010
802.1Q Tag: 100, Internal index: 4, Admin State: Enabled, Origin: Static
Private VLAN Mode: Primary
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  0 (Active = 0)
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/1.0*, tagged, trunk, pvlan-trunk
Secondary VLANs: Isolated 0, Community  2, Inter-switch-isolated  1
  Community VLANs :
      finance-comm
      hr-comm
  Inter-switch-isolated VLAN :
      __pvlan_pvlan100_isiv__
```

Meaning

The output shows that the PVLAN (**pvlan100**) is configured on Switch 3 and that it includes no isolated VLANs, two community VLANs, and an interswitch isolated VLAN. But Switch 3 is functioning as a distribution switch, so the output does not include access interfaces within the PVLAN. It shows only the **pvlan-trunk** interfaces that connect **pvlan100** from Switch 3 to the other switches (Switch 1 and Switch 2) in the same PVLAN.

Example: Configuring a Private VLAN Spanning Multiple Switches With an IRB Interface

IN THIS SECTION

For security reasons, it is often useful to restrict the flow of broadcast and unknown unicast traffic and even to limit the communication between known hosts. The private VLAN (PVLAN) feature allows an administrator to split a broadcast domain into multiple isolated broadcast subdomains, essentially putting a VLAN inside a VLAN. A PVLAN can span multiple switches. This example describes how to create a PVLAN spanning multiple switches. The example creates one primary PVLAN, containing multiple secondary VLANs.

Just like regular VLANs, PVLANs are isolated at Layer 2 and normally require that a Layer 3 device be used if you want to route traffic. Starting with Junos OS 14.1X53-D30, you can use an integrated routing and bridging (IRB) interface to route Layer 3 traffic between devices connected to a PVLAN. Using an IRB interface in this way can also allow the devices in the PVLAN to communicate at Layer 3 with devices in other community or isolated VLANs or with devices outside the PVLAN. This example also demonstrates how to include an IRB interface in a PVLAN configuration.

Requirements

This example uses the following hardware and software components:

- Three QFX Series or EX4600 switches
- Junos OS release with PVLAN for QFX Series or EX4600

Overview and Topology

IN THIS SECTION

In a large office with multiple buildings and VLANs, you might need to isolate some workgroups or other endpoints for security reasons or to partition the broadcast domain. This configuration example shows how to create a PVLAN spanning multiple switches, with one primary VLAN containing two community VLANs (one for HR and one for Finance), and an interswitch isolated VLAN (for the mail server, the backup server, and the CVS server). The PVLAN comprises three switches—two access switches and one distribution switch. The devices in the PVLAN are connected at Layer 3 to each other and to devices outside the PVLAN through an IRB interface configured on the distribution switch.

NOTE: The isolated ports on Switch 1 and on Switch 2 do not have Layer 2 connectivity with one another even though they are included within the same domain. See "Understanding Private VLANs" on page 428.

Figure 26 on page 542 shows the topology for this example.

Figure 26: PVLAN Topology Spanning Multiple Switches with an IRB Interface

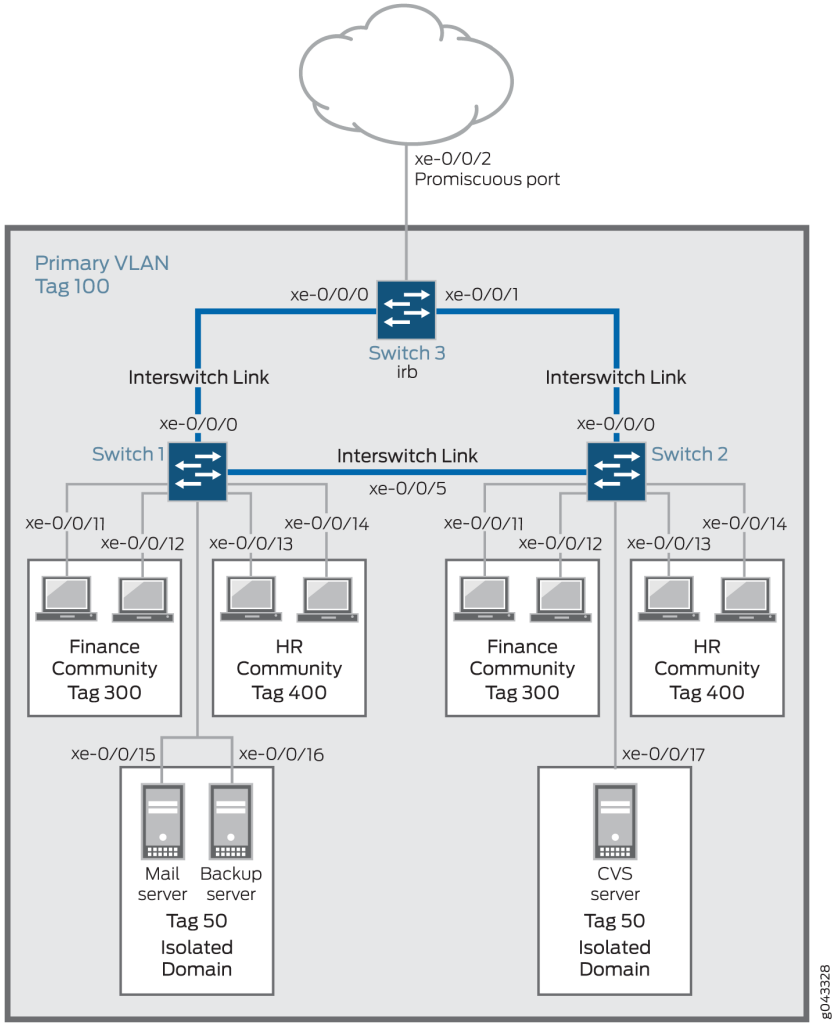


Table 86 on page 542, Table 87 on page 543, and Table 88 on page 544 list the settings for the example topology.

Table 86: Components of Switch 1 in the Topology for Configuring a PVLAN Spanning Multiple Devices

Property	Settings
VLAN names and tag IDs	`primary-vlan`, tag `100` `isolated-vlan-id`, tag `50` `finance-comm`, tag `300` `hr-comm`, tag `400`

Table 86: Components of Switch 1 in the Topology for Configuring a PVLAN Spanning Multiple Devices *(Continued)*

Property	Settings
Interswitch link interfaces	xe-0/0/0.0, connects Switch 1 to Switch 3 xe-0/0/5.0, connects Switch 1 to Switch 2
Isolated Interfaces in primary VLAN	xe-0/0/15.0, mail server xe-0/0/16.0, backup server
Interfaces in VLAN finance-com	xe-0/0/11.0 xe-0/0/12.0
Interfaces in VLAN hr-comm	xe-0/0/13.0 xe-0/0/14.0

Table 87: Components of Switch 2 in the Topology for Configuring a PVLAN Spanning Multiple Devices

Property	Settings
VLAN names and tag IDs	primary-vlan, tag 100 isolated-vlan-id, tag 50 finance-comm, tag 300 hr-comm, tag 400
Interswitch link interfaces	xe-0/0/0.0, connects Switch 2 to Switch 3 xe-0/0/5.0, connects Switch 2 to Switch 1
Isolated Interface in primary VLAN	xe-0/0/17.0, CVS server
Interfaces in VLAN finance-com	xe-0/0/11.0 xe-0/0/12.0

Table 87: Components of Switch 2 in the Topology for Configuring a PVLAN Spanning Multiple Devices *(Continued)*

Property	Settings
Interfaces in VLAN `hr-comm`	`xe-0/0/13.0` `xe-0/0/14.0`

Table 88: Components of Switch 3 in the Topology for Configuring a PVLAN Spanning Multiple Devices

Property	Settings
VLAN names and tag IDs	`primary-vlan`, tag `100` `isolated-vlan-id`, tag `50finance-comm`, tag `300` `hr-comm`, tag `400`
Interswitch link interfaces	`xe-0/0/0.0`, connects Switch 3 to Switch 1. `xe-0/0/1.0`, connects Switch 3 to Switch 2.
Promiscuous port	`xe-0/0/2`, connects the PVLAN to another network. **NOTE**: You must configure the trunk port that connects the PVLAN to another switch or router outside the PVLAN as a member of the PVLAN, which implicitly configures it as a promiscuous port.
IRB interface	`xe-0/0/0` `xe-0/0/1` Configure unrestricted proxy ARP on the IRB interface to allow ARP resolution to occur so that devices that use IPv4 can communicate at Layer 3. For IPv6 traffic, you must explicitly map an IRB address to the destination address to allow ARP resolution.

Topology

Configuration Overview

When configuring a PVLAN on multiple switches, the following rules apply:

- The primary VLAN must be a tagged VLAN.
- The primary VLAN is the only VLAN that can be a member of an interswitch link interface.

When configuring an IRB interface in a PVLAN, these rules apply:

- You can create only one IRB interface in a PVLAN, regardless of how many switches participate in the PVLAN.
- The IRB interface must be a member of the primary VLAN in the PVLAN.
- Each host device that you want to connect at Layer 3 must use an IP address of the IRB as its default gateway address.

Configuring a PVLAN on Switch 1

IN THIS SECTION

CLI Quick Configuration

To quickly create and configure a PVLAN spanning multiple switches, copy the following commands and paste them into the terminal window of Switch 1:

```
[edit]
set interfaces xe-0/0/0 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/0 unit 0 family ethernet-switching inter-switch-link
set interfaces xe-0/0/0 unit 0 family ethernet-switching vlan members 100
set interfaces xe-0/0/5 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/5 unit 0 family ethernet-switching inter-switch-link
set interfaces xe-0/0/5 unit 0 family ethernet-switching vlan members 100
set vlans finance-comm vlan-id 300 private-vlan community
set vlans hr-comm vlan-id 400 private-vlan community
set vlans isolated-vlan vlan-id 50 private-vlan isolated
set vlans pvlan100 vlan-id 100 community-vlans [300 400] isolated-vlan 50
set interfaces xe-0/0/11 unit 0 family ethernet-switching vlan members 300
set interfaces xe-0/0/12 unit 0 family ethernet-switching vlan members 300
```

```
set interfaces xe-0/0/13 unit 0 family ethernet-switching vlan members 400
set interfaces xe-0/0/14 unit 0 family ethernet-switching vlan members 400
set interfaces xe-0/0/15 unit 0 family ethernet-switching vlan members 50
set interfaces xe-0/0/16 unit 0 family ethernet-switching vlan members 50
```

Procedure

Step-by-Step Procedure

1. Configure interface xe-0/0/0 to be a trunk:

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

2. Configure interface xe-0/0/0 to be an interswitch link that carries all the VLANs:

```
[edit interfaces]
user@switch# set xe-0/0/0 unit 0 family ethernet-switching inter-switch-link
```

3. Configure pvlan100 (the primary VLAN) to be a member of interface xe-0/0/0:

```
[edit interfaces]
user@switch# set xe-0/0/0 unit 0 family ethernet-switching vlan members 100
```

4. Configure interface xe-0/0/5 to be a trunk:

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

5. Configure interface xe-0/0/5 to be an interswitch link that carries all the VLANs:

```
[edit interfaces]
user@switch# set xe-0/0/5 unit 0 family ethernet-switching inter-switch-link
```

6. Configure pvlan100 to be a member of interface xe-0/0/5:

```
[edit interfaces]
user@switch# set xe-0/0/5 unit 0 family ethernet-switching vlan members 100
```

7. Create the community VLAN for the finance organization:

```
[edit vlans]
set finance-comm vlan-id 300 private-vlan community
```

8. Create the community VLAN for the HR organization:

```
[edit vlans]
set hr-comm vlan-id 400 private-vlan community
```

9. Create the isolated VLAN for the mail and backup servers:

```
[edit vlans]
set isolated-vlan vlan-id 50 private-vlan isolated
```

10. Create the primary VLAN and make the community and isolated VLANs members of it:

```
[edit vlans]
set pvlan100 vlan-id 100 community-vlans [300 400] isolated-vlan 50
```

11. Configure VLAN 300 (the a community VLAN) to be a member of interface xe-0/0/11:

```
[edit interfaces]
user@switch# set xe-0/0/11 unit 0 family ethernet-switching vlan members 300
```

12. Configure VLAN 300 (a community VLAN) to be a member of interface xe-0/0/12:

```
[edit interfaces]
user@switch# set xe-0/0/12 unit 0 family ethernet-switching vlan members 300
```

13. Configure VLAN 400 (a community VLAN) to be a member of interface xe-0/0/13:

```
[edit interfaces]
user@switch# set xe-0/0/13 unit 0 family ethernet-switching vlan members 400
```

14. Configure VLAN 400 (a community VLAN) to be a member of interface xe-0/0/14:

```
[edit interfaces]
user@switch# set xe-0/0/14 unit 0 family ethernet-switching vlan members 400
```

15. Configure VLAN 50 (the isolated VLAN) to be a member of interface xe-0/0/15:

```
[edit interfaces]
user@switch# set xe-0/0/15 unit 0 family ethernet-switching vlan members 50
```

16. Configure VLAN 50 (the isolated VLAN) to be a member of interface xe-0/0/16:

```
[edit interfaces]
user@switch# set xe-0/0/16 unit 0 family ethernet-switching vlan members 50
```

Results

Check the results of the configuration:

```
[edit]
user@switch# show
vlans {
    finance-comm {
        vlan-id 300;
        private-vlan community;
    }
    hr-comm {
        vlan-id 400;
        private-vlan community;
    }
    isolated-vlan{
        vlan-id 50;
        private-vlan isolated;
```

```
    }
    pvlan100 {
        vlan-id 100;
        isolated-vlan 50;
        community-vlans [300 400]
    }
}
```

Configuring a PVLAN on Switch 2

IN THIS SECTION

CLI Quick Configuration

To quickly create and configure a private VLAN spanning multiple switches, copy the following commands and paste them into the terminal window of Switch 2:

NOTE: The configuration of Switch 2 is the same as the configuration of Switch 1 except for the isolated VLAN. For Switch 2, the isolated VLAN interface is `xe-0/0/17.0` .

```
[edit]
set interfaces xe-0/0/0 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/0 unit 0 family ethernet-switching inter-switch-link
set interfaces xe-0/0/0 unit 0 family ethernet-switching vlan members 100
set interfaces xe-0/0/5 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/5 unit 0 family ethernet-switching inter-switch-link
set interfaces xe-0/0/5 unit 0 family ethernet-switching vlan members 100
set vlans finance-comm vlan-id 300 private-vlan community
set vlans hr-comm vlan-id 400 private-vlan community
set vlans isolated-vlan vlan-id 50 private-vlan isolated
set vlans pvlan100 vlan-id 100 community-vlans [300 400] isolated-vlan 50
set interfaces xe-0/0/11 unit 0 family ethernet-switching vlan members 300
set interfaces xe-0/0/12 unit 0 family ethernet-switching vlan members 300
```

```
set interfaces xe-0/0/13 unit 0 family ethernet-switching vlan members 400
set interfaces xe-0/0/14 unit 0 family ethernet-switching vlan members 400
set interfaces xe-0/0/17 unit 0 family ethernet-switching vlan members 50
```

Procedure

Step-by-Step Procedure

1. Configure interface xe-0/0/0 to be a trunk:

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

2. Configure interface xe-0/0/0 to be an interswitch link that carries all the VLANs:

```
[edit interfaces]
user@switch# set xe-0/0/0 unit 0 family ethernet-switching inter-switch-link
```

3. Configure pvlan100 (the primary VLAN) to be a member of interface xe-0/0/0:

```
[edit interfaces]
user@switch# set xe-0/0/0 unit 0 family ethernet-switching vlan members 100
```

4. Configure interface xe-0/0/5 to be a trunk:

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

5. Configure interface xe-0/0/5 to be an interswitch link that carries all the VLANs:

```
[edit interfaces]
user@switch# set xe-0/0/5 unit 0 family ethernet-switching inter-switch-link
```

6. Configure pvlan100 to be a member of interface xe-0/0/5:

```
[edit interfaces]
user@switch# set xe-0/0/5 unit 0 family ethernet-switching vlan members 100
```

7. Create the community VLAN for the finance organization:

```
[edit vlans]
set finance-comm vlan-id 300 private-vlan community
```

8. Create the community VLAN for the HR organization:

```
[edit vlans]
set hr-comm vlan-id 400 private-vlan community
```

9. Create the isolated VLAN for the mail and backup servers:

```
[edit vlans]
set isolated-vlan vlan-id 50 private-vlan isolated
```

10. Create the primary VLAN and make the community and isolated VLANs members of it:

```
[edit vlans]
set pvlan100 vlan-id 100 community-vlans [300 400] isolated-vlan 50
```

11. Configure VLAN 300 (the a community VLAN) to be a member of interface xe-0/0/11:

```
[edit interfaces]
user@switch# set xe-0/0/11 unit 0 family ethernet-switching vlan members 300
```

12. Configure VLAN 300 (a community VLAN) to be a member of interface xe-0/0/12:

```
[edit interfaces]
user@switch# set xe-0/0/12 unit 0 family ethernet-switching vlan members 300
```

13. Configure VLAN 400 (a community VLAN) to be a member of interface xe-0/0/13:

```
[edit interfaces]
user@switch# set xe-0/0/13 unit 0 family ethernet-switching vlan members 400
```

14. Configure VLAN 400 (a community VLAN) to be a member of interface xe-0/0/14:

```
[edit interfaces]
user@switch# set xe-0/0/14 unit 0 family ethernet-switching vlan members 400
```

15. Configure VLAN 50 (the isolated VLAN) to be a member of interface xe-0/0/17:

```
[edit interfaces]
user@switch# set xe-0/0/17 unit 0 family ethernet-switching vlan members 50
```

Results

Check the results of the configuration:

```
[edit]
user@switch# show
vlans {
    finance-comm {
        vlan-id 300;
        private-vlan community;
    }
    hr-comm {
        vlan-id 400;
        private-vlan community;
    }
    isolated-vlan{
        vlan-id 50;
        private-vlan isolated;
    }
    pvlan100 {
        vlan-id 100;
        isolated-vlan 50;
        community-vlans [300 400]
```

```
    }
}
```

Configuring a PVLAN on Switch 3

IN THIS SECTION

CLI Quick Configuration

To quickly configure Switch 3 to function as the distribution switch of this PVLAN, copy the following commands and paste them into the terminal window of Switch 3:

NOTE: Interface xe-0/0/2.0 is a trunk port connecting the PVLAN to another network.

```
[edit]
[edit]
set interfaces xe-0/0/0 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/0 unit 0 family ethernet-switching inter-switch-link
set interfaces xe-0/0/0 unit 0 family ethernet-switching vlan members 100
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 inter-switch-link
set interfaces xe-0/0/1 unit 0 family ethernet-switching vlan members 100
set interfaces xe-0/0/2 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/2 unit 0 family ethernet-switching vlan members 100
set vlans pvlan100 vlan-id 100
set interfaces irb unit 100 family inet address 192.168.1.1/24
set vlans pvlan100 l3-interface irb.100
set interfaces irb unit 100 proxy-arp unrestricted
```

Procedure

Step-by-Step Procedure

To configure Switch 3 to function as the distribution switch for this PVLAN, use the following procedure:

1. Configure interface xe-0/0/0 to be a trunk:

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

2. Configure interface xe-0/0/0 to be an interswitch link that carries all the VLANs:

```
[edit interfaces]
user@switch# set xe-0/0/0 unit 0 family ethernet-switching inter-switch-link
```

3. Configure pvlan100 (the primary VLAN) to be a member of interface xe-0/0/0:

```
[edit interfaces]
user@switch# set xe-0/0/0 unit 0 family ethernet-switching vlan members 100
```

4. Configure interface xe-0/0/5 to be a trunk:

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

5. Configure interface xe-0/0/5 to be an interswitch link that carries all the VLANs:

```
[edit interfaces]
user@switch# set xe-0/0/5 unit 0 family ethernet-switching inter-switch-link
```

6. Configure pvlan100 to be a member of interface xe-0/0/5:

```
[edit interfaces]
user@switch# set xe-0/0/5 unit 0 family ethernet-switching vlan members 100
```

7. Configure interface xe-0/0/2 (the promiscuous interface) to be a trunk:

```
[edit interfaces]
user@switch# set xe-0/0/2 unit 0 family ethernet-switching interface-mode trunk
```

8. Configure pvlan100 to be a member of interface xe-0/0/2:

```
[edit interfaces]
user@switch# set xe-0/0/2 unit 0 family ethernet-switching vlan members 100
```

9. Create the primary VLAN:

```
[edit vlans]
set vlans pvlan100 vlan-id 100
```

10. Create the IRB interface `irb` and assign it an address in the subnet used by the devices attached to Switches 1 and 2:

```
[edit interfaces]
set irb unit 100 family inet address 192.168.1.1/24
```

NOTE: Each host device that you want to connect at Layer 3 must be in the same subnet as the IRB interface and use the IP address of the IRB interface as its default gateway address.

11. Complete the IRB interface configuration by binding the interface to the primary VLAN `pvlan100`:

```
[edit vlans]
set pvlan100 l3-interface irb.100
```

12. Configure unrestricted proxy ARP for each unit of the IRB interface so that ARP resolution works for IPv4 traffic:

```
[edit interfaces]
set irb unit 100 proxy-arp unrestricted
```

NOTE: Because the devices in the community and isolated VLANs are isolated at Layer 2, this step is required to allow ARP resolution to occur between the VLANs so that devices using IPv4 can communicate at Layer 3. (For IPv6 traffic, you must explicitly map an IRB address to the destination address to allow ARP resolution.)

Results

Check the results of the configuration:

```
[edit]
user@switch# show
vlans {
    pvlan100{
        vlan-id 100;
    }
}
```

Verification

IN THIS SECTION

To confirm that the configuration is working properly, perform these tasks:

Verifying That the Primary VLAN and Secondary VLANs Were Created on Switch 1

Purpose

Verify that the PVLAN configuration spanning multiple switches is working properly on Switch 1:

Action

Use the `show vlans extensive` command:

```
user@switch> show vlans extensive
VLAN: __pvlan_pvlan100_xe-0/0/15.0__, Created at: Wed Sep 16 23:15:27 2015
Internal index: 5, Admin State: Enabled, Origin: Static
Private VLAN Mode: Isolated, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  1 (Active = 1)
      xe-0/0/0.0*, tagged, trunk      xe-0/0/5.0*, tagged, trunk      xe-0/0/15.0*, untagged,
access

VLAN: __pvlan_pvlan100_xe-0/0/16.0__, Created at: Wed Sep 16 23:15:27 2015
Internal index: 6, Admin State: Enabled, Origin: Static
Private VLAN Mode: Isolated, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  1 (Active = 1)
      xe-0/0/0.0*, tagged, trunk      xe-0/0/5.0*, tagged, trunk      xe-0/0/16.0*, untagged,
access

VLAN: __pvlan_pvlan100_isiv__, Created at: Wed Sep 16 23:15:27 2015
802.1Q Tag: 50, Internal index: 7, Admin State: Enabled, Origin: Static
Private VLAN Mode: Inter-switch-isolated, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  0 (Active = 0)
      xe-0/0/0.0*, tagged, trunk      xe-0/0/5.0*, tagged, trunk
VLAN: default, Created at: Wed Sep 16 03:03:18 2015
Internal index: 2, Admin State: Enabled, Origin: Static
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 0 (Active = 0), Untagged  0 (Active = 0)

VLAN: finance-comm, Created at: Wed Sep 16 23:15:27 2015
802.1Q Tag: 300, Internal index: 8, Admin State: Enabled, Origin: Static
Private VLAN Mode: Community, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  2 (Active = 2)
      xe-0/0/0.0*, tagged, trunk      xe-0/0/5.0*, tagged, trunk      xe-0/0/11.0*, untagged,
access
      xe-0/0/12.0*, untagged, access

VLAN: hr-comm, Created at: Wed Sep 16 23:15:27 2015
```

```
802.1Q Tag: 400, Internal index: 9, Admin State: Enabled, Origin: Static
Private VLAN Mode: Community, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  2 (Active = 2)
      xe-0/0/0.0*, tagged, trunk      xe-0/0/5.0*, tagged, trunk      xe-0/0/13.0*, untagged,
access
      xe-0/0/14.0*, untagged, access

VLAN: pvlan100, Created at: Wed Sep 16 23:15:27 2015
802.1Q Tag: 100, Internal index: 4, Admin State: Enabled, Origin: Static
Private VLAN Mode: Primary
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  6 (Active = 6)
      xe-0/0/0.0*, tagged, trunk
      xe-0/0/5.0*, tagged, trunk      xe-0/0/11.0*, untagged, access
      xe-0/0/12.0*, untagged, access
      xe-0/0/13.0*, untagged, access
      xe-0/0/14.0*, untagged, access
      xe-0/0/15.0*, untagged, access
      xe-0/0/16.0*, untagged, access
Secondary VLANs: Isolated 2, Community  2, Inter-switch-isolated  1
  Isolated VLANs :
      __pvlan_pvlan100_xe-0/0/15.0__
      __pvlan_pvlan100_xe-0/0/16.0__
  Community VLANs :
      finance-comm
      hr-comm
  Inter-switch-isolated VLAN :
      __pvlan_pvlan100_isiv__
```

Meaning

The output shows that a PVLAN was created on Switch 1 and shows that it includes two isolated VLANs, two community VLANs, and an interswitch isolated VLAN. The presence of the trunk and Inter-switch-isolated fields indicates that this PVLAN is spanning more than one switch.

Verifying That the Primary VLAN and Secondary VLANs Were Created on Switch 2

Purpose

Verify that the PVLAN configuration spanning multiple switches is working properly on Switch 2:

Action

Use the `show vlans extensive` command:

```
user@switch> show vlans extensive
VLAN: __pvlan_pvlan100_xe-0/0/17.0__, Created at: Wed Sep 16 23:19:22 2015
Internal index: 5, Admin State: Enabled, Origin: Static
Private VLAN Mode: Isolated, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  1 (Active = 1)
      xe-0/0/0.0*, tagged, trunk
      xe-0/0/5.0*, tagged, trunk
      xe-0/0/17.0*, untagged, access

VLAN: __pvlan_pvlan100_isiv__, Created at: Wed Sep 16 23:19:22 2015
802.1Q Tag: 50, Internal index: 6, Admin State: Enabled, Origin: Static
Private VLAN Mode: Inter-switch-isolated, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  0 (Active = 0)
      xe-0/0/0.0*, tagged, trunk
      xe-0/0/5.0*, tagged, trunk

VLAN: default, Created at: Wed Sep 16 03:03:18 2015
Internal index: 2, Admin State: Enabled, Origin: Static
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 0 (Active = 0), Untagged  0 (Active = 0)

VLAN: finance-comm, Created at: Wed Sep 16 23:19:22 2015
802.1Q Tag: 300, Internal index: 7, Admin State: Enabled, Origin: Static
Private VLAN Mode: Community, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  2 (Active = 2)
      xe-0/0/0.0*, tagged, trunk
      xe-0/0/5.0*, tagged, trunk
      xe-0/0/11.0*, untagged, access
      xe-0/0/12.0*, untagged, access

VLAN: hr-comm, Created at: Wed Sep 16 23:19:22 2015
802.1Q Tag: 400, Internal index: 8, Admin State: Enabled, Origin: Static
Private VLAN Mode: Community, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  2 (Active = 2)
```

```
      xe-0/0/0.0*, tagged, trunk
      xe-0/0/5.0*, tagged, trunk
      xe-0/0/13.0*, untagged, access
      xe-0/0/14.0*, untagged, access

VLAN: pvlan100, Created at: Wed Sep 16 23:19:22 2015
802.1Q Tag: 100, Internal index: 4, Admin State: Enabled, Origin: Static
Private VLAN Mode: Primary
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  5 (Active = 5)
      xe-0/0/0.0*, tagged, trunk
      xe-0/0/5.0*, tagged, trunk
      xe-0/0/11.0*, untagged, access
      xe-0/0/12.0*, untagged, access
      xe-0/0/13.0*, untagged, access
      xe-0/0/14.0*, untagged, access
      xe-0/0/17.0*, untagged, access
Secondary VLANs: Isolated 1, Community  2, Inter-switch-isolated  1
  Isolated VLANs :
      __pvlan_pvlan100_xe-0/0/17.0__
  Community VLANs :
      finance-comm
      hr-comm
  Inter-switch-isolated VLAN :
      __pvlan_pvlan100_isiv__
```

Meaning

The output shows that a PVLAN was created on Switch 2 and shows that it includes one isolated VLAN, two community VLANs, and an interswitch isolated VLAN. The presence of the trunk and Inter-switch-isolated fields indicates that this PVLAN is spanning more than one switch. When you compare this output to the output of Switch 1, you can see that both switches belong to the same PVLAN (`pvlan100`).

Verifying That the Primary VLAN and Secondary VLANs Were Created on Switch 3

Purpose

Verify that the PVLAN configuration spanning multiple switches is working properly on Switch 3:

Action

Use the `show vlans extensive` command:

```
user@switch> show vlans extensive
VLAN: __pvlan_pvlan100_isiv__, Created at: Wed Sep 16 23:22:40 2015
802.1Q Tag: 50, Internal index: 5, Admin State: Enabled, Origin: Static
Private VLAN Mode: Inter-switch-isolated, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  0 (Active = 0)
      xe-0/0/0.0*, tagged, trunk
      xe-0/0/1.0*, tagged, trunk

VLAN: default, Created at: Wed Sep 16 03:03:18 2015
Internal index: 2, Admin State: Enabled, Origin: Static
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 0 (Active = 0), Untagged  0 (Active = 0)

VLAN: finance-comm, Created at: Wed Sep 16 23:22:40 2015
802.1Q Tag: 300, Internal index: 6, Admin State: Enabled, Origin: Static
Private VLAN Mode: Community, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  0 (Active = 0)
      xe-0/0/0.0*, tagged, trunk
      xe-0/0/1.0*, tagged, trunk

VLAN: hr-comm, Created at: Wed Sep 16 23:22:40 2015
802.1Q Tag: 400, Internal index: 7, Admin State: Enabled, Origin: Static
Private VLAN Mode: Community, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  0 (Active = 0)
      xe-0/0/0.0*, tagged, trunk
      xe-0/0/1.0*, tagged, trunk

VLAN: pvlan100, Created at: Wed Sep 16 23:22:40 2015
802.1Q Tag: 100, Internal index: 4, Admin State: Enabled, Origin: Static
Private VLAN Mode: Primary
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  0 (Active = 0)
      xe-0/0/0.0*, tagged, trunk
      xe-0/0/1.0*, tagged, trunk
Secondary VLANs: Isolated 0, Community  2, Inter-switch-isolated  1
```

```
Community VLANs :
    finance-comm
    hr-comm
Inter-switch-isolated VLAN :
    __pvlan_pvlan100_isiv__
```

Meaning

The output shows that the PVLAN (`pvlan100`) is configured on Switch 3 and that it includes no isolated VLANs, two community VLANs, and an interswitch isolated VLAN. But Switch 3 is functioning as a distribution switch, so the output does not include access interfaces within the PVLAN. It shows only the trunk interfaces that connect `pvlan100` from Switch 3 to the other switches (Switch 1 and Switch 2) in the same PVLAN.

Example: Configuring a Private VLAN Spanning Multiple EX Series Switches

IN THIS SECTION

For security reasons, it is often useful to restrict the flow of broadcast and unknown unicast traffic and to even limit the communication between known hosts. The private VLAN (PVLAN) feature on EX Series switches allows an administrator to split a broadcast domain into multiple isolated broadcast subdomains, essentially putting a VLAN inside a VLAN. A PVLAN can span multiple switches.

This example describes how to create a PVLAN spanning multiple EX Series switches. The example creates one primary PVLAN, containing multiple secondary VLANs:

NOTE: Configuring a voice over IP (VoIP) VLAN on PVLAN interfaces is not supported.

Requirements

This example uses the following hardware and software components:

- Three EX Series switches
- Junos OS Release 10.4 or later for EX Series switches

Before you begin configuring a PVLAN, make sure you have created and configured the necessary VLANs. See *Configuring VLANs for EX Series Switches*.

Overview and Topology

IN THIS SECTION

In a large office with multiple buildings and VLANs, you might need to isolate some workgroups or other endpoints for security reasons or to partition the broadcast domain. This configuration example shows how to create a PVLAN spanning multiple EX Series switches, with one primary VLAN containing two community VLANs (one for HR and one for Finance), and an Interswitch isolated VLAN (for the mail server, the backup server, and the CVS server). The PVLAN comprises three switches, two access switches and one distribution switch. The PVLAN is connected to a router through a promiscuous port, which is configured on the distribution switch.

NOTE: The isolated ports on Switch 1 and on Switch 2 do not have Layer 2 connectivity with each other even though they are included within the same domain. See "Understanding Private VLANs" on page 428.

Figure 27 on page 564 shows the topology for this example—two access switches connecting to a distribution switch, which has a connection (through a promiscuous port) to the router.

Figure 27: PVLAN Topology Spanning Multiple Switches

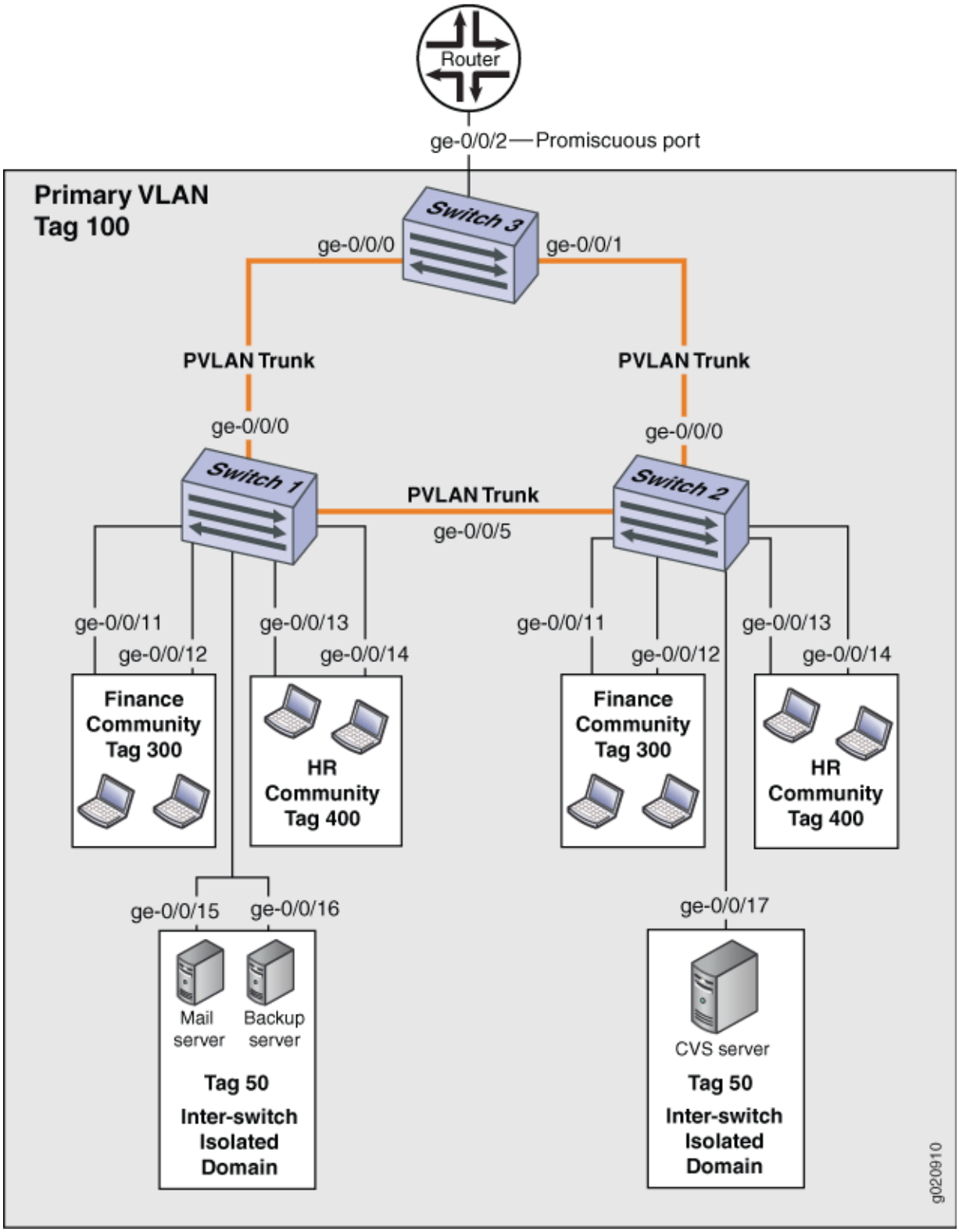


Table 89 on page 565, Table 90 on page 565, and Table 91 on page 566 list the settings for the example topology.

Table 89: Components of Switch 1 in the Topology for Configuring a PVLAN Spanning Multiple EX Series Switches

Property	Settings
VLAN names and tag IDs	**primary-vlan**, tag **100** **isolation-id**, tag **50** **finance-comm**, tag **300** **hr-comm**, tag **400**
PVLAN trunk interfaces	**ge-0/0/0.0**, Connects Switch 1 to Switch 3 **ge-0/0/5.0**, Connects Switch 1 to Switch 2
Interfaces in VLAN **isolation**	**ge-0/0/15.0**, Mail server **ge-0/0/16.0**, Backup server
Interfaces in VLAN **finance-com**	**ge-0/0/11.0** **ge-0/0/12.0**
Interfaces in VLAN **hr-comm**	**ge-0/0/13.0** **ge-0/0/14.0**

Table 90: Components of Switch 2 in the Topology for Configuring a PVLAN Spanning Multiple EX Series Switches

Property	Settings
VLAN names and tag IDs	**primary-vlan**, tag **100** **isolation-id**, tag **50** **finance-comm**, tag **300** **hr-comm**, tag **400**

Table 90: Components of Switch 2 in the Topology for Configuring a PVLAN Spanning Multiple EX Series Switches *(Continued)*

Property	Settings
PVLAN trunk interfaces	**ge-0/0/0.0**, Connects Switch 2 to Switch 3 **ge-0/0/5.0**, Connects Switch 2 to Switch 1
Interfaces in VLAN **isolation**	**ge-0/0/17.0**,CVS server
Interfaces in VLAN **finance-com**	**ge-0/0/11.0** **ge-0/0/12.0**
Interfaces in VLAN **hr-comm**	**ge-0/0/13.0** **ge-0/0/14.0**

Table 91: Components of Switch 3 in the Topology for Configuring a PVLAN Spanning Multiple EX Series Switches

Property	Settings
VLAN names and tag IDs	**primary-vlan**, tag **100** **isolation-id**, tag **50** **finance-comm**, tag **300** **hr-comm**, tag **400**
PVLAN trunk interfaces	**ge-0/0/0.0**, Connects Switch 3 to Switch 1 **ge-0/0/1.0**, Connects Switch 3 to Switch 2
Promiscuous port	**ge-0/0/2**, Connects the PVLAN to the router **NOTE**: You must configure the trunk port that connects the PVLAN to another switch or router outside the PVLAN as a member of the PVLAN, which implicitly configures it as a promiscuous port.

Topology

Configuring a PVLAN on Switch 1

IN THIS SECTION

CLI Quick Configuration

When configuring a PVLAN on multiple switches, these rules apply:

- The primary VLAN must be a tagged VLAN. We recommend that you configure the primary VLAN first.
- Configuring a voice over IP (VoIP) VLAN on PVLAN interfaces is not supported.
- If you are going to configure a community VLAN ID, you must first configure the primary VLAN and the PVLAN trunk port.
- If you are going to configure an isolation VLAN ID, you must first configure the primary VLAN and the PVLAN trunk port.
- Secondary VLANs and the PVLAN trunk port must be committed on a single commit if MVRP is configured on the PVLAN trunk port.

To quickly create and configure a PVLAN spanning multiple switches, copy the following commands and paste them into the terminal window of Switch 1:

```
[edit]
set vlans finance-comm vlan-id 300
set vlans finance-comm interface ge-0/0/11.0
set vlans finance-comm interface ge-0/0/12.0
set vlans finance-comm primary-vlan pvlan100
set vlans hr-comm vlan-id 400
set vlans hr-comm interface ge-0/0/13.0
set vlans hr-comm interface ge-0/0/14.0
set vlans hr-comm primary-vlan pvlan100
set vlans pvlan100 vlan-id 100
set vlans pvlan100  interface ge-0/0/15.0
set vlans pvlan100  interface ge-0/0/16.0
```

```
set vlans pvlan100 interface ge-0/0/0.0 pvlan-trunk
set vlans pvlan100 interface ge-0/0/5.0 pvlan-trunk
set vlans pvlan100 no-local-switching
set vlans pvlan100 isolation-id 50
```

Procedure

Step-by-Step Procedure

Complete the configuration steps below in the order shown—also, complete all steps before committing the configuration in a single commit. This is the easiest way to avoid error messages triggered by violating any of these three rules:

- If you are going to configure a community VLAN ID, you must first configure the primary VLAN and the PVLAN trunk port.
- If you are going to configure an isolation VLAN ID, you must first configure the primary VLAN and the PVLAN trunk port.
- Secondary vlans and a PVLAN trunk must be committed on a single commit.

To configure a PVLAN on Switch 1 that will span multiple switches:

1. Set the VLAN ID for the primary VLAN:

```
[edit vlans]
user@switch# set pvlan100 vlan-id 100
```

2. Set the PVLAN trunk interfaces that will connect this VLAN across neighboring switches:

```
[edit vlans]
user@switch# set pvlan100 interface ge-0/0/0.0 pvlan-trunk
user@switch# set  pvlan100 interface ge-0/0/5.0 pvlan-trunk
```

3. Set the primary VLAN to have no local switching:

```
 [edit vlans]
user@switch# set pvlan100 no-local-switching
```

4. Set the VLAN ID for the **finance-comm** community VLAN that spans the switches:

```
[edit vlans]
user@switch# finance-comm vlan-id 300
```

```
user@switch# set pvlan100 vlan-id 100
```

5. Configure access interfaces for the **finance-comm** VLAN:

```
 [edit vlans]
user@switch# set finance-comm interface interfacege-0/0/11.0
```

```
 user@switch# set finance-comm interface ge-0/0/12.0
```

6. Set the primary VLAN of this secondary community VLAN, **finance-comm** :

```
[edit vlans]
user@switch# set vlans finance-comm primary-vlan pvlan100
```

7. Set the VLAN ID for the HR community VLAN that spans the switches.

```
[edit vlans]
user@switch# hr-comm vlan-id 400
```

8. Configure access interfaces for the **hr-comm** VLAN:

```
 [edit vlans]
user@switch# set hr-comm interface ge-0/0/13.0
 user@switch# set hr-comm interface ge-0/0/14.0
```

9. Set the primary VLAN of this secondary community VLAN, **hr-comm** :

```
[edit vlans]
user@switch# set vlans hr-comm primary-vlan pvlan100
```

10. Set the inter-switch isolated ID to create an inter-switch isolated domain that spans the switches:

```
[edit vlans]
user@switch# set pvlan100 isolation-id 50
```

NOTE: To configure an isolated port, include it as one of the members of the primary VLAN but do not configure it as belonging to one of the community VLANs.

Results

Check the results of the configuration:

```
[edit]
user@switch# show
vlans {
    finance-comm {
        vlan-id 300;
        interface {
            ge-0/0/11.0;
            ge-0/0/12.0;
        }
        primary-vlan pvlan100;
    }
    hr-comm {
        vlan-id 400;
        interface {
            ge-0/0/13.0;
            ge-0/0/14.0;
        }
        primary-vlan pvlan100;
    }
    pvlan100 {
        vlan-id 100;
        interface {
            ge-0/0/15.0;
            ge-0/0/16.0;
            ge-0/0/0.0 {
                pvlan-trunk;
            }
```

```
            ge-0/0/5.0 {
                pvlan-trunk;
            }
        }
        no-local-switching;
        isolation-id 50;
    }
}
```

Configuring a PVLAN on Switch 2

IN THIS SECTION

CLI Quick Configuration

To quickly create and configure a private VLAN spanning multiple switches, copy the following commands and paste them into the terminal window of Switch 2:

NOTE: The configuration of Switch 2 is the same as the configuration of Switch 1 except for the interface in the inter-switch isolated domain. For Switch 2, the interface is **ge-0/0/17.0**.

```
[edit]
set vlans finance-comm vlan-id 300
set vlans finance-comm interface ge-0/0/11.0
set vlans finance-comm interface ge-0/0/12.0
set vlans finance-comm primary-vlan pvlan100
set vlans hr-comm vlan-id 400
set vlans hr-comm interface ge-0/0/13.0
set vlans hr-comm interface ge-0/0/14.0
set vlans hr-comm primary-vlan pvlan100
set vlans pvlan100 vlan-id 100
set vlans pvlan100  interface ge-0/0/17.0
```

```
set vlans pvlan100 interface ge-0/0/0.0 pvlan-trunk
set vlans pvlan100 interface ge-0/0/5.0 pvlan-trunk
set vlans pvlan100 no-local-switching
set vlans pvlan100 isolation-id 50
```

Procedure

Step-by-Step Procedure

To configure a PVLAN on Switch 2 that will span multiple switches:

1. Set the VLAN ID for the **finance-comm** community VLAN that spans the switches:

```
[edit vlans]
user@switch# finance-comm vlan-id 300
```

```
user@switch# set pvlan100 vlan-id 100
```

2. Configure access interfaces for the **finance-comm** VLAN:

```
 [edit vlans]
user@switch# set finance-comm interface ge-0/0/11.0
```

```
 user@switch# set finance-comm interface ge-0/0/12.0
```

3. Set the primary VLAN of this secondary community VLAN, **finance-comm** :

```
[edit vlans]
user@switch# set vlans finance-comm primary-vlan pvlan100
```

4. Set the VLAN ID for the HR community VLAN that spans the switches.

```
[edit vlans]
user@switch# hr-comm vlan-id 400
```

5. Configure access interfaces for the **hr-comm** VLAN:

```
[edit vlans]
user@switch# set hr-comm interface ge-0/0/13.0
user@switch# set hr-comm interface ge-0/0/14.0
```

6. Set the primary VLAN of this secondary community VLAN, **hr-comm** :

```
[edit vlans]
user@switch# set vlans hr-comm primary-vlan pvlan100
```

7. Set the VLAN ID for the primary VLAN:

```
[edit vlans]
user@switch# set pvlan100 vlan-id 100
```

8. Set the PVLAN trunk interfaces that will connect this VLAN across neighboring switches:

```
[edit vlans]
user@switch# set pvlan100 interface ge-0/0/0.0 pvlan-trunk
user@switch# set  pvlan100 interface ge-0/0/5.0 pvlan-trunk
```

9. Set the primary VLAN to have no local switching:

```
[edit vlans]
user@switch# set pvlan100 no-local-switching
```

10. Set the inter-switch isolated ID to create an inter-switch isolated domain that spans the switches:

```
[edit vlans]
user@switch# set pvlan100 isolation-id 50
```

NOTE: To configure an isolated port, include it as one of the members of the primary VLAN but do not configure it as belonging to one of the community VLANs.

Results

Check the results of the configuration:

```
[edit]
user@switch# show
vlans {
    finance-comm {
        vlan-id 300;
        interface {
            ge-0/0/11.0;
            ge-0/0/12.0;
        }
        primary-vlan pvlan100;
    }
    hr-comm {
        vlan-id 400;
        interface {
            ge-0/0/13.0;
            ge-0/0/14.0;
        }
        primary-vlan pvlan100;
    }
    pvlan100 {
        vlan-id 100;
        interface {
            ge-0/0/15.0;
            ge-0/0/16.0;
            ge-0/0/0.0 {
                pvlan-trunk;
            }
            ge-0/0/5.0 {
                pvlan-trunk;
            }
            ge-0/0/17.0;
        }
        no-local-switching;
        isolation-id 50;
    }
}
```

Configuring a PVLAN on Switch 3

IN THIS SECTION

CLI Quick Configuration

To quickly configure Switch 3 to function as the distribution switch of this PVLAN, copy the following commands and paste them into the terminal window of Switch 3:

NOTE: Interface ge-0/0/2.0 is a trunk port connecting the PVLAN to a router.

```
[edit]
set vlans finance-comm vlan-id 300
set vlans finance-comm primary-vlan pvlan100
set vlans hr-comm vlan-id 400
set vlans hr-comm primary-vlan pvlan100
set vlans pvlan100 vlan-id 100
set vlans pvlan100 interface ge-0/0/0.0 pvlan-trunk
set vlans pvlan100 interface ge-0/0/1.0 pvlan-trunk
set vlans pvlan100 no-local-switching
set vlans pvlan100 isolation-id 50
```

Procedure

Step-by-Step Procedure

To configure Switch 3 to function as the distribution switch for this PVLAN, use the following procedure:

1. Set the VLAN ID for the **finance-comm** community VLAN that spans the switches:

```
[edit vlans]
user@switch# finance-comm vlan-id 300
```

```
[edit vlans]
user@switch# set pvlan100 vlan-id 100
```

2. Set the primary VLAN of this secondary community VLAN, **finance-comm**:

```
[edit vlans]
user@switch# set vlans finance-comm primary-vlan pvlan100
```

3. Set the VLAN ID for the HR community VLAN that spans the switches:

```
[edit vlans]
user@switch# hr-comm vlan-id 400
```

4. Set the primary VLAN of this secondary community VLAN, **hr-comm**:

```
[edit vlans]
user@switch# set vlans hr-comm primary-vlan pvlan100
```

5. Set the VLAN ID for the primary VLAN:

```
[edit vlans]
user@switch# set pvlan100 vlan-id 100
```

6. Set the PVLAN trunk interfaces that will connect this VLAN across neighboring switches:

```
[edit vlans]
user@switch# set pvlan100 interface ge-0/0/0.0 pvlan-trunk
user@switch# set  pvlan100 interface ge-0/0/5.0 pvlan-trunk
```

7. Set the primary VLAN to have no local switching:

```
[edit vlans]
user@switch# set pvlan100 no-local-switching
```

8. Set the inter-switch isolated ID to create an inter-switch isolated domain that spans the switches:

```
[edit vlans]
user@switch# set pvlan100 isolation-id 50
```

NOTE: To configure an isolated port, include it as one of the members of the primary VLAN but do not configure it as belonging to one of the community VLANs.

Results

Check the results of the configuration:

```
[edit]
user@switch# show
vlans {
    finance-comm {
        vlan-id 300;
        primary-vlan pvlan100;
    }
    hr-comm {
        vlan-id 400;
        primary-vlan pvlan100;
    }
    pvlan100 {
        vlan-id 100;
        interface {
            ge-0/0/0.0 {
                pvlan-trunk;
            }
            ge-0/0/1.0 {
                pvlan-trunk;
            }
            ge-0/0/2.0;
```

```
        }
        no-local-switching;
        isolation-id 50;
    }
}
```

Verification

IN THIS SECTION

To confirm that the configuration is working properly, perform these tasks:

Verifying That the Primary VLAN and Secondary VLANs Were Created on Switch 1

Purpose

Verify that the PVLAN configuration spanning multiple switches is working properly on Switch 1:

Action

Use the `show vlans extensive` command:

```
user@switch> show vlans extensive
VLAN: __pvlan_pvlan100_ge-0/0/15.0__, Created at: Thu Sep 16 23:15:27 2010
Internal index: 5, Admin State: Enabled, Origin: Static
Private VLAN Mode: Isolated, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  1 (Active = 1)
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/5.0*, tagged, trunk, pvlan-trunk
      ge-0/0/15.0*, untagged, access

VLAN: __pvlan_pvlan100_ge-0/0/16.0__, Created at: Thu Sep 16 23:15:27 2010
```

```
Internal index: 6, Admin State: Enabled, Origin: Static
Private VLAN Mode: Isolated, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  1 (Active = 1)
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/5.0*, tagged, trunk, pvlan-trunk
      ge-0/0/16.0*, untagged, access

VLAN: __pvlan_pvlan100_isiv__, Created at: Thu Sep 16 23:15:27 2010
802.1Q Tag: 50, Internal index: 7, Admin State: Enabled, Origin: Static
Private VLAN Mode: Inter-switch-isolated, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  0 (Active = 0)
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/5.0*, tagged, trunk, pvlan-trunk

VLAN: default, Created at: Thu Sep 16 03:03:18 2010
Internal index: 2, Admin State: Enabled, Origin: Static
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 0 (Active = 0), Untagged  0 (Active = 0)

VLAN: finance-comm, Created at: Thu Sep 16 23:15:27 2010
802.1Q Tag: 300, Internal index: 8, Admin State: Enabled, Origin: Static
Private VLAN Mode: Community, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  2 (Active = 2)
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/5.0*, tagged, trunk, pvlan-trunk
      ge-0/0/11.0*, untagged, access
      ge-0/0/12.0*, untagged, access

VLAN: hr-comm, Created at: Thu Sep 16 23:15:27 2010
802.1Q Tag: 400, Internal index: 9, Admin State: Enabled, Origin: Static
Private VLAN Mode: Community, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  2 (Active = 2)
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/5.0*, tagged, trunk, pvlan-trunk
      ge-0/0/13.0*, untagged, access
      ge-0/0/14.0*, untagged, access

VLAN: pvlan100, Created at: Thu Sep 16 23:15:27 2010
802.1Q Tag: 100, Internal index: 4, Admin State: Enabled, Origin: Static
```

```
Private VLAN Mode: Primary
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  6 (Active = 6)
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/5.0*, tagged, trunk, pvlan-trunk
      ge-0/0/11.0*, untagged, access
      ge-0/0/12.0*, untagged, access
      ge-0/0/13.0*, untagged, access
      ge-0/0/14.0*, untagged, access
      ge-0/0/15.0*, untagged, access
      ge-0/0/16.0*, untagged, access
Secondary VLANs: Isolated 2, Community  2, Inter-switch-isolated  1
  Isolated VLANs :
      __pvlan_pvlan100_ge-0/0/15.0__
      __pvlan_pvlan100_ge-0/0/16.0__
  Community VLANs :
      finance-comm
      hr-comm
  Inter-switch-isolated VLAN :
      __pvlan_pvlan100_isiv__
```

Meaning

The output shows that a PVLAN was created on Switch 1 and shows that it includes two isolated VLANs, two community VLANs, and an interswitch isolated VLAN. The presence of the **pvlan-trunk** and **Inter-switch-isolated** fields indicates that this PVLAN is spanning more than one switch.

Verifying That the Primary VLAN and Secondary VLANs Were Created on Switch 2

Purpose

Verify that the PVLAN configuration spanning multiple switches is working properly on Switch 2:

Action

Use the `show vlans extensive` command:

```
user@switch> show vlans extensive
VLAN: __pvlan_pvlan100_ge-0/0/17.0__, Created at: Thu Sep 16 23:19:22 2010
Internal index: 5, Admin State: Enabled, Origin: Static
Private VLAN Mode: Isolated, Primary VLAN: pvlan100
```

```
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  1 (Active = 1)
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/5.0*, tagged, trunk, pvlan-trunk
      ge-0/0/17.0*, untagged, access

VLAN: __pvlan_pvlan100_isiv__, Created at: Thu Sep 16 23:19:22 2010
802.1Q Tag: 50, Internal index: 6, Admin State: Enabled, Origin: Static
Private VLAN Mode: Inter-switch-isolated, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  0 (Active = 0)
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/5.0*, tagged, trunk, pvlan-trunk

VLAN: default, Created at: Thu Sep 16 03:03:18 2010
Internal index: 2, Admin State: Enabled, Origin: Static
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 0 (Active = 0), Untagged  0 (Active = 0)

VLAN: finance-comm, Created at: Thu Sep 16 23:19:22 2010
802.1Q Tag: 300, Internal index: 7, Admin State: Enabled, Origin: Static
Private VLAN Mode: Community, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  2 (Active = 2)
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/5.0*, tagged, trunk, pvlan-trunk
      ge-0/0/11.0*, untagged, access
      ge-0/0/12.0*, untagged, access

VLAN: hr-comm, Created at: Thu Sep 16 23:19:22 2010
802.1Q Tag: 400, Internal index: 8, Admin State: Enabled, Origin: Static
Private VLAN Mode: Community, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  2 (Active = 2)
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/5.0*, tagged, trunk, pvlan-trunk
      ge-0/0/13.0*, untagged, access
      ge-0/0/14.0*, untagged, access

VLAN: pvlan100, Created at: Thu Sep 16 23:19:22 2010
802.1Q Tag: 100, Internal index: 4, Admin State: Enabled, Origin: Static
Private VLAN Mode: Primary
Protocol: Port Mode, Mac aging time: 300 seconds
```

```
Number of interfaces: Tagged 2 (Active = 2), Untagged  5 (Active = 5)
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/5.0*, tagged, trunk, pvlan-trunk
      ge-0/0/11.0*, untagged, access
      ge-0/0/12.0*, untagged, access
      ge-0/0/13.0*, untagged, access
      ge-0/0/14.0*, untagged, access
      ge-0/0/17.0*, untagged, access
Secondary VLANs: Isolated 1, Community  2, Inter-switch-isolated  1
  Isolated VLANs :
      __pvlan_pvlan100_ge-0/0/17.0__
  Community VLANs :
      finance-comm
      hr-comm
  Inter-switch-isolated VLAN :
      __pvlan_pvlan100_isiv__
```

Meaning

The output shows that a PVLAN was created on Switch 1 and shows that it includes two isolated VLANs, two community VLANs, and an interswitch isolated VLAN. The presence of the **pvlan-trunk** and **Inter-switch-isolated** fields indicates that this is PVLAN spanning more than one switch. When you compare this output to the output of Switch 1, you can see that both switches belong to the same PVLAN (**pvlan100**).

Verifying That the Primary VLAN and Secondary VLANs Were Created on Switch 3

Purpose

Verify that the PVLAN configuration spanning multiple switches is working properly on Switch 3:

Action

Use the `show vlans extensive` command:

```
user@switch> show vlans extensive
VLAN: __pvlan_pvlan100_isiv__, Created at: Thu Sep 16 23:22:40 2010
802.1Q Tag: 50, Internal index: 5, Admin State: Enabled, Origin: Static
Private VLAN Mode: Inter-switch-isolated, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  0 (Active = 0)
```

```
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/1.0*, tagged, trunk, pvlan-trunk

VLAN: default, Created at: Thu Sep 16 03:03:18 2010
Internal index: 2, Admin State: Enabled, Origin: Static
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 0 (Active = 0), Untagged  0 (Active = 0)

VLAN: finance-comm, Created at: Thu Sep 16 23:22:40 2010
802.1Q Tag: 300, Internal index: 6, Admin State: Enabled, Origin: Static
Private VLAN Mode: Community, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  0 (Active = 0)
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/1.0*, tagged, trunk, pvlan-trunk

VLAN: hr-comm, Created at: Thu Sep 16 23:22:40 2010
802.1Q Tag: 400, Internal index: 7, Admin State: Enabled, Origin: Static
Private VLAN Mode: Community, Primary VLAN: pvlan100
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  0 (Active = 0)
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/1.0*, tagged, trunk, pvlan-trunk

VLAN: pvlan100, Created at: Thu Sep 16 23:22:40 2010
802.1Q Tag: 100, Internal index: 4, Admin State: Enabled, Origin: Static
Private VLAN Mode: Primary
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 2 (Active = 2), Untagged  0 (Active = 0)
      ge-0/0/0.0*, tagged, trunk, pvlan-trunk
      ge-0/0/1.0*, tagged, trunk, pvlan-trunk
Secondary VLANs: Isolated 0, Community  2, Inter-switch-isolated  1
  Community VLANs :
      finance-comm
      hr-comm
  Inter-switch-isolated VLAN :
      __pvlan_pvlan100_isiv__
```

Meaning

The output shows that the PVLAN (**pvlan100**) is configured on Switch 3 and that it includes two isolated VLANs, two community VLANs, and an interswitch isolated VLAN. But Switch 3 is functioning as a

distribution switch, so the output does not include access interfaces within the PVLAN. It shows only the **pvlan-trunk** interfaces that connect **pvlan100** from Switch 3 to the other switches (Switch 1 and Switch 2) in the same PVLAN.

Example: Configuring PVLANs with Secondary VLAN Trunk Ports and Promiscuous Access Ports on a QFX Series Switch

IN THIS SECTION

This example shows how to configure secondary VLAN trunk ports and promiscuous access ports as part of a private VLAN configuration. Secondary VLAN trunk ports carry secondary VLAN traffic.

NOTE: This example uses Junos OS for switches that do not support the Enhanced Layer 2 Software (ELS) configuration style. For more about ELS, see "Using the Enhanced Layer 2 Software CLI" on page 15.

For a given private VLAN, a secondary VLAN trunk port can carry traffic for only one secondary VLAN. However, a secondary VLAN trunk port can carry traffic for multiple secondary VLANs as long as each secondary VLAN is a member of a different private (primary) VLAN. For example, a secondary VLAN trunk port can carry traffic for a community VLAN that is part of primary VLAN pvlan100 and also carry traffic for an isolated VLAN that is part of primary VLAN pvlan400.

To configure a trunk port to carry secondary VLAN traffic, use the *isolated* and `interface` statements, as shown in steps "12" on page 591 and "13" on page 591 of the example configuration for Switch 1.

NOTE: When traffic egresses from a secondary VLAN trunk port, it normally carries the tag of the primary VLAN that the secondary port is a member of. If you want traffic that

egresses from a secondary VLAN trunk port to retain its secondary VLAN tag, use the *extend-secondary-vlan-id* statement.

A promiscuous access port carries untagged traffic and can be a member of only one primary VLAN. Traffic that ingresses on a promiscuous access port is forwarded to the ports of the secondary VLANs that are members of the primary VLAN that the promiscuous access port is a member of. This traffic carries the appropriate secondary VLAN tags when it egresses from the secondary VLAN ports if the secondary VLAN port is a trunk port.

To configure an access port to be promiscuous, use the *promiscuous* statement, as shown in step "12" on page 598 of the example configuration for Switch 2.

If traffic ingresses on a secondary VLAN port and egresses on a promiscuous access port, the traffic is untagged on egress. If tagged traffic ingresses on a promiscuous access port, the traffic is discarded.

Requirements

This example uses the following hardware and software components:

- Two QFX devices
- Junos OS Release 12.2 or later for the QFX Series

Overview and Topology

Figure 28 on page 586 shows the topology used in this example. Switch 1 includes several primary and secondary private VLANs and also includes two secondary VLAN trunk ports configured to carry secondary VLANs that are members of primary VLANs pvlan100 and pvlan400.

Switch 2 includes the same private VLANs. The figure shows xe-0/0/0 on Switch 2 as configured with promiscuous access ports or promiscuous trunk ports. The example configuration included here configures this port as a promiscuous access port.

The figure also shows how traffic would flow after ingressing on the secondary VLAN trunk ports on Switch 1.

Figure 28: PVLAN Topology with Secondary VLAN Trunk Ports and Promiscuous Access Port

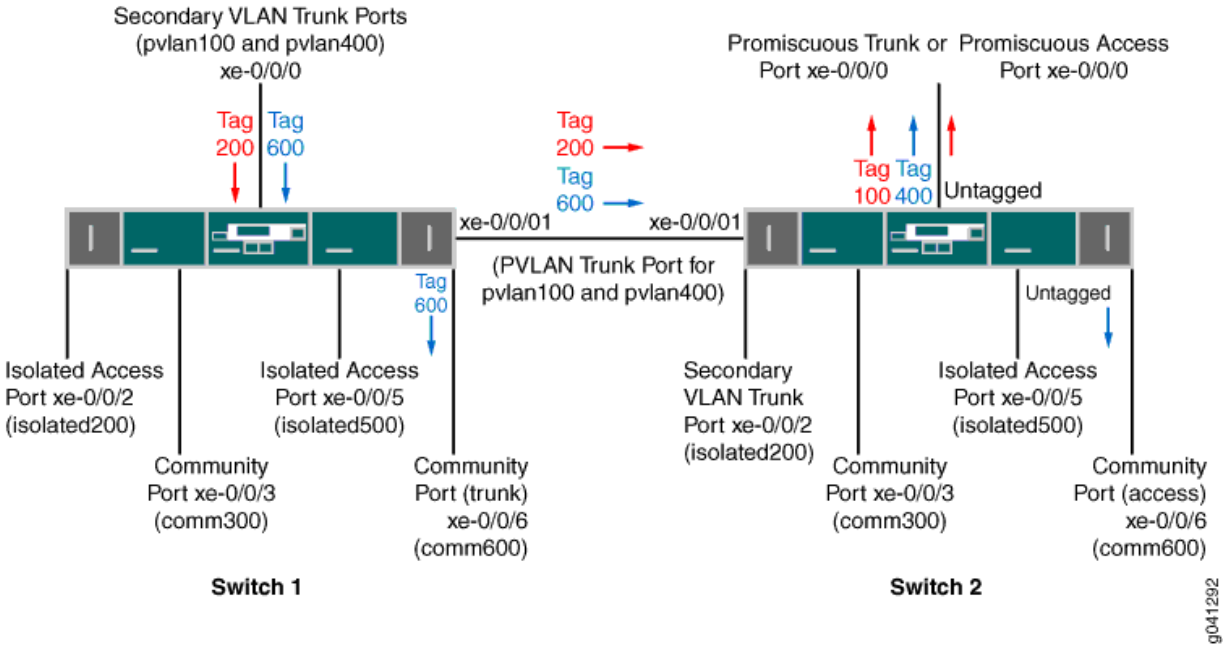


Table 92 on page 586 and Table 93 on page 587 list the settings for the example topology on both switches.

Table 92: Components of the Topology for Configuring a Secondary VLAN Trunk on Switch 1

Component	Description
pvlan100, ID 100	Primary VLAN
pvlan400, ID 400	Primary VLAN
comm300, ID 300	Community VLAN, member of pvlan100
comm600, ID 600	Community VLAN, member of pvlan400
isolation-vlan-id 200	VLAN ID for isolated VLAN, member of pvlan100
isolation–vlan-id 500	VLAN ID for isolated VLAN, member of pvlan400
xe-0/0/0.0	Secondary VLAN trunk port for primary VLANs pvlan100 and pvlan400

Table 92: Components of the Topology for Configuring a Secondary VLAN Trunk on Switch 1 ***(Continued)***

Component	Description
xe-0/0/1.0	PVLAN trunk port for primary VLANs pvlan100 and pvlan400
xe-0/0/2.0	Isolated access port for pvlan100
xe-0/0/3.0	Community access port for comm300
xe-0/0/5.0	Isolated access port for pvlan400
xe-0/0/6.0	Community trunk port for comm600

Table 93: Components of the Topology for Configuring a Secondary VLAN Trunk on Switch 2

Component	Description
pvlan100, ID 100	Primary VLAN
pvlan400, ID 400	Primary VLAN
comm300, ID 300	Community VLAN, member of pvlan100
comm600, ID 600	Community VLAN, member of pvlan400
isolation-vlan-id 200	VLAN ID for isolated VLAN, member of pvlan100
isolation–vlan-id 500	VLAN ID for isolated VLAN, member of pvlan400
xe-0/0/0.0	Promiscuous access port for primary VLANs pvlan100
xe-0/0/1.0	PVLAN trunk port for primary VLANs pvlan100 and pvlan400
xe-0/0/2.0	Secondary trunk port for isolated VLAN, member of pvlan100

Table 93: Components of the Topology for Configuring a Secondary VLAN Trunk on Switch 2 *(Continued)*

Component	Description
xe-0/0/3.0	Community access port for comm300
xe-0/0/5.0	Isolated access port for pvlan400
xe-0/0/6.0	Community access port for comm600

Configuring the PVLANs on Switch 1

IN THIS SECTION

CLI Quick Configuration

To quickly create and configure the PVLANs on Switch 1, copy the following commands and paste them into a switch terminal window:

```
[edit]
set interfaces xe-0/0/0 unit 0 family ethernet-switching port-mode trunk
set interfaces xe-0/0/1 unit 0 family ethernet-switching port-mode trunk
set interfaces xe-0/0/1 unit 0 family ethernet-switching vlan members pvlan100
set interfacesxe-0/0/1 unit 0 family ethernet-switching vlan members pvlan400

set interfaces xe-0/0/2 unit 0 family ethernet-switching port-mode access
set interfaces xe-0/0/3 unit 0 family ethernet-switching port-mode access
set interfaces xe-0/0/5 unit 0 family ethernet-switching port-mode access
set interfaces xe-0/0/6 unit 0 family ethernet-switching port-mode trunk
set vlans pvlan100 vlan-id 100
set vlans pvlan400 vlan-id 400
```

```
set vlans pvlan100 pvlan
set vlans pvlan400 pvlan
set vlans pvlan100 interface xe-0/0/1.0 pvlan-trunk

set vlans pvlan400 interface xe-0/0/1.0 pvlan-trunk
set vlans comm300 vlan-id 300
set vlans comm300 primary-vlan pvlan100
set vlans comm300 interface xe-0/0/3.0
set vlans comm600 vlan-id 600
set vlans comm600 primary-vlan pvlan400
set vlans comm600 interface xe-0/0/6.0
set vlans pvlan100 pvlan isolation-vlan-id 200
set vlans pvlan400 pvlan isolation-vlan-id 500
set vlans pvlan100 interface xe-0/0/0.0 isolated
set vlans pvlan400 interface xe-0/0/0.0 isolated
set vlans comm600 interface xe-0/0/0.0
set vlans pvlan100 interface xe-0/0/2.0 isolated
set vlans pvlan400 interface xe-0/0/5.0 isolated
```

Procedure

Step-by-Step Procedure

To configure the private VLANs and secondary VLAN trunk ports:

1. Configure the interfaces and port modes:

```
[edit interfaces]
user@switch# set xe-0/0/0 unit 0 family ethernet-switching port-mode trunk
user@switch# set xe-0/0/1 unit 0 family ethernet-switching port-mode trunk
user@switch# set xe-0/0/1 unit 0 family ethernet-switching vlan members pvlan100
user@switch# set xe-0/0/1 unit 0 family ethernet-switching vlan members pvlan400
user@switch# set xe-0/0/2 unit 0 family ethernet-switching port-mode access
user@switch# set xe-0/0/3 unit 0 family ethernet-switching port-mode access
user@switch# set xe-0/0/5 unit 0 family ethernet-switching port-mode access
user@switch# set xe-0/0/6 unit 0 family ethernet-switching port-mode access
```

2. Create the primary VLANs:

```
[edit vlans]
user@switch# set pvlan100 vlan-id 100
user@switch# set pvlan400 vlan-id 400
```

NOTE: Primary VLANs must always be tagged VLANs, even if they exist on only one device.

3. Configure the primary VLANs to be private:

```
 [edit vlans]
user@switch# set pvlan100 pvlan
user@switch# set pvlan400 pvlan
```

4. Configure the PVLAN trunk port to carry the private VLAN traffic between the switches:

```
[edit vlans]
user@switch# set pvlan100 interface xe-0/0/1.0 pvlan-trunk
user@switch# set pvlan400 interface xe-0/0/1.0 pvlan-trunk
```

5. Create secondary VLAN comm300 with VLAN ID 300:

```
[edit vlans]
user@switch# set comm300 vlan-id 300
```

6. Configure the primary VLAN for comm300:

```
[edit vlans]
user@switch# set comm300 primary-vlan pvlan100
```

7. Configure the interface for comm300:

```
[edit vlans]
user@switch# set comm300 interface xe-0/0/3.0
```

8. Create secondary VLAN comm600 with VLAN ID 600:

```
[edit vlans]
user@switch# set comm600 vlan-id 600
```

9. Configure the primary VLAN for comm600:

```
[edit vlans]
user@switch# set comm600 primary-vlan pvlan400
```

10. Configure the interface for comm600:

```
[edit vlans]
user@switch# set comm600 interface xe-0/0/6.0
```

11. Configure the interswitch isolated VLANs:

```
[edit vlans]
user@switch# set pvlan100 pvlan isolation-vlan-id 200
user@switch# set pvlan400 pvlan isolation-vlan-id 500
```

NOTE: When you configure a secondary VLAN trunk port to carry an isolated VLAN, you must also configure an *isolation-vlan-id*. This is true even if the isolated VLAN exists only on one switch.

12. Enable trunk port xe-0/0/0 to carry secondary VLANs for the primary VLANs:

```
[edit vlans]
user@switch# set pvlan100 interface xe-0/0/0.0 isolated
user@switch# set pvlan400 interface xe-0/0/0.0 isolated
```

13. Configure trunk port xe-0/0/0 to carry comm600 (member of pvlan400):

```
[edit vlans]
user@switch# set comm600 interface xe-0/0/0.0
```

NOTE: You do not need to explicitly configure xe-0/0/0 to carry the isolated VLAN traffic (tags 200 and 500) because all the isolated ports in pvlan100 and pvlan400–including xe-0/0/0.0–are automatically included in the isolated VLANs created when you configured `isolation-vlan-id 200` and `isolation-vlan-id 500`.

14. Configure xe-0/0/2 and xe-0/0/6 to be isolated:

```
[edit vlans]
user@switch# set pvlan100 interface xe-0/0/2.0 isolated
user@switch# set pvlan400 interface xe-0/0/5.0 isolated
```

Results

Check the results of the configuration on Switch 1:

```
[edit]
user@switch#  show
interfaces {
    xe-0/0/0 {
        unit 0 {
            family ethernet-switching {
                port-mode trunk;
                vlan {
                    members pvlan100;
                    members pvlan400;
                }
            }
        }
    }
    xe-0/0/1 {
        unit 0 {
            family ethernet-switching {
                port-mode trunk;
                vlan {
                    members pvlan100;
                    members pvlan400;
                }
            }
        }
```

```
    }
    xe-0/0/2 {
        unit 0 {
            family ethernet-switching {
                port-mode access;
            }
        }
    }
    xe-0/0/3 {
        unit 0 {
            family ethernet-switching {
                port-mode access;
            }
        }
    }
    xe-0/0/5 {
        unit 0 {
            family ethernet-switching {
                port-mode access;
            }
        }
    }
    xe-0/0/6 {
        unit 0 {
            family ethernet-switching {
                port-mode trunk;
            }
        }
    }
}
vlans {
    comm300 {
        vlan-id 300;
            interface {
            xe-0/0/3.0;
        }
        primary-vlan pvlan100;
    }
    comm600 {
        vlan-id 600;
            interface {
            xe-0/0/6.0;
        }
```

```
        primary-vlan pvlan400;
    }
    pvlan100 {
        vlan-id 100;
        interface {
            xe-0/0/0.0;
            xe-0/0/2.0;
            xe-0/0/3.0;
            xe-0/0/1.0 {
                pvlan-trunk;
            }
        }
        no-local-switching;
        isolation-id 200;
    }
    pvlan400 {
        vlan-id 400;
        interface {
            xe-0/0/0.0;
            xe-0/0/5.0;
            xe-0/0/6.0;
            xe-0/0/1.0 {
                pvlan-trunk;
            }
        }
        no-local-switching;
        isolation-id 500;
    }
}
```

Configuring the PVLANs on Switch 2

IN THIS SECTION

The configuration for Switch 2 is almost identical to the configuration for Switch 1. The most significant difference is that xe-0/0/0 on Switch 2 is configured as a promiscuous trunk port or a promiscuous access port, as Figure 28 on page 586 shows. In the following configuration, xe-0/0/0 is configured as a promiscuous access port for primary VLAN pvlan100.

If traffic ingresses on VLAN-enabled port and egresses on a promiscuous access port, the VLAN tags are dropped on egress and the traffic is untagged at that point. For example, traffic for comm600 ingresses on the secondary VLAN trunk port configured on xe-0/0/0.0 on Switch 1 and carries tag 600 as it is forwarded through the secondary VLAN. When it egresses from xe-0/0/0.0 on Switch 2, it will be untagged if you configure xe-0/0/0.0 as a promiscuous access port as shown in this example. If you instead configure xe-0/0/0.0 as a promiscuous trunk port (port-mode trunk), the traffic for comm600 carries its primary VLAN tag (400) when it egresses.

CLI Quick Configuration

To quickly create and configure the PVLANs on Switch 2, copy the following commands and paste them into a switch terminal window:

```
[edit]
set interfaces xe-0/0/0 unit 0 family ethernet-switching port-mode access
set interfaces xe-0/0/1 unit 0 family ethernet-switching port-mode trunk
set interfaces xe-0/0/1 unit 0 family ethernet-switching vlan members pvlan100
set interfaces xe-0/0/1 unit 0 family ethernet-switching vlan members pvlan400

set interfaces xe-0/0/2 unit 0 family ethernet-switching port-mode trunk
set interfaces xe-0/0/3 unit 0 family ethernet-switching port-mode access
set interfaces xe-0/0/5 unit 0 family ethernet-switching port-mode access
set interfaces xe-0/0/6 unit 0 family ethernet-switching port-mode access
set vlans pvlan100 vlan-id 100
set vlans pvlan400 vlan-id 400
set vlans pvlan100 pvlan
set vlans pvlan400 pvlan
set vlans pvlan100 interface xe-0/0/1.0 pvlan-trunk

set vlans pvlan400 interface xe-0/0/1.0 pvlan-trunk
set vlans comm300 vlan-id 300
set vlans comm300 primary-vlan pvlan100
set vlans comm300 interface xe-0/0/3.0
set vlans comm600 vlan-id 600
set vlans comm600 primary-vlan pvlan400
set vlans comm600 interface xe-0/0/6.0
set vlans pvlan100 pvlan isolation-vlan-id 200
```

```
set vlans pvlan400 pvlan isolation-vlan-id 500
set vlans pvlan100 interface xe-0/0/0.0 promiscuous
set vlans pvlan100 interface xe-0/0/2.0 isolated
set vlans pvlan400 interface xe-0/0/5.0 isolated
```

Procedure

Step-by-Step Procedure

To configure the private VLANs and secondary VLAN trunk ports:

1. Configure the interfaces and port modes:

```
[edit interfaces]
user@switch# set xe-0/0/0 unit 0 family ethernet-switching port-mode access
```

```
user@switch# set xe-0/0/1 unit 0 family ethernet-switching port-mode trunk
user@switch# set xe-0/0/1 unit 0 family ethernet-switching vlan members pvlan100
user@switch# set xe-0/0/1 unit 0 family ethernet-switching vlan members pvlan400

user@switch# set xe-0/0/2 unit 0 family ethernet-switching port-mode trunk
user@switch# set xe-0/0/3 unit 0 family ethernet-switching port-mode access
user@switch# set xe-0/0/5 unit 0 family ethernet-switching port-mode access
user@switch# set xe-0/0/6 unit 0 family ethernet-switching port-mode access
```

2. Create the primary VLANs:

```
[edit vlans]
user@switch# set pvlan100 vlan-id 100
user@switch# set pvlan400 vlan-id 400
```

3. Configure the primary VLANs to be private:

```
 [edit vlans]
user@switch# set pvlan100 pvlan
user@switch# set pvlan400 pvlan
```

4. Configure the PVLAN trunk port to carry the private VLAN traffic between the switches:

```
[edit vlans]
user@switch# set pvlan100 interface xe-0/0/1.0 pvlan-trunk
user@switch# set pvlan400 interface xe-0/0/1.0 pvlan-trunk
```

5. Create secondary VLAN comm300 with VLAN ID 300:

```
[edit vlans]
user@switch# set comm300 vlan-id 300
```

6. Configure the primary VLAN for comm300:

```
[edit vlans]
user@switch# set comm300 primary-vlan pvlan100
```

7. Configure the interface for comm300:

```
[edit vlans]
user@switch# set comm300 interface xe-0/0/3.0
```

8. Create secondary VLAN comm600 with VLAN ID 600:

```
[edit vlans]
user@switch# set comm600 vlan-id 600
```

9. Configure the primary VLAN for comm600:

```
[edit vlans]
user@switch# set comm600 primary-vlan pvlan400
```

10. Configure the interface for comm600:

```
[edit vlans]
user@switch# set comm600 interface xe-0/0/6.0
```

11. Configuring the PVLANs on Switch 1

    Configure the interswitch isolated VLANs:

```
[edit vlans]
user@switch# set pvlan100 pvlan isolation-vlan-id 200
user@switch# set pvlan400 pvlan isolation-vlan-id 500
```

12. Configure access port xe-0/0/0 to be promiscuous for pvlan100:

```
[edit vlans]
user@switch# set pvlan100 interface xe-0/0/0.0 promiscuous
```

NOTE: A promiscuous access port can be a member of only one primary VLAN.

13. Configure xe-0/0/2 and xe-0/0/6 to be isolated:

```
[edit vlans]
user@switch# set pvlan100 interface xe-0/0/2.0 isolated
user@switch# set pvlan400 interface xe-0/0/5.0 isolated
```

Results

Check the results of the configuration on Switch 2:

```
[edit]
user@switch#  show
interfaces {
    xe-0/0/0 {
        unit 0 {
            family ethernet-switching {
                port-mode access;
                vlan {
                    members pvlan100;
                }
            }
        }
    }
```

```
        xe-0/0/1 {
        unit 0 {
            family ethernet-switching {
                port-mode trunk;
                vlan {
                    members pvlan100;
                    members pvlan400;
                }
            }
        }
    }
    xe-0/0/2 {
        unit 0 {
            family ethernet-switching {
                port-mode trunk;
            }
        }
    }
    xe-0/0/3 {
        unit 0 {
            family ethernet-switching {
                port-mode access;
            }
        }
    }
    xe-0/0/5 {
        unit 0 {
            family ethernet-switching {
                port-mode access;
            }
        }
    }
    xe-0/0/6 {
        unit 0 {
            family ethernet-switching {
                port-mode access;
            }
        }
    }
vlans {
    comm300 {
        vlan-id 300;
```

```
            interface {
            xe-0/0/3.0;
        }
        primary-vlan pvlan100;
    }
    comm600 {
        vlan-id 600;
            interface {
            xe-0/0/6.0;
        }
        primary-vlan pvlan400;
    }
    pvlan100 {
        vlan-id 100;
        interface {
            xe-0/0/0.0;
            xe-0/0/2.0;
            xe-0/0/3.0;
            xe-0/0/1.0 {
                pvlan-trunk;
            }
        }
        no-local-switching;
        isolation-id 200;
    }
    pvlan400 {
        vlan-id 400;
        interface {
            xe-0/0/5.0;
            xe-0/0/6.0;
            xe-0/0/1.0 {
                pvlan-trunk;
            }
        }
        no-local-switching;
        isolation-id 500;
    }
}
```

Verification

IN THIS SECTION

To confirm that the configuration is working properly, perform these tasks:

Verifying That the Private VLAN and Secondary VLANs Were Created

Purpose

Verify that the primary VLAN and secondary VLANs were properly created on Switch 1.

Action

Use the `show vlans` command:

```
user@switch>  show vlans private-vlan

Name               Role        Tag      Interfaces
pvlan100           Primary     100         xe-0/0/0.0, xe-0/0/1.0, xe-0/0/2.0, xe-0/0/3.0
__iso_pvlan100__   Isolated    200      xe-0/0/2.0
comm300            Community   300      xe-0/0/3.0

pvlan400           Primary     400      xe-0/0/0.0, xe-0/0/1.0, xe-0/0/5.0, xe-0/0/6.0
__iso_pvlan400__   Isolated    500      xe-0/0/5.0
comm600            Community   600     xe-0/0/6.0
```

Meaning

The output shows that the private VLANs were created and identifies the interfaces and secondary VLANs associated with them.

Verifying The Ethernet Switching Table Entries

Purpose

Verify that the Ethernet switching table entries were created for primary VLAN pvlan100.

Action

Show the Ethernet switching table entries for pvlan100.

```
user@switch> show ethernet-switching table vlan pvlan100 private-vlan
Ethernet-switching table: 0 unicast entries
  pvlan100             *                 Flood          - All-members
  pvlan100             00:10:94:00:00:02 Learn            xe-0/0/2.0
  __iso_pvlan100__     *                 Flood          - All-members
  __iso_pvlan100__     00:10:94:00:00:02 Replicated      - xe-0/0/2.0
```

SEE ALSO

Understanding Egress Firewall Filters with PVLANs

Verifying That a Private VLAN Is Working on a Switch

IN THIS SECTION

Purpose

After creating and configuring private VLANs (PVLANs), verify that they are set up properly.

Action

1. To determine whether you successfully created the primary and secondary VLAN configurations:

   - For a PVLAN on a single switch, use the `show configuration vlans` command:

```
user@switch> show configuration vlans
community1 {
    interface {
        interface a;
        interface b;
    }
    primary-vlan pvlan;
}
community2 {
    interface {
        interface d;
        interface e;
    }
    primary-vlan pvlan;
}
pvlan {
    vlan-id 1000;
    interface {
        isolated1;
        isolated2;
        trunk1;
        trunk2;
    }
    no-local-switching;
}
```

   - For a PVLAN spanning multiple switches, use the **show vlans** `extensive` command:

```
user@switch> show vlans extensive
VLAN: COM1, Created at: Tue May 11 18:16:05 2010
802.1Q Tag: 100, Internal index: 3, Admin State: Enabled, Origin: Static
Private VLAN Mode: Community, Primary VLAN: primary
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 3 (Active = 3), Untagged  1 (Active = 1)
      ge-0/0/20.0*, tagged, trunk
```

```
      ge-0/0/22.0*, tagged, trunk, pvlan-trunk
      ge-0/0/23.0*, tagged, trunk, pvlan-trunk
      ge-0/0/7.0*, untagged, access

VLAN: __pvlan_primary_ge-0/0/0.0__, Created at: Tue May 11 18:16:05 2010
Internal index: 5, Admin State: Enabled, Origin: Static
Private VLAN Mode: Isolated, Primary VLAN: primary
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 3 (Active = 3), Untagged  1 (Active = 1)
      ge-0/0/20.0*, tagged, trunk
      ge-0/0/22.0*, tagged, trunk, pvlan-trunk
      ge-0/0/23.0*, tagged, trunk, pvlan-trunk
      ge-0/0/0.0*, untagged, access


VLAN: __pvlan_primary_ge-0/0/2.0__, Created at: Tue May 11 18:16:05 2010
Internal index: 6, Admin State: Enabled, Origin: Static
Private VLAN Mode: Isolated, Primary VLAN: primary
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 3 (Active = 3), Untagged  1 (Active = 0)
      ge-0/0/20.0*, tagged, trunk
      ge-0/0/22.0*, tagged, trunk, pvlan-trunk
      ge-0/0/23.0*, tagged, trunk, pvlan-trunk
      ge-0/0/2.0, untagged, access


VLAN: __pvlan_primary_isiv__, Created at: Tue May 11 18:16:05 2010
802.1Q Tag: 50, Internal index: 7, Admin State: Enabled, Origin: Static
Private VLAN Mode: Inter-switch-isolated, Primary VLAN: primary
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 3 (Active = 3), Untagged  0 (Active = 0)
      ge-0/0/20.0*, tagged, trunk
      ge-0/0/22.0*, tagged, trunk, pvlan-trunk
      ge-0/0/23.0*, tagged, trunk, pvlan-trunk


VLAN: community2, Created at: Tue May 11 18:16:05 2010
802.1Q Tag: 20, Internal index: 8, Admin State: Enabled, Origin: Static
Private VLAN Mode: Community, Primary VLAN: primary
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 3 (Active = 3), Untagged  2 (Active = 2)
      ge-0/0/20.0*, tagged, trunk
      ge-0/0/22.0*, tagged, trunk, pvlan-trunk
```

```
      ge-0/0/23.0*, tagged, trunk, pvlan-trunk
      ge-0/0/1.0*, untagged, access
      ge-1/0/6.0*, untagged, access


VLAN: primary, Created at: Tue May 11 18:16:05 2010
802.1Q Tag: 10, Internal index: 2, Admin State: Enabled, Origin: Static
Private VLAN Mode: Primary
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 3 (Active = 3), Untagged  5 (Active = 4)
      ge-0/0/20.0*, tagged, trunk
      ge-0/0/22.0*, tagged, trunk, pvlan-trunk
      ge-0/0/23.0*, tagged, trunk, pvlan-trunk
      ge-0/0/0.0*, untagged, access
      ge-0/0/1.0*, untagged, access
      ge-0/0/2.0, untagged, access
      ge-0/0/7.0*, untagged, access
      ge-1/0/6.0*, untagged, access

Secondary VLANs: Isolated 2, Community  2, Inter-switch-isolated  1
  Isolated VLANs :
      __pvlan_primary_ge-0/0/0.0__
      __pvlan_primary_ge-0/0/2.0__
  Community VLANs :
      COM1
      community2
  Inter-switch-isolated VLAN :
      __pvlan_primary_isiv__
```

2. Use the `show vlans extensive` command to view VLAN information and link status for a PVLAN on a single switch or for a PVLAN spanning multiple switches.

   - For a PVLAN on a single switch:

```
user@switch> show vlans pvlan extensive
VLAN: pvlan, Created at: time
802.1Q Tag: vlan-id, Internal index: index-number, Admin State: Enabled, Origin: Static
Private VLAN Mode: Primary
Protocol: Port Mode
Number of interfaces: Tagged 2 (Active = 0), Untagged  6 (Active = 0)
      trunk1, tagged, trunk
```

```
      interface a, untagged, access
      interface b, untagged, access
      interface c, untagged, access
      interface d, untagged, access
      interface e, untagged, access
      interface f, untagged, access
      trunk2, tagged, trunk
Secondary VLANs: Isolated 2, Community  2
  Isolated VLANs :
      __pvlan_pvlan_isolated1__
      __pvlan_pvlan_isolated2__
  Community VLANs :
      community1
      community2
```

- For a PVLAN spanning multiple switches:

```
user@switch> show vlans extensive
VLAN: COM1, Created at: Tue May 11 18:16:05 2010
802.1Q Tag: 100, Internal index: 3, Admin State: Enabled, Origin: Static
Private VLAN Mode: Community, Primary VLAN: primary
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 3 (Active = 3), Untagged  1 (Active = 1)
      ge-0/0/20.0*, tagged, trunk
      ge-0/0/22.0*, tagged, trunk, pvlan-trunk
      ge-0/0/23.0*, tagged, trunk, pvlan-trunk
      ge-0/0/7.0*, untagged, access

VLAN: __pvlan_primary_ge-0/0/0.0__, Created at: Tue May 11 18:16:05 2010
Internal index: 5, Admin State: Enabled, Origin: Static
Private VLAN Mode: Isolated, Primary VLAN: primary
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 3 (Active = 3), Untagged  1 (Active = 1)
      ge-0/0/20.0*, tagged, trunk
      ge-0/0/22.0*, tagged, trunk, pvlan-trunk
      ge-0/0/23.0*, tagged, trunk, pvlan-trunk
      ge-0/0/0.0*, untagged, access


VLAN: __pvlan_primary_ge-0/0/2.0__, Created at: Tue May 11 18:16:05 2010
Internal index: 6, Admin State: Enabled, Origin: Static
Private VLAN Mode: Isolated, Primary VLAN: primary
```

```
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 3 (Active = 3), Untagged  1 (Active = 0)
      ge-0/0/20.0*, tagged, trunk
      ge-0/0/22.0*, tagged, trunk, pvlan-trunk
      ge-0/0/23.0*, tagged, trunk, pvlan-trunk
      ge-0/0/2.0, untagged, access


VLAN: __pvlan_primary_isiv__, Created at: Tue May 11 18:16:05 2010
802.1Q Tag: 50, Internal index: 7, Admin State: Enabled, Origin: Static
Private VLAN Mode: Inter-switch-isolated, Primary VLAN: primary
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 3 (Active = 3), Untagged  0 (Active = 0)
      ge-0/0/20.0*, tagged, trunk
      ge-0/0/22.0*, tagged, trunk, pvlan-trunk
      ge-0/0/23.0*, tagged, trunk, pvlan-trunk


VLAN: community2, Created at: Tue May 11 18:16:05 2010
802.1Q Tag: 20, Internal index: 8, Admin State: Enabled, Origin: Static
Private VLAN Mode: Community, Primary VLAN: primary
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 3 (Active = 3), Untagged  2 (Active = 2)
      ge-0/0/20.0*, tagged, trunk
      ge-0/0/22.0*, tagged, trunk, pvlan-trunk
      ge-0/0/23.0*, tagged, trunk, pvlan-trunk
      ge-0/0/1.0*, untagged, access
      ge-1/0/6.0*, untagged, access


VLAN: primary, Created at: Tue May 11 18:16:05 2010
802.1Q Tag: 10, Internal index: 2, Admin State: Enabled, Origin: Static
Private VLAN Mode: Primary
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 3 (Active = 3), Untagged  5 (Active = 4)
      ge-0/0/20.0*, tagged, trunk
      ge-0/0/22.0*, tagged, trunk, pvlan-trunk
      ge-0/0/23.0*, tagged, trunk, pvlan-trunk
      ge-0/0/0.0*, untagged, access
      ge-0/0/1.0*, untagged, access
      ge-0/0/2.0, untagged, access
      ge-0/0/7.0*, untagged, access
      ge-1/0/6.0*, untagged, access
```

```
Secondary VLANs: Isolated 2, Community  2, Inter-switch-isolated  1
  Isolated VLANs :
      __pvlan_primary_ge-0/0/0.0__
      __pvlan_primary_ge-0/0/2.0__
  Community VLANs :
      COM1
      community2
  Inter-switch-isolated VLAN :
      __pvlan_primary_isiv__
```

3. Use the `show ethernet-switching table` command to view logs for MAC learning on the VLANs:

```
user@switch> show ethernet-switching table
Ethernet-switching table: 8 entries, 1 learned

  VLAN              MAC address       Type         Age Interfaces

  default           *                 Flood          - All-members

  pvlan             *                 Flood          - All-members

  pvlan             MAC1              Replicated     - interface a

  pvlan             MAC2              Replicated     - interface c

  pvlan             MAC3              Replicated     - isolated2

  pvlan             MAC4              Learn          0 trunk1

  __pvlan_pvlan_isolated1__ *         Flood          - All-members

  __pvlan_pvlan_isolated1__ MAC4      Replicated     - trunk1

  __pvlan_pvlan_isolated2__ *         Flood          - All-members

  __pvlan_pvlan_isolated2__ MAC3      Learn          0 isolated2

  __pvlan_pvlan_isolated2__ MAC4      Replicated     - trunk1

  community1        *                 Flood          - All-members
```

```
community1        MAC1              Learn          0 interface a

community1        MAC4              Replicated     - trunk1

community2        *                 Flood          - All-members

community2        MAC2              Learn          0 interface c

community2        MAC4              Replicated     - trunk1
```

NOTE: If you have configured a PVLAN spanning multiple switches, you can use the same command on all the switches to check the logs for MAC learning on those switches.

Meaning

In the output displays for a PVLAN on a single switch, you can see that the primary VLAN contains two community domains (**community1** and **community2**), two isolated ports, and two trunk ports. The PVLAN on a single switch has only one tag (**1000**), which is for the primary VLAN.

The PVLAN that spans multiple switches contains multiple tags:

- The community domain **COM1** is identified with tag **100**.
- The community domain **community2** is identified with tag **20**.
- The interswitch isolated domain is identified with tag **50**.
- The primary VLAN **primary** is identified with tag **10**.

Also, for the PVLAN that spans multiple switches, the trunk interfaces are identified as **pvlan-trunk**.

Troubleshooting Private VLANs on QFX Switches

IN THIS SECTION

Use the following information to troubleshoot a private VLAN configuration.

Limitations of Private VLANs

The following constraints apply to private VLAN configurations:

- IGMP snooping is not supported with private VLANs.
- Routed VLAN interfaces are not supported on private VLANs
- Routing between secondary VLANs in the same primary VLAN is not supported.
- If you want to change a primary VLAN to be a secondary VLAN, you must first change it to a normal VLAN and commit the change. For example, you would follow this procedure:

  1. Change the primary VLAN to be a normal VLAN.
  2. Commit the configuration.
  3. Change the normal VLAN to be a secondary VLAN.
  4. Commit the configuration.

  Follow the same sequence of commits if you want to change a secondary VLAN to be a primary VLAN. That is, make the secondary VLAN a normal VLAN and commit that change and then change the normal VLAN to be a primary VLAN.

Forwarding with Private VLANs

IN THIS SECTION

Problem

Description

- When isolated VLAN or community VLAN tagged traffic is received on a PVLAN trunk port, MAC addresses are learned from the primary VLAN. This means that output from the *show ethernet-switching table* command shows that MAC addresses are learned from the primary VLAN and replicated to secondary VLANs. This behavior has no effect on forwarding decisions.
- If a packet with a secondary VLAN tag is received on a promiscuous port, it is accepted and forwarded.
- If a packet is received on a PVLAN trunk port and meets both of the conditions listed below, it is dropped.
  - The packet has a community VLAN tag.
  - The packet is destined to a unicast MAC address or multicast group MAC address that was learned on an isolated VLAN.
- If a packet is received on a PVLAN trunk port and meets both of the conditions listed below, it is dropped.
  - The packet has an isolated VLAN tag.
  - The packet is destined to a unicast MAC address or multicast group MAC address that was learned on a community VLAN.
- If a packet with a primary VLAN tag is received by a secondary (isolated or community) VLAN port, the secondary port forwards the packet.
- If you configure a community VLAN on one device and configure another community VLAN on a second device and both community VLANs use the same VLAN ID, traffic for one of the VLANs can be forwarded to the other VLAN. For example, assume the following configuration:

- Community VLAN comm1 on switch 1 has VLAN ID 50 and is a member of primary VLAN pvlan100.
- Community VLAN comm2 on switch 2 also has VLAN ID 50 and is a member of primary VLAN pvlan200.
- Primary VLAN pvlan100 exists on both switches.

If traffic for comm1 is sent from switch 1 to switch 2, it will be sent to the ports participating in comm2. (The traffic will also be forwarded to the ports in comm1, as you would expect.)

Solution

These are expected behaviors.

Egress Firewall Filters with Private VLANs

IN THIS SECTION

Problem

Description

If you apply a firewall filter in the output direction to a primary VLAN, the filter also applies to the secondary VLANs that are members of the primary VLAN when the traffic egresses with the primary VLAN tag or isolated VLAN tag, as listed below:

- Traffic forwarded from a secondary VLAN trunk port to a promiscuous port (trunk or access)
- Traffic forwarded from a secondary VLAN trunk port that carries an isolated VLAN to a PVLAN trunk port.
- Traffic forwarded from a promiscuous port (trunk or access) to a secondary VLAN trunk port
- Traffic forwarded from a PVLAN trunk port. to a secondary VLAN trunk port
- Traffic forwarded from a community port to a promiscuous port (trunk or access)

If you apply a firewall filter in the output direction to a primary VLAN, the filter does *not* apply to traffic that egresses with a community VLAN tag, as listed below:

- Traffic forwarded from a community trunk port to a PVLAN trunk port
- Traffic forwarded from a secondary VLAN trunk port that carries a community VLAN to a PVLAN trunk port
- Traffic forwarded from a promiscuous port (trunk or access) to a community trunk port
- Traffic forwarded from a PVLAN trunk port. to a community trunk port

If you apply a firewall filter in the output direction to a community VLAN, the following behaviors apply:

- The filter is applied to traffic forwarded from a promiscuous port (trunk or access) to a community trunk port (because the traffic egresses with the community VLAN tag).
- The filter is applied to traffic forwarded from a community port to a PVLAN trunk port (because the traffic egresses with the community VLAN tag).
- The filter is *not* applied to traffic forwarded from a community port to a promiscuous port (because the traffic egresses with the primary VLAN tag or untagged).

Solution

These are expected behaviors. They occur only if you apply a firewall filter to a private VLAN in the output direction and do not occur if you apply a firewall filter to a private VLAN in the input direction.

Egress Port Mirroring with Private VLANs

IN THIS SECTION

Problem

Description

If you create a port-mirroring configuration that mirrors private VLAN (PVLAN) traffic on egress, the mirrored traffic (the traffic that is sent to the analyzer system) has the VLAN tag of the ingress VLAN instead of the egress VLAN. For example, assume the following PVLAN configuration:

- Promiscuous trunk port that carries primary VLANs pvlan100 and pvlan400.
- Isolated access port that carries secondary VLAN isolated200. This VLAN is a member of primary VLAN pvlan100.
- Community port that carries secondary VLAN comm300. This VLAN is also a member of primary VLAN pvlan100.
- Output interface (monitor interface) that connects to the analyzer system. This interface forwards the mirrored traffic to the analyzer.

If a packet for pvlan100 enters on the promiscuous trunk port and exits on the isolated access port, the original packet is untagged on egress because it is exiting on an access port. However, the mirror copy retains the tag for pvlan100 when it is sent to the analyzer.

Here is another example: If a packet for comm300 ingresses on the community port and egresses on the promiscuous trunk port, the original packet carries the tag for pvlan100 on egress, as expected. However, the mirrored copy retains the tag for comm300 when it is sent to the analyzer.

Solution

This is expected behavior.

Understanding Private VLANs

IN THIS SECTION

VLANs limit broadcasts to specified users. Private VLANs (PVLANs) take this concept a step further by limiting communication within a VLAN. PVLANs accomplish this by restricting traffic flows through their member switch ports (which are called *private ports*) so that these ports communicate only with a specified uplink trunk port or with specified ports within the same VLAN. The uplink trunk port or link aggregation group (LAG) is usually connected to a router, firewall, server, or provider network. Each PVLAN typically contains many private ports that communicate only with a single uplink port, thereby preventing the ports from communicating with each other.

PVLANs provide Layer 2 isolation between ports within a VLAN, splitting a broadcast domain into multiple discrete broadcast subdomains by creating secondary VLANs (*community* VLANs and an *isolated* VLAN) inside a primary VLAN. Ports within the same community VLAN can communicate with each other. Ports within an isolated VLAN can communicate *only* with a single uplink port.

Just like regular VLANs, PVLANs are isolated on Layer 2 and require one of the following options to route Layer 3 traffic among the secondary VLANs:

- A promiscuous port connection with a router
- A routed VLAN interface (RVI)

NOTE: To route Layer 3 traffic among secondary VLANs, a PVLAN needs only one of the options mentioned above. If you use an RVI, you can still implement a promiscuous port connection to a router with the promiscuous port set up to handle only traffic that enters and exits the PVLAN.

PVLANs are useful for restricting the flow of broadcast and unknown unicast traffic and for limiting the communication between known hosts. Service providers use PVLANs to keep their customers isolated from each other. Another typical use for a PVLAN is to provide per-room Internet access in a hotel.

NOTE: You can configure a PVLAN to span switches that support PVLANs.

This topic explains the following concepts regarding PVLANs on EX Series switches:

Benefits of PVLANs

The need to segregate a single VLAN is particularly useful in the following deployment scenarios:

- Server farms—A typical Internet service provider uses a server farm to provide Web hosting for numerous customers. Locating the various servers within a single server farm provides ease of management. Security concerns arise if all servers are in the same VLAN because Layer 2 broadcasts go to all servers in the VLAN.

- Metropolitan Ethernet networks—A metro service provider offers Layer 2 Ethernet access to assorted homes, rental communities, and businesses. The traditional solution of deploying one VLAN per customer is not scalable and is difficult to manage, leading to potential waste of IP addresses. PVLANs provide a more secure and more efficient solution.

Typical Structure and Primary Application of PVLANs

A PVLAN can be configured on a single switch or can be configured to span multiple switches. The types of domains and ports are:

- Primary VLAN—The primary VLAN of the PVLAN is defined with an 802.1Q tag (VLAN ID) for the complete PVLAN. The primary PVLAN can contain multiple secondary VLANs (one isolated VLAN and multiple community VLANs).

- Isolated VLAN/isolated port—A primary VLAN can contain only one isolated VLAN. An interface within an isolated VLAN can forward packets only to a promiscuous port or the Inter-Switch Link (ISL) port. An isolated interface cannot forward packets to another isolated interface; and an isolated interface cannot receive packets from another isolated interface. If a customer device needs to have access *only* to a gateway router, the device must be attached to an isolated trunk port.

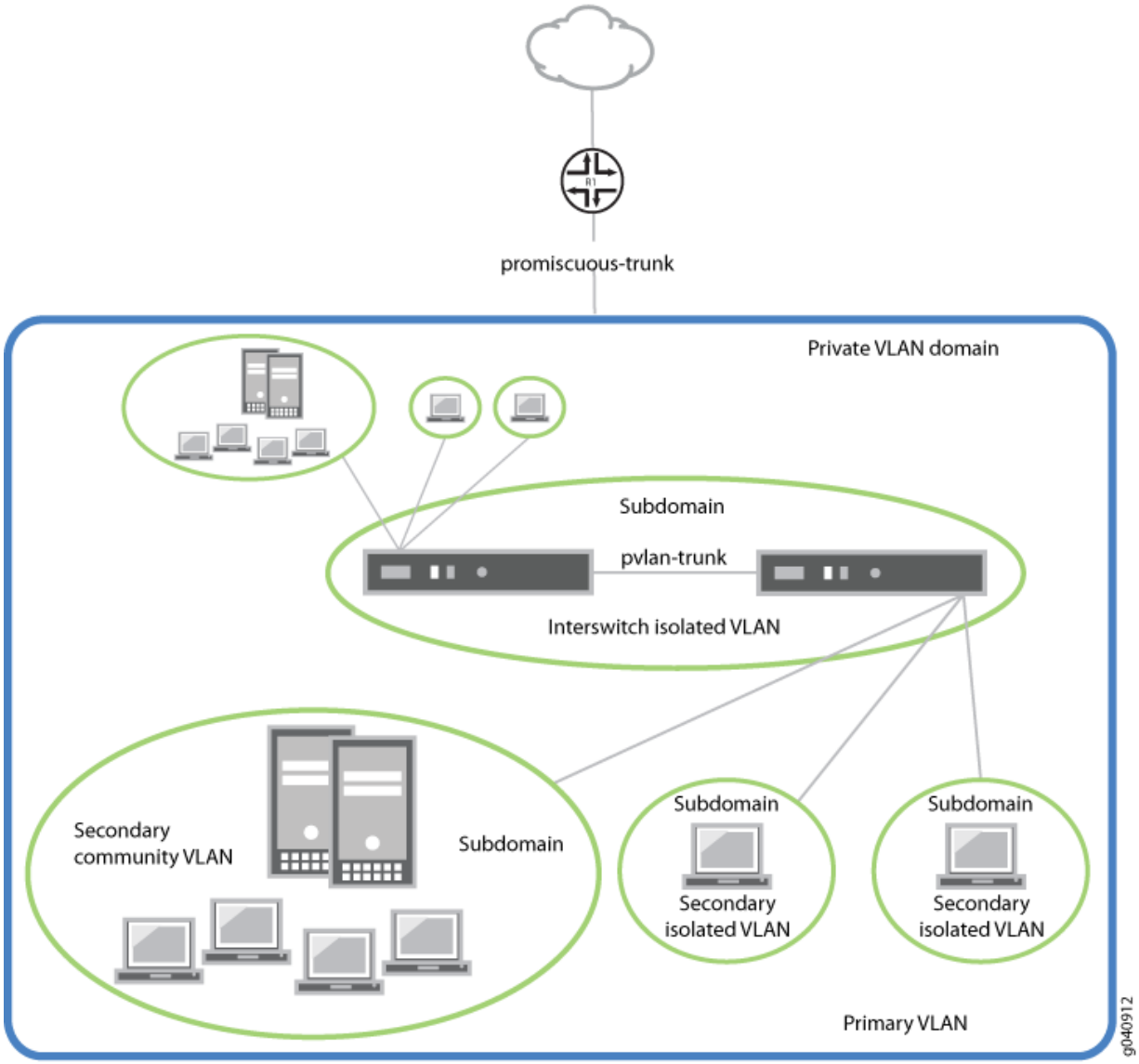
- Community VLAN/community port—You can configure multiple community VLANs within a single PVLAN. An interface within a specific community VLAN can establish Layer 2 communications with any other interface that belongs to the same community VLAN. An interface within a community VLAN can also communicate with a promiscuous port or the ISL port. If you have, for example, two customer devices that you need to isolate from other customer devices but that must be able to communicate with one another, use community ports.

- Promiscuous port—A promiscuous port has Layer 2 communications with all interfaces in the PVLAN, regardless of whether an interface belongs to an isolated VLAN or a community VLAN. A promiscuous port is a member of the primary VLAN but is not included within any secondary subdomain. Layer 3 gateways, DHCP servers, and other trusted devices that need to communicate with endpoint devices are typically connected to a promiscuous port.

- Inter-Switch Link (ISL)—An ISL is a trunk port that connects multiple switches in a PVLAN and contains two or more VLANs. It is required only when a PVLAN spans multiple switches.

The configured PVLAN is the *primary* domain (primary VLAN). Within the PVLAN, you configure *secondary* VLANs, which become subdomains nested within the primary domain. A PVLAN can be configured on a single switch or can be configured to span multiple switches. The PVLAN shown in Figure 29 on page 617 includes two switches, with a primary PVLAN domain and various subdomains.

Figure 29: Subdomains in a PVLAN



As shown in Figure 31 on page 620, a PVLAN has only one primary domain and multiple secondary domains. The types of domains are:

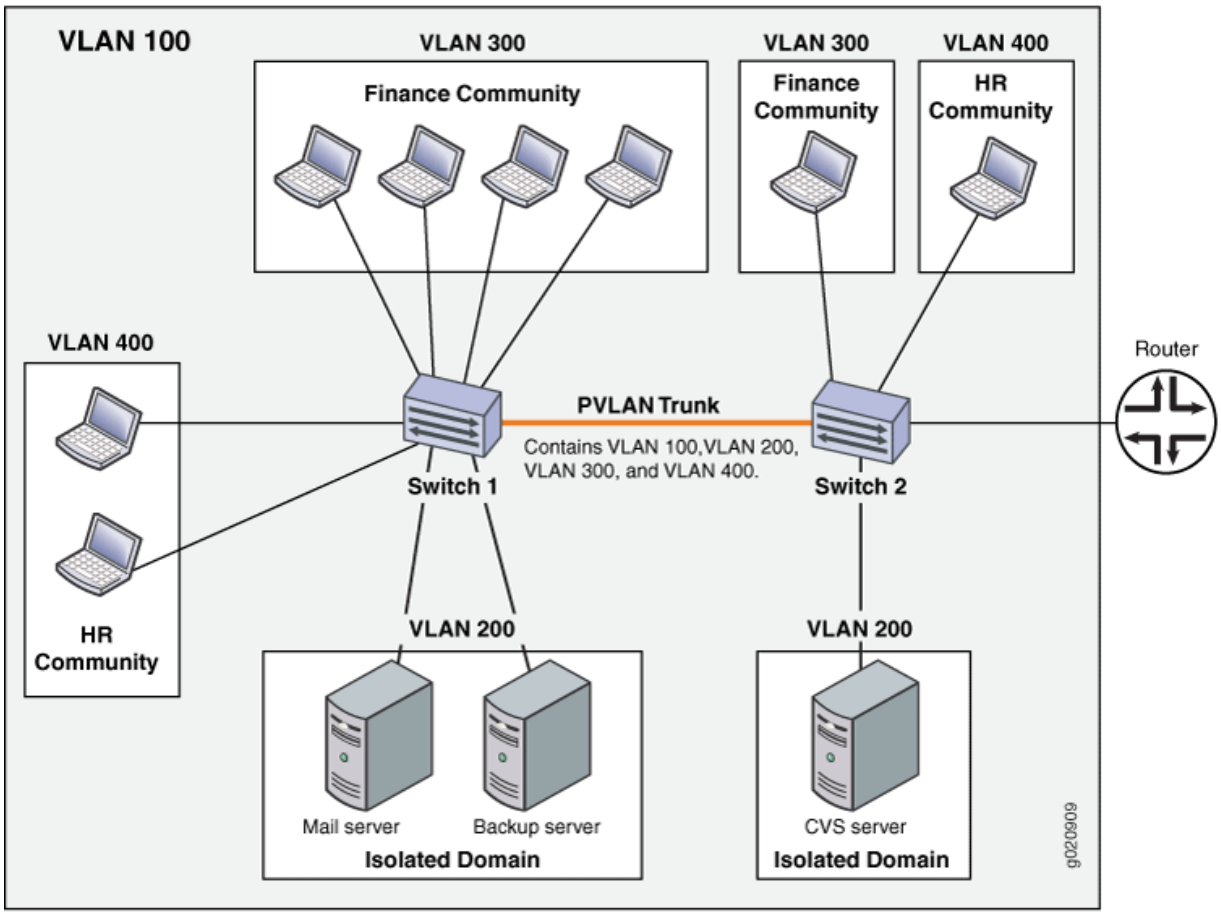
- Primary VLAN—VLAN used to forward frames downstream to isolated and community VLANs. The primary VLAN of the PVLAN is defined with an 802.1Q tag (VLAN ID) for the complete PVLAN. The primary PVLAN can contain multiple secondary VLANs (one isolated VLAN and multiple community VLANs).
- Secondary isolated VLAN—VLAN that receives packets only from the primary VLAN and forwards frames upstream to the primary VLAN. The isolated VLAN is a secondary VLAN nested within the primary VLAN. A primary VLAN can contain only one isolated VLAN. An interface within an isolated VLAN (isolated interface) can forward packets only to a promiscuous port or the PVLAN trunk port.

An isolated interface cannot forward packets to another isolated interface; nor can an isolated interface receive packets from another isolated interface. If a customer device needs to have access *only* to a router, the device must be attached to an isolated trunk port.

- Secondary interswitch isolated VLAN—VLAN used to forward isolated VLAN traffic from one switch to another through PVLAN trunk ports. 802.1Q tags are required for interswitch isolated VLANs because IEEE 802.1Q uses an internal tagging mechanism by which a trunking device inserts a 4-byte VLAN frame identification tab into the packet header. An interswitch isolated VLAN is a secondary VLAN nested within the primary VLAN.
- Secondary community VLAN—VLAN used to transport frames among members of a community (a subset of users within the VLAN) and to forward frames upstream to the primary VLAN. A community VLAN is a secondary VLAN nested within the primary VLAN. You can configure multiple community VLANs within a single PVLAN. An interface within a specific community VLAN can establish Layer 2 communications with any other interface that belongs to the same community VLAN. An interface within a community VLAN can also communicate with a promiscuous port or the PVLAN trunk port.

Figure 30 on page 619 shows a PVLAN spanning multiple switches, where the primary VLAN (`100`) contains two community domains (`300` and `400`) and one interswitch isolated domain.

Figure 30: PVLAN Spanning Multiple Switches

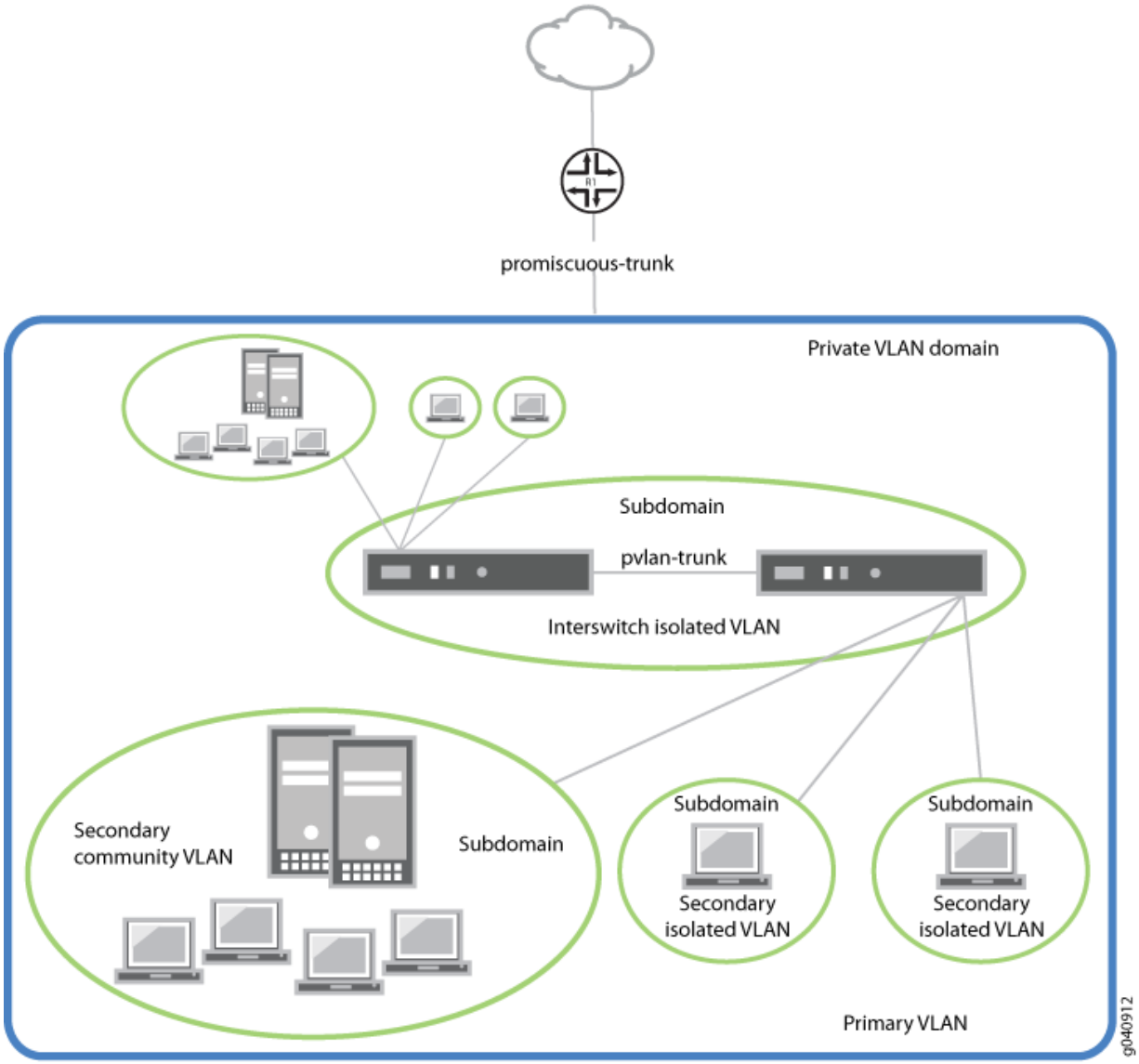


NOTE: Primary and secondary VLANs count against the limit of 4089 VLANs supported on the QFX Series. For example, each VLAN in Figure 30 on page 619 counts against this limit.

Typical Structure and Primary Application of PVLANs on MX Series Routers

The configured PVLAN becomes the primary domain, and secondary VLANs become subdomains that are nested inside the primary domain. A PVLAN can be created on a single router. The PVLAN shown in Figure 31 on page 620 includes one router, with one primary PVLAN domain and multiple secondary subdomains.

Figure 31: Subdomains in a PVLAN With One Router



The types of domains are:

- Primary VLAN—VLAN used to forward frames downstream to isolated and community VLANs.
- Secondary isolated VLAN—VLAN that receives packets only from the primary VLAN and forwards frames upstream to the primary VLAN.
- Secondary interswitch isolated VLAN—VLAN used to forward isolated VLAN traffic from one router to another through PVLAN trunk ports.
- Secondary community VLAN—VLAN used to transport frames among members of a community, which is a subset of users within the VLAN, and to forward frames upstream to the primary VLAN.

NOTE: PVLANs are supported on MX80 routers, on MX240, MX480, and MX960 routers with DPCs in enhanced LAN mode, on MX Series routers with MPC1, MPC2, and Adaptive Services PICs.

Typical Structure and Primary Application of PVLANs on EX Series Switches

NOTE: The primary VLAN of the PVLAN is defined with an 802.1Q tag (VLAN ID) for the complete PVLAN. On EX9200 switches, each secondary VLAN must also be defined with its own separate VLAN ID.

Figure 32 on page 622 shows a PVLAN on a single switch, where the primary VLAN (VLAN `100`) contains two community VLANs (VLAN `300` and VLAN `400`) and one isolated VLAN (VLAN `50`).

Figure 32: Private VLAN on a Single EX Switch

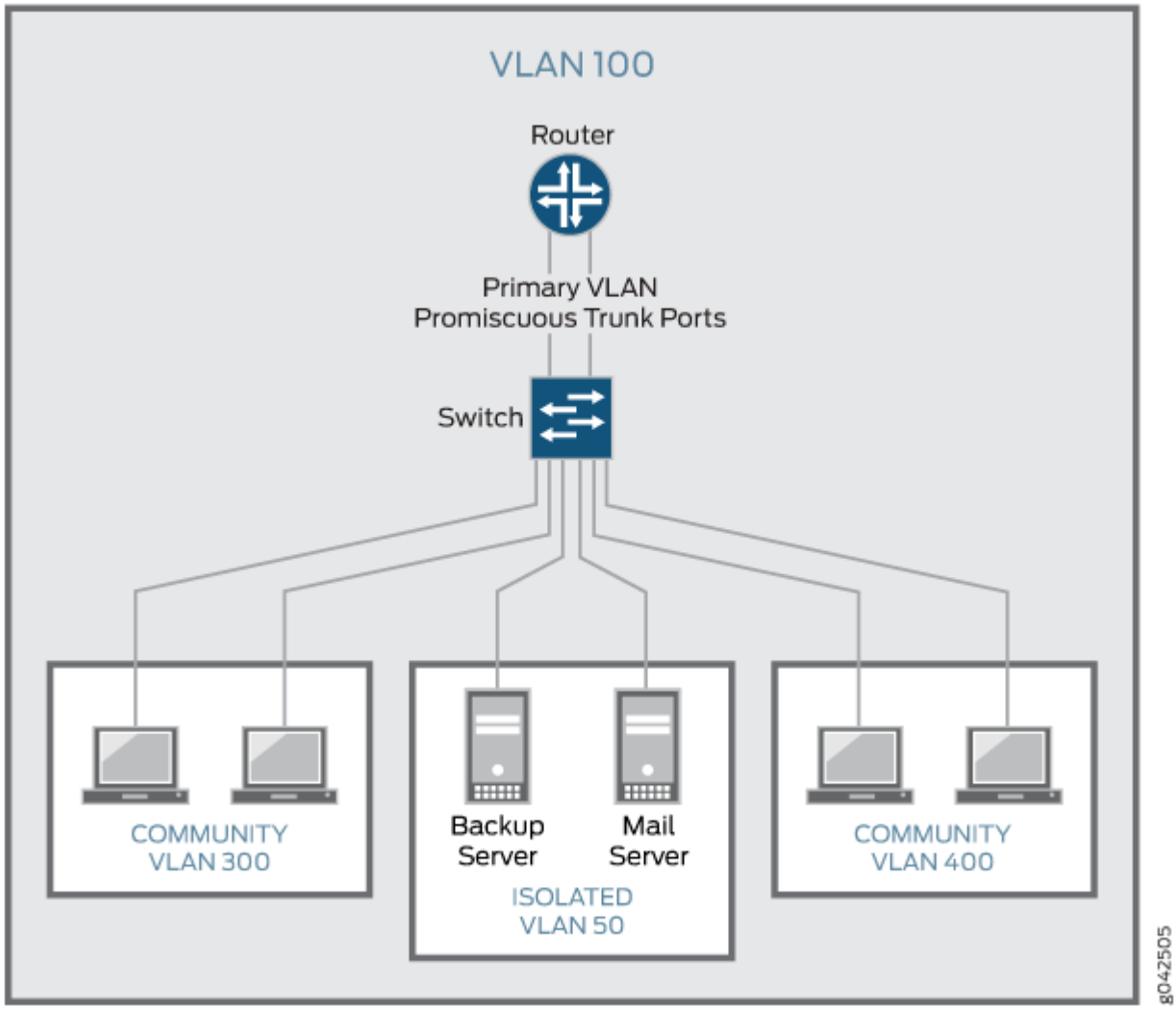
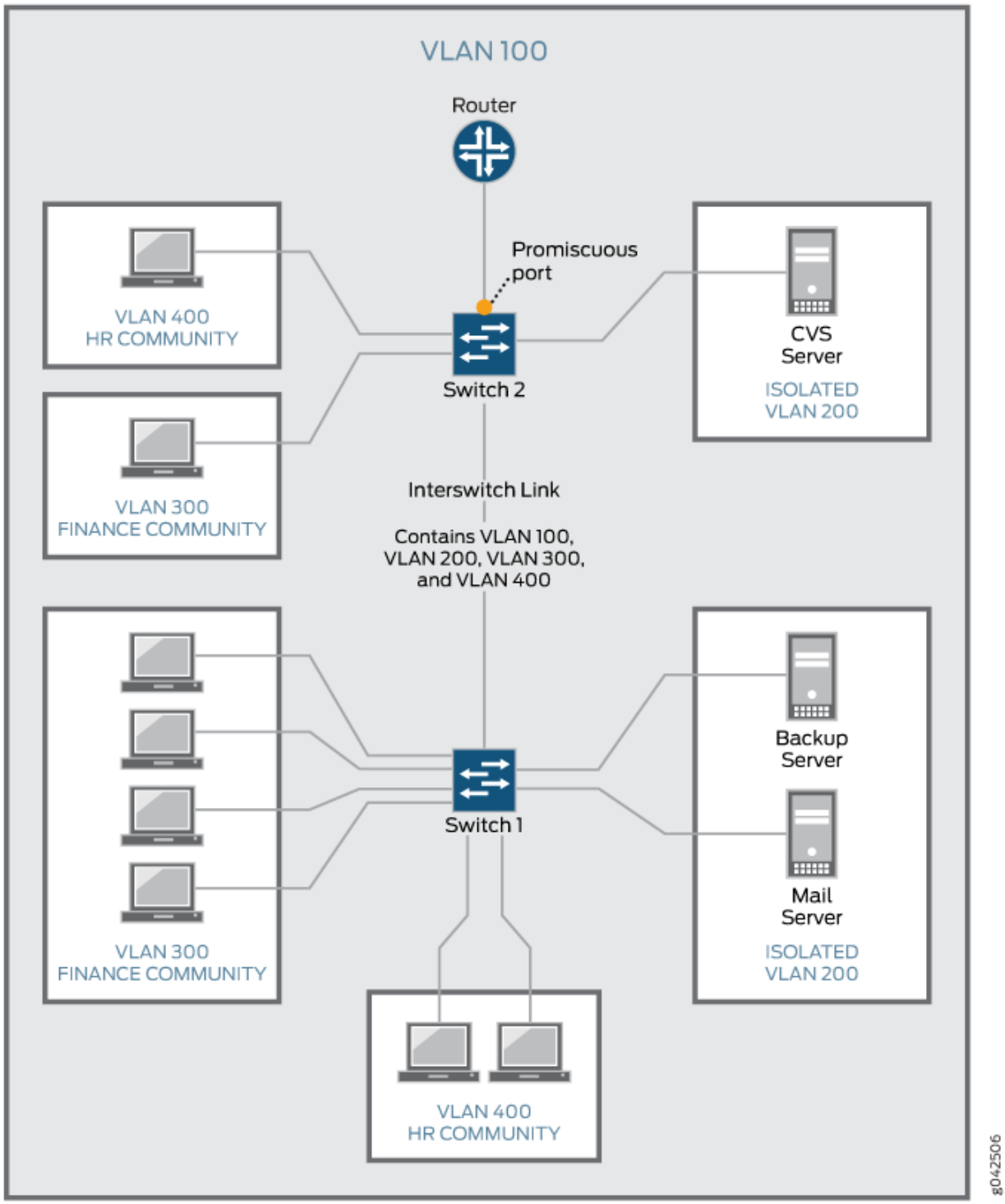


Figure 33 on page 623 shows a PVLAN spanning multiple switches, where the primary VLAN (VLAN `100`) contains two community VLANs (VLAN `300` and VLAN `400`) and one isolated VLAN (VLAN 200). It also shows that Switches 1 and 2 are connected through an interswitch link (PVLAN trunk link).

Figure 33: PVLAN Spanning Multiple EX Series Switches



Also, the PVLANs shown in Figure 32 on page 622 and Figure 33 on page 623 use a promiscuous port connected to a router as the means to route Layer 3 traffic among the community and isolated VLANs. Instead of using the promiscuous port connected to a router, you can configure an RVI on the switch in Figure 32 on page 622 or one of the switches shown in Figure 33 on page 623 (on some EX switches).

To route Layer 3 traffic between isolated and community VLANs, you must either connect a router to a promiscuous port, as shown in Figure 32 on page 622 and Figure 33 on page 623, or configure an RVI.

If you choose the RVI option, you must configure one RVI for the primary VLAN in the PVLAN domain. This RVI serves the entire PVLAN domain regardless of whether the domain includes one or more switches. After you configure the RVI, Layer 3 packets received by the secondary VLAN interfaces are mapped to and routed by the RVI.

When setting up the RVI, you must also enable proxy Address Resolution Protocol (ARP) so that the RVI can handle ARP requests received by the secondary VLAN interfaces.

For information about configuring PVLANs on a single switch and on multiple switches, see "Creating a Private VLAN on a Single EX Series Switch (CLI Procedure)" on page 485. For information about configuring an RVI, see "Configuring a Routed VLAN Interface in a Private VLAN on an EX Series Switch" on page 759.

Routing Between Isolated and Community VLANs

To route Layer 3 traffic between isolated and community VLANs, you must connect an external router or switch to a trunk port of the primary VLAN. The trunk port of the primary VLAN is a *promiscuous* port; therefore, it can communicate with *all* the ports in the PVLAN.

PVLANs Use 802.1Q Tags to Identify Packets

When packets are marked with a customer-specific 802.1Q tag, that tag identifies ownership of the packets for any switch or router in the network. Sometimes, 802.1Q tags are needed within PVLANs to keep track of packets from different subdomains. Table 94 on page 625 indicates when a VLAN 802.1Q tag is needed on the primary VLAN or on secondary VLANs.

Table 94: When VLANs in a PVLAN Need 802.1Q Tags

	On a Single Switch	On Multiple Switches
Primary VLAN	Specify an 802.1Q tag by setting a VLAN ID.	Specify an 802.1Q tag by setting a VLAN ID.
Secondary VLAN	No tag needed on VLANs.	VLANs need 802.1Q tags: • Specify an 802.1Q tag for each community VLAN by setting a VLAN ID. • Specify the 802.1Q tag for an isolation VLAN ID by setting an isolation ID.

PVLANs Use IP Addresses Efficiently

PVLANs provide IP address conservation and efficient allocation of IP addresses. In a typical network, VLANs usually correspond to a single IP subnet. In PVLANs, the hosts in all secondary VLANs belong to the same IP subnet because the subnet is allocated to the primary VLAN. Hosts within the secondary VLAN are assigned IP addresses based on IP subnets associated with the primary VLAN, and their IP subnet masking information reflects that of the primary VLAN subnet. However, each secondary VLAN is a separate broadcast domain.

PVLAN Port Types and Forwarding Rules

PVLANs can use up to six different port types. The network depicted inFigure 30 on page 619 uses a promiscuous port to transport information to the router, community ports to connect the finance and HR communities to their respective switches, isolated ports to connect the servers, and a PVLAN trunk port to connect the two switches. PVLAN ports have different restrictions:

- Promiscuous trunk port—A promiscuous port has Layer 2 communications with all the interfaces that are in the PVLAN, regardless of whether the interface belongs to an isolated VLAN or a community VLAN. A promiscuous port is a member of the primary VLAN, but is not included within one of the secondary subdomains. Layer 3 gateways, DHCP servers, and other trusted devices that need to communicate with endpoint devices are typically connected to a promiscuous port.

- PVLAN trunk link—The PVLAN trunk link, which is also known as the interswitch link, is required only when a PVLAN is configured to span multiple switches. The PVLAN trunk link connects the multiple switches that compose the PVLAN.

- PVLAN trunk port—A PVLAN trunk port is required in multiswitch PVLAN configurations to span the switches. The PVLAN trunk port is a member of all VLANs within the PVLAN (that is, the primary VLAN, the community VLANs, and the interswitch isolated VLAN), and it carries traffic from the primary VLAN and all secondary VLANs. It can communicate with all ports other than the isolated ports.

  Communication between a PVLAN trunk port and an isolated port is usually unidirectional. A PVLAN trunk port's membership in the interswitch isolated VLAN is egress-only, meaning that an isolated port can forward packets to a PVLAN trunk port, but a PVLAN trunk port does not forward packets to an isolated port (unless the packets ingressed on a promiscuous access port and are therefore being forwarded to all the secondary VLANs in the same primary VLAN as the promiscuous port).

- Secondary VLAN trunk port (not shown)—Secondary trunk ports carry secondary VLAN traffic. For a given private VLAN, a secondary VLAN trunk port can carry traffic for only one secondary VLAN. However, a secondary VLAN trunk port can carry traffic for multiple secondary VLANs as long as each secondary VLAN is a member of a different primary VLAN. For example, a secondary VLAN trunk port can carry traffic for a community VLAN that is part of primary VLAN pvlan100 and also carry traffic for an isolated VLAN that is part of primary VLAN pvlan400.

- Community port—Community ports communicate among themselves and with their promiscuous ports. Community ports serve only a select group of users. These interfaces are separated at Layer 2 from all other interfaces in other communities or isolated ports within their PVLAN.

- Isolated access port—Isolated ports have Layer 2 connectivity only with promiscuous ports and PVLAN trunk ports—an isolated port cannot communicate with another isolated port even if these two ports are members of the same isolated VLAN (or interswitch isolated VLAN) domain. Typically, a server, such as a mail server or a backup server, is connected on an isolated port. In a hotel, each room would typically be connected on an isolated port, meaning that room-to-room communication is not possible, but each room can access the Internet on the promiscuous port.

- Promiscuous access port (not shown)—These ports carry untagged traffic. Traffic that ingresses on a promiscuous access port is forwarded to all secondary VLAN ports on the device. If traffic ingresses into the device on a VLAN-enabled port and egresses on a promiscuous access port, the traffic is untagged on egress. If tagged traffic ingresses on a promiscuous access port, the traffic is discarded.

- Interswitch link port—An interswitch link (ISL) port is a trunk port that connects two routers when a PVLAN spans those routers. The ISL port is a member of all VLANs within the PVLAN (that is, the primary VLAN, the community VLANs, and the isolated VLAN).

  Communication between an ISL port and an isolated port is unidirectional. An ISL port's membership in the interswitch isolated VLAN is egress-only, meaning that incoming traffic on the ISL port is never assigned to the isolated VLAN. An isolated port can forward packets to a PVLAN trunk port, but a

PVLAN trunk port cannot forward packets to an isolated port. Table 96 on page 627 summarizes whether Layer 2 connectivity exists between the different types of ports.

Table 95 on page 627 summarizes Layer 2 connectivity between the different types of ports within a PVLAN on EX Series switches that support ELS.

Table 95: PVLAN Ports and Layer 2 Forwarding on EX Series switches that support ELS

From Port Type	To Isolated Ports?	To Promiscuous Ports?	To Community Ports?	To Inter-Switch Link Port?
Isolated	Deny	Permit	Deny	Permit
Promiscuous	Permit	Permit	Permit	Permit
Community 1	Deny	Permit	Permit	Permit

Table 96: PVLAN Ports and Layer 2 Connectivity

Port Type	Promiscuous Trunk	PVLAN Trunk	Secondary Trunk	Community	Isolated Access	Promiscuous access
Promiscuous trunk	Yes	Yes	Yes	Yes	Yes	Yes
PVLAN trunk	Yes	Yes	Yes	Yes—same community only	Yes	Yes
Secondary Trunk	Yes	Yes	No	Yes	No	Yes
Community	Yes	Yes	Yes	Yes—same community only	No	Yes
Isolated access	Yes	Yes—unidirectional only	No	No	No	Yes

Table 96: PVLAN Ports and Layer 2 Connectivity *(Continued)*

Port Type	Promiscuous Trunk	PVLAN Trunk	Secondary Trunk	Community	Isolated Access	Promiscuous access
Promiscuous access	Yes	Yes	Yes	Yes	Yes	No

Table 97 on page 628 summarizes whether or not Layer 2 connectivity exists between the different types of ports within a PVLAN.

Table 97: PVLAN Ports and Layer 2 Connectivity on EX Series Switches without ELS Support

Port Type To: → From:↓	Promiscuous	Community	Isolated	PVLAN Trunk	RVI
Promiscuous	Yes	Yes	Yes	Yes	Yes
Community	Yes	Yes—same community only	No	Yes	Yes
Isolated	Yes	No	No	Yes **NOTE**: This communication is unidirectional.	Yes
PVLAN trunk	Yes	Yes—same community only	Yes **NOTE**: This communication is unidirectional.	Yes	Yes
RVI	Yes	Yes	Yes	Yes	Yes

As noted in Table 97 on page 628, Layer 2 communication between an isolated port and a PVLAN trunk port is unidirectional. That is, an isolated port can only send packets to a PVLAN trunk port, and a PVLAN trunk port can only receive packets from an isolated port. Conversely, a PVLAN trunk port

cannot send packets to an isolated port, and an isolated port cannot receive packets from a PVLAN trunk port.

NOTE: If you enable `no-mac-learning` on a primary VLAN, all isolated VLANs (or the interswitch isolated VLAN) in the PVLAN inherit that setting. However, if you want to disable MAC address learning on any community VLANs, you must configure `no-mac-learning` on each of those VLANs.

Creating a PVLAN

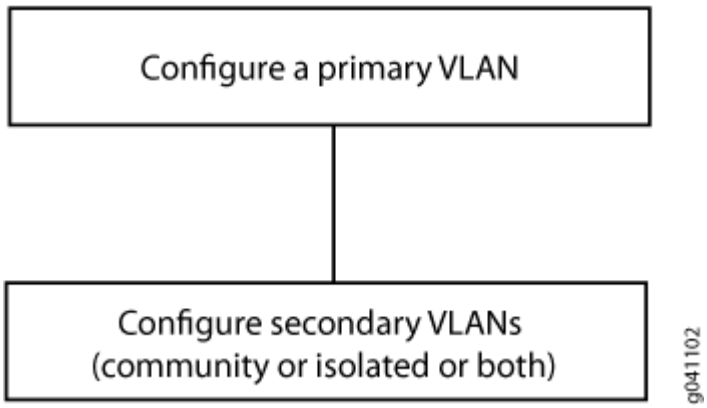
The flowchart shown in Figure 34 on page 629 gives you a general idea of the process for creating PVLANs. If you complete your configuration steps in the order shown, you will not violate these PVLAN rules. (In the PVLAN rules, configuring the PVLAN trunk port applies only to a PVLAN that spans multiple routers.)

- The primary VLAN must be a tagged VLAN.
- If you are going to configure a community VLAN ID, you must first configure the primary VLAN.
- If you are going to configure an isolation VLAN ID, you must first configure the primary VLAN.

NOTE: Configuring a voice over IP (VoIP) VLAN on PVLAN interfaces is not supported.

Configuring a VLAN on a single router is relatively simple, as shown in Figure 34 on page 629.

Figure 34: Configuring a PVLAN on a Single Switch



Configuring a primary VLAN consists of these steps:

1. Configure the primary VLAN name and 802.1Q tag.
2. Set **no-local-switching** on the primary VLAN.
3. Configure the promiscuous trunk port and access ports.
4. Make the promiscuous trunk and access ports members of the primary VLAN.

Within a primary VLAN, you can configure secondary community VLANs or secondary isolated VLANs or both. Configuring a secondary community VLAN consists of these steps:

1. Configure a VLAN using the usual process.
2. Configure access interfaces for the VLAN.
3. Assign a primary VLAN to the community VLAN,

Isolated VLANs are created internally when the isolated VLAN has access interfaces as members and the option **no-local-switching** is enabled on the primary VLAN.

802.1Q tags are required for interswitch isolated VLANs because IEEE 802.1Q uses an internal tagging mechanism by which a trunking device inserts a 4-byte VLAN frame identification tab into the packet header.

Trunk ports are only needed for multirouter PVLAN configurations—the trunk port carries traffic from the primary VLAN and all secondary VLANs.

Limitations of Private VLANs

The following constraints apply to private VLAN configurations:

- An access interface can belong to only one PVLAN domain, that is, it cannot participate in two different primary VLANs.
- A trunk interface can be a member of two secondary VLANs as long as the secondary VLANs are in two *different* primary VLANs. A trunk interface cannot be a member of two secondary VLANs that are in the *same* primary VLAN.
- A single region of Multiple Spanning Tree Protocol (MSTP) must be configured on all VLANs that are included within the PVLAN.
- VLAN Spanning Tree Protocol (VSTP) is not supported.
- IGMP snooping is not supported with private VLANs.
- Routed VLAN interfaces are not supported on private VLANs

- Routing between secondary VLANs in the same primary VLAN is not supported.
- Some configuration statements cannot be specified on a secondary VLAN. You can configure the following statements at the `[edit vlans vlan-name switch-options]` hierarchy level *only* on the primary PVLAN.
- If you want to change a primary VLAN to be a secondary VLAN, you must first change it to a normal VLAN and commit the change. For example, you would follow this procedure:

  1. Change the primary VLAN to be a normal VLAN.
  2. Commit the configuration.
  3. Change the normal VLAN to be a secondary VLAN.
  4. Commit the configuration.

  Follow the same sequence of commits if you want to change a secondary VLAN to be a primary VLAN. That is, make the secondary VLAN a normal VLAN and commit that change and then change the normal VLAN to be a primary VLAN.

The following features are *not* supported on PVLANs on Junos OS switches with support for the ELS configuration style:

- Egress VLAN firewall filters
- Ethernet ring protection (ERP)
- Flexible VLAN tagging
- *global-mac-statistics*
- Multichassis link aggregation groups (MC-LAGs)
- Port mirroring
- Q-in-Q tunneling
- VLAN Spanning Tree Protocol (VSTP)
- Voice over IP (VoIP)

You can configure the following statements at the `[edit vlans vlan-name switch-options]` hierarchy level only on the primary PVLAN:

- *mac-table-size*
- *no-mac-learning*
- *mac-statistics*

- *interface-mac-limit*

RELATED DOCUMENTATION

Understanding Bridging and VLANs on Switches | 140

Bridge Domains Setup in PVLANs on MX Series Routers

Bridge domain capabilities are used to support PVLANs on MX Series routers. Although this functionality is similar to the PVLAN mechanism on EX Series switches, the difference is that only one isolation VLAN can be configured for all isolated ports on MX routers instead of one isolation VLAN permissible per isolated port on EX Series switches.

Assume a sample deployment in which a primary VLAN named VP contains ports p1, p2, t1, t2, i1, i2, cx1, and cx2. The port types of these configured ports are as follows:

- Promiscuous ports = p1, p2
- ISL ports = t1, t2
- Isolated ports = i1, i2
- Community VLAN = Cx
- Community ports = cx1, cx2

Bridge domains are provisioned for each of the VLANs, namely, Vp, Vi, and Vcx. Assume the bridge domains to be configured as follows:

Vp—BD_primary_Vp (ports contained are p1, t1, i1, i2, cx1, cx2)

Vi—BD_isolate_Vi (ports contained are p1, t1, *i1, *i2)

Vcx—BD_community_Vcx (ports contained are p1, t1, cx1, cx2)

The bridge domains for community, primary, and isolated VLANs are automatically created by the system internally when you configure a bridge domain with a trunk interface, access interface, or interswitch link. The bridge domains contain the same VLAN ID corresponding to the VLANs. To use bridge domains for PVLANs, you must configure the following additional attributes:

- `community-vlans` option—This option is specified on all community vlans and for community BDs created internally.

- `isolated-vlan` option—This option denotes the vlan tag to be used for isolation BD created internally for each PVLAN/BD. This setting is required.

- `inter-switch-link` option with the `interface-mode trunk` statement at the `[edit interfaces interface-name family bridge]` or the `[edit interfaces interface-name unit logical-unit-number family bridge]` hierarchy level —This configuration specifies whether the particular interface assumes the role of interswitch link for the PVLAN domains of which it is a member.

You can use the `vlan-id` configuration statement for PVLAN ports to identify the port role. All the logical interfaces involved in PVLANs must be configured with a VLAN ID and the Layer 2 process uses this VLAN tag to classify a port role as promiscuous, isolated, or community port by comparing this value with the VLANs configured in the PVLAN bridge domain (using the `bridge-domains` statement at the `[edit]` hierarchy level). The ISL port role is identified by the `inter-switch-link` option. The VLAN ID for ISL port is required and must be set to to the primary VLAN ID. The ISL must be a trunk interface. A list of VLAN IDs is not needed because the Layer 2 process creates such a list internally based on PVLAN bridge domain configuration. For untagged promiscuous, isolated or community, logical interfaces or ports, access mode must be used as the interface mode. For tagged promiscuous, isolated, or community interfaces, trunk mode must be specified as the interface mode.

The bridge domain interface families are enhanced to include ingress-only and egress-only association. The association for the interface family bridge domain (IFBD) is created in the following manner:

- For BD_primary_Vp, IFBD for i1, i2, cx1 and cx2 are egress only.

- BD_isolate_Vi, IFBD for p1 will be egress only and for i1 and i2 are ingress only.

- BD_community_Vcx , IFBD for p1 are egress only. VLAN translation rules ensure the following VLAN mappings to work properly:

  - VLAN mapping on promiscuous ports: On promiscuous ports, the Vlan Vi is mapped to Vlan Vp on egress interfaces. Similarly on promiscuous ports, Vcx is also be mapped to Vp.

  - VLAN mapping on isolation ports: On tagged isolated ports, the VLAN tag, Vp, is mapped to Vi on egress.

  - VLAN mapping on community ports: On tagged community ports, the VLAN tag, Vp, is mapped to Vcx on egress.

A management bridge domain for PVLAN that exists only in the Layer 2 address learning process called PBD to denote bridge domain for VLAN is used by the system. This bridge domain has the same name as the user-configured name. Under this bridge domain, one primary PVLAN bridge domain for the primary vlan, one isolation bridge domain for the isolation vlan, and one community bridge domain for

each community vlan are programmed internally. You might find separate bridge domains for the PVLAN ports to be useful if you want to configure a policy for a specific community VLAN or isolation VLAN.

The management bridge domain maintains a list to include all internal bridge domains that belong to this PVLAN bridge domain. Isolation and community bridge domains contain a pointer or a flag to indicate that this bridge domain is for PVLANs and maintain the information about the primary bridge domain index and primary VLAN. All this information is available across the bridge domain interfaces that are mapped to this bridge domain. MAC learning occurs only in the primary bridge domain and the MAC forwarding entry is programmed into the primary bridge domain only. As a result, the isolation bridge domain and all community bridge domains share the same forwarding table as the primary bridge domain.

For the isolation bridge domain, BD_isolate_Vi, isolation port i1 and i2 function as a non-local-switch access port and the flood group for this bridge domain contains only the promiscuous port, p1, and ISL ports, t1 and t2.

Bridging Functions With PVLANs

This topic describes how bridging is implemented on MX Series routers that will help with understanding the unique enhancements involved in implementing PVLAN bridging procedures. Consider two ports in a bridging domain with the respective ports on different FPCs and different Packet Forwarding Engines. When a packet enters a port, the following is the flow, assuming it is a tagged packet:

1. As the starting process, a VLAN lookup is performed to determine which bridging domain the packet forms. The result of the lookup identifies the bridging domain id (bd_id), mesh group id (mg_id). With these parameters, other related information configured for this bridging domain is discovered.
2. A source MAC address (SMAC) lookup is performed to find out whether this MAC addresses is learned or not. If it is not a learned address, an MLP packet (route for flooding traffic to MAC learning chips) is sent to all the other Packet Forwarding Engines that are mapped with this bridging domain. In addition, an MLP packet is also sent to the host.
3. A destination MAC address (DMAC) lookup using the tuple (bridge domain ID, VLAN, and destination MAC address).
4. If a match is observed for the MAC address, the result of the lookup points to the egress next-hop. The egress Packet Forwarding Engine is used to forward the packet.
5. If a miss occurs during the lookup, the flood next-hop is determined using the mesh group ID to flood the packet.

The following two significant conditions are considered in PVLAN bridging: Only a specific port to another port forwarding is permitted. A packet drop occurs on the egress interface after traversing and consuming the fabric bandwidth. To avoid traffic dropping, the decision on whether the packet needs to be dropped arrives before traversing the fabric, thereby saving the fabric bandwidth during DoS attacks. Because multiple overlapping bridge domains exist, which denotes that the same port (promiscuous or interswitch link) appears as a member in multiple bridge domains, the MAC addresses learned in one port must be visible to ports on another bridge domain. For example, a MAC address learned on a promiscuous port must be visible to both an isolated port (isolated bridge domain) and a community port (community bridge domain) on the various community bridge domains.

To resolve this problem, a shared VLAN is used for PVLAN bridging. In the shared VLAN model, all the MACs learned across all the ports are stored in the same bridge domain (primary VLAN BD) and same VLAN (primary VLAN). When the VLAN lookup is done for the packet, the PVLAN port, PVLAN bridge domain, and the PVLAN tag or ID are also used. The following processes occur with a shared VLAN methodology:

- A source MAC address (SMAC) lookup is performed to find out whether this MAC address is learned or not. If it is not a learned address, an MLP packet (route for flooding traffic to MAC learning chips) is sent to all the other Packet Forwarding Engines that are mapped with this bridging domain. In addition, an MLP packet is also sent to the host.
- A destination MAC address (DMAC) lookup using the tuple (bridge domain ID, VLAN, and destination MAC address).
- If a match is observed for the MAC address, the result of the lookup points to the egress next-hop. The egress Packet Forwarding Engine is used to forward the packet.
- If a miss occurs during the lookup, the flood next-hop is determined using the mesh group ID to flood the packet.
- If a match occurs, the group ID is derived from the VLAN lookup table and the following validation is performed to enforce primary VLAN forwarding:

```
Steps     Source        Destination              Action
Step 1                    0           {*}                   Permit
Step 2                    {*}         0                   Permit
Step 3                    1               1                   Drop
Step 4     X <-> Y (X > 1 and Y > 1 and X ≠ Y    Drop
```

Here, {*} is a wildcard in regular expression notation referring to any value. Step 1 ensures all forwarding from promiscuous or inter switch link ports to any other port is permitted. Step 2 ensures all forwarding from any port to promiscuous or interswitch link ports is permitted. Step 3 ensures any isolated port to another isolated port is dropped. Step 4 ensures community port forwarding is permitted only within same community(X == Y) and dropped when its across community (X ≠ Y).

Flow of Frames on PVLAN Ports Overview

IN THIS SECTION

This topic describes the manner in which traffic that enters the different PVLAN ports, such as promiscuous, isolated, and interswitch link VLANs, is processed. Sample configuration scenarios are used to describe the transmission and processing of packets.

Assume a sample deployment in which a primary VLAN, or PVLAN, named VP contains ports, p1, p2, t1, t2, i1, i2, cx1, and cx2. The port types of these configured ports are as follows:

- Promiscuous ports = p1, p2
- ISL ports = t1, t2
- Isolated ports = i1, i2
- Community VLAN = Cx
- Community ports = cx1, cx2

VLANs are provisioned for each of the PVLAN types:

Vp—VLAN_primary_Vp (ports contained are p1, t1, i1, i2, cx1, cx2)

Vi—VLAN_isolate_Vi (ports contained are p1, t1, *i1, *i2)

Vcx—VLAN_community_Vcx (ports contained are p1, t1, cx1, cx2)

The VLANs are automatically created by the system internally when you configure a VLAN with a trunk interface, access interface, or interswitch link. To use PVLANs, you must configure the following additional attributes:

Ingress Traffic on Isolated Ports

Consider an ingress port, i1. i1 is mapped to a VLAN named VLAN_isolate_Vi. VLAN_isolate_Vi does not have any isolated ports as an egress member. Frames can only be sent in the egress direction on p1 and t1. When a frame is sent out on p1, it is tagged with the tag of Primary VLAN Vp. A VLAN translation of Vi to Vp is performed. When a frame is propagated out of t1, it is tagged with the tag Vi.

Ingress Traffic on Community Ports

Consider an ingress port as cx1. cx1 is mapped to VLAN VLAN_community_Vcx. Because of the VLAN membership with the VLAN, frames can be sent out of p1, t1, cx1, cx2. When a frame is traversed out on p1, it is tagged with tag of Primary VLAN Vp [VLAN translation]. When a frame goes out of t1, it is tagged with tag Vcx.

Ingress Traffic on Promiscuous Ports

Consider a promisicious port p1 as the ingress port. p1 is mapped to VLAN VLAN_primary_Vp. Frames can go out of any member port. When a frame goes out of t1, it is tagged with tag Vp. If another promiscuous port exists, that frame is also sent out with Vp.

Ingress Traffic on Interswitch Links

With the Vlan tag Vp, assume the ingress port as t1 mapped to VLAN VLAN_primary_Vp. Frames can go out of any member port. When a frame goes out of p1, it is tagged with tag Vp. With the Vlan tag Vi, t1 mapped to VLAN VLAN_isolate_Vi. The frame can not egress isolated ports as they are ingress-only members of VLAN_isolate_Vi. When a frame goes out on p1, it is tagged with tag of Primary VLAN Vp (VLAN translation). When a frame goes out of any other trunk port, it contains the Vi tag. With the Vlan tag Vcx, t1 is mapped to VLAN_community_Vcx. Frames can go out of p1, t1, cx1, and cx2. When a frame goes out on p1, it is tagged with the tag of primary VLAN Vp (VLAN translation).

Packet Forwarding in PVLANs

Consider a primary VLAN with the following configuration of ports:

```
Promiscuous     P1    P2
Inter Switch Link        L1    L2
Isolated                    I1    I2
Community1                C11    C12
Community2                C21    C22
```

Internally, one global VLAN called the primary vlan VLAN is created that consists of all the ports. One isolation VLAN consisting of all isolation ports in addition the promiscuous and ISL ports and one VLAN per community is defined consisting of community ports in addition to the promiscuous and ISL ports internally configured in the system. The VLANs with the PVLAN ports are as follows:

```
Primary Vlan VLAN    P1    P2    L1    L2    I1    I2    C11    C12    C21    C22
Isolated VLAN              I1    I2    P1    P2    L1    L2
Community1 VLAN           C11    C12    P1    P2    L1    L2
Community 2 VLAN            C21    C22    P1    P2    L1    L2
```

The following PVLAN forwarding events take place among these ports with the appropriate VLAN translation as described in the following table:

Table 98: PVLAN Forwarding Events

Port Type To: → From:↓	Isolated	Community	Promiscuous	Inter-switch Link
Isolated	Dropped	Dropped	Primary VLAN tag to Isolation VLAN tag.	If received with the primary VLAN tag, translate to the isolation VLAN Tag; else dropped

Table 98: PVLAN Forwarding Events *(Continued)*

Port Type To: → From:↓	Isolated	Community	Promiscuous	Inter-switch Link
Promiscuous	Dropped	No translation if it is the same community; else dropped.	Primary VLAN tag to Community VLAN tag.	If received with primary VLAN tag, translate to community VLAN tag; else no translation if received with same community vlan else dropped.
Community	Isolated VLAN tag to Primary VLAN tag	Community VLAN tag to Primary VLAN tag	No translation	If received with isolation or community VLAN tag, translate to Primary VLAN tag; else no translation
Interswitch Link	No translation	No translation	No translation	No translation

Guidelines for Configuring PVLANs on MX Series Routers

Consider the following guidelines while you configure PVLANs on MX Series routers that function in enhanced LAN mode:

- PVLANs are supported on MX80 routers, on MX240, MX480, and MX960 routers with DPCs in LAN mode, on MX Series routers with MPCs.
- Isolated ports, promiscuous ports, community ports, and interswitch links (ISL) adhere to the following rules of tagging and forwarding:
  - The frames received on the primary VLAN on promiscuous ports can go to any port.
  - The frames received on isolated ports can only go to promiscuous ports and ISL ports.
  - The frames received on community ports can only go to ports of the same community, promiscuous ports, and ISL ports.

  - The frames received on ISL ports with an isolation VLAN tag or ID can only go to promiscuous ports or ISL ports.
  - The frames received on ISL ports with a community VLAN tag can only go to promiscuous ports, ISL ports, or ports belonging to a corresponding community port.
  - The frames being sent out of promiscuous ports should have a primary VLAN tag or should be untagged. It is considered untagged if the port is configured as an untagged member of the primary VLAN. The frames going out of isolated or community ports are generally untagged. However, they can also be tagged depending on the port configuration. In any case, the configured VLAN tag must be the same as the related isolated VLAN tag or community VLAN tag.
  - The frames going out of ISL ports are tagged with the primary VLAN if they are received on a promiscuous port. An untagged frame cannot exit out of an ISL port in the context of a primary VLAN, isolated VLAN, or community VLAN, but for any other VLAN, it can be untagged depending on the configuration.
  - The frames going out of ISL ports are tagged with an isolated VLAN (isolation ID) if received on the isolated port.
  - The frames going out of ISL ports are tagged with the community VLAN tag, if it is received on the corresponding community port.
- Graceful Routing Engine switchover (GRES) is supported for PVLANs.
- A virtual switch instance that contains a bridge domain associated with logical interfaces is supported.
- Aggregated Ethernet (ae) interfaces for all types of ports are supported.
- Virtual private LAN service (VPLS) instances is not supported. Integrated routing and bridging (IRB) interfaces in PVLANs are supported.
- MX Series Virtual Chassis configuration is not supported.
- MC-LAG interfaces are not supported. All ports that are associated with PVLAN bridge domains cannot be mc-ae interfaces.
- IGMP snooping is not supported. Q-in-Q tunneling is not supported.

Configuring PVLANs on MX Series Routers in Enhanced LAN Mode

You can configure a private VLAN (PVLAN) on a single MX Series router to span multiple MX Series routers. VLANs limit broadcasts to specified users. You need to specify the interswitch link (ISL) for a PVLAN, the PVLAN port types, and secondary VLANs for the PVLAN. You must create a virtual switch routing instance with a bridge domain, and associate the interfaces with the bridge domain. You can specify the secondary VLANs as isolated or community VLANs in the bridge domain.

Before you begin configuring a PVLAN, make sure you have:

- Created and configured the necessary VLANs. See "Configuring VLAN and Extended VLAN Encapsulation" on page 299 and "Enabling VLAN Tagging" on page 277.
- Configured MX240, MX480, and MX960 routers to function in enhanced LAN mode by entering the `network-services lan` statement at the `[edit chassis]` hierarchy level.

  You must reboot the router when you configure or delete the enhanced LAN mode on the router. Configuring the `network-services lan` option implies that the system is running in the enhanced IP mode. When you configure a device to function in MX-LAN mode, only the supported configuration statements and operational show commands that are available for enabling or viewing in this mode are displayed in the CLI interface.

  If your system contains parameters that are not supported in MX-LAN mode in a configuration file, you cannot commit those unsupported attributes. You must remove the settings that are not supported and then commit the configuration. After the successful CLI commit, a system reboot is required for the attributes to become effective. Similarly, if you remove the `network-services lan` statement, the system does not run in MX-LAN mode. Therefore, all of the settings that are supported outside of the MX-LAN mode are displayed and are available for definition in the CLI interface. If your configuration file contains settings that are supported only in MX-LAN mode, you must remove those attributes before you commit the configuration. After the successful CLI commit, a system reboot is required for the CLI parameters to take effect. The Layer 2 Next-Generation CLI configuration settings are supported in MX-LAN mode. As a result, the typical format of CLI configurations might differ in MX-LAN mode.

To configure a PVLAN:

1. Create a promiscuous port for the PVLAN.

```
[edit interfaces]
user@host# set interface interface-name unit logical-unit-number family bridge interface-mode
```

```
trunk
user@host# set interface interface-name unit logical-unit-number family bridge vlan-id vlan-id
```

2. Create the interswitch link (ISL) trunk port for the PVLAN.

```
[edit interfaces]
user@host# set interface interface-name unit logical-unit-number family bridge interface-mode
trunk inter-switch-link
user@host# set interface interface-name unit logical-unit-number family bridge vlan-id vlan-id
```

3. Create the isolated port for the PVLAN. The port is identified as an isolated port or a community port, based on the VLAN ID or the list of VLAN IDs to which the interface corresponds. For example, if you configure a port with a VLAN ID of 50, and if you specify a VLAN ID of 50 as the isolated VLAN or tag in the bridge domain, the port is considered as an isolation port.

```
[edit interfaces]
user@host# set interface interface-name unit logical-unit-number family bridge interface-mode
access
user@host# set interface interface-name unit logical-unit-number family bridge vlan-id vlan-id
```

4. Create the community port for the PVLAN. The port is identified as an isolated port or a community port, based on the VLAN ID or the list of VLAN IDs to which the interface corresponds. For example, if you configure a port with a VLAN ID of 50, and if you specify a VLAN ID of 50 as the community VLAN or tag in the bridge domain, the port is considered as a community port.

```
[edit interfaces]
user@host# set interface interface-name unit logical-unit-number family bridge interface-mode
access
user@host# set interface interface-name unit logical-unit-number family bridge vlan-id vlan-id
```

5. Create a virtual switch instance with a bridge domain and associate the logical interfaces.

```
[edit routing-instances]
user@host# set routing-instance-name instance-type virtual-switch
user@host# set routing-instance-name interface interface-name unit logical-unit-number
user@host# set routing-instance-name bridge-domains bridge-domain-name
```

6. Specify the primary, isolated, and community VLAN IDs, and associate the VLANs with the bridge domain.

```
[edit routing-instances instance-name bridge-domains bridge-domain-name]
user@host# set vlan-id vlan-id
user@host# set isolated-vlan vlan-id
user@host# set community-vlans [ number number-number ]
```

Example: Configuring PVLANs with Secondary VLAN Trunk Ports and Promiscuous Access Ports on a QFX Series Switch

IN THIS SECTION

This example shows how to configure secondary VLAN trunk ports and promiscuous access ports as part of a private VLAN configuration. Secondary VLAN trunk ports carry secondary VLAN traffic.

NOTE: This example uses Junos OS for switches that do not support the Enhanced Layer 2 Software (ELS) configuration style. For more about ELS, see "Using the Enhanced Layer 2 Software CLI" on page 15.

For a given private VLAN, a secondary VLAN trunk port can carry traffic for only one secondary VLAN. However, a secondary VLAN trunk port can carry traffic for multiple secondary VLANs as long as each secondary VLAN is a member of a different private (primary) VLAN. For example, a secondary VLAN

trunk port can carry traffic for a community VLAN that is part of primary VLAN pvlan100 and also carry traffic for an isolated VLAN that is part of primary VLAN pvlan400.

To configure a trunk port to carry secondary VLAN traffic, use the *isolated* and `interface` statements, as shown in steps "12" on page 650 and "13" on page 651 of the example configuration for Switch 1.

NOTE: When traffic egresses from a secondary VLAN trunk port, it normally carries the tag of the primary VLAN that the secondary port is a member of. If you want traffic that egresses from a secondary VLAN trunk port to retain its secondary VLAN tag, use the *extend-secondary-vlan-id* statement.

A promiscuous access port carries untagged traffic and can be a member of only one primary VLAN. Traffic that ingresses on a promiscuous access port is forwarded to the ports of the secondary VLANs that are members of the primary VLAN that the promiscuous access port is a member of. This traffic carries the appropriate secondary VLAN tags when it egresses from the secondary VLAN ports if the secondary VLAN port is a trunk port.

To configure an access port to be promiscuous, use the *promiscuous* statement, as shown in step "12" on page 657 of the example configuration for Switch 2.

If traffic ingresses on a secondary VLAN port and egresses on a promiscuous access port, the traffic is untagged on egress. If tagged traffic ingresses on a promiscuous access port, the traffic is discarded.

Requirements

This example uses the following hardware and software components:

- Two QFX devices
- Junos OS Release 12.2 or later for the QFX Series

Overview and Topology

Figure 35 on page 645 shows the topology used in this example. Switch 1 includes several primary and secondary private VLANs and also includes two secondary VLAN trunk ports configured to carry secondary VLANs that are members of primary VLANs pvlan100 and pvlan400.

Switch 2 includes the same private VLANs. The figure shows xe-0/0/0 on Switch 2 as configured with promiscuous access ports or promiscuous trunk ports. The example configuration included here configures this port as a promiscuous access port.

The figure also shows how traffic would flow after ingressing on the secondary VLAN trunk ports on Switch 1.

Figure 35: PVLAN Topology with Secondary VLAN Trunk Ports and Promiscuous Access Port

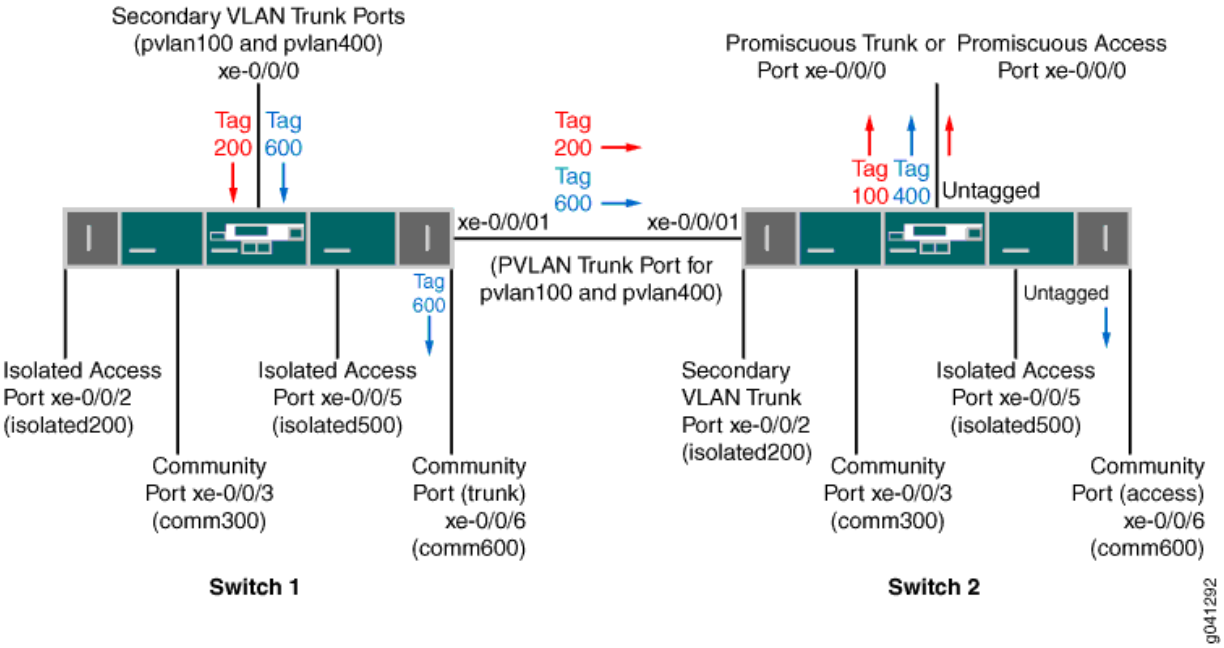


Table 99 on page 645 and Table 100 on page 646 list the settings for the example topology on both switches.

Table 99: Components of the Topology for Configuring a Secondary VLAN Trunk on Switch 1

Component	Description
pvlan100, ID 100	Primary VLAN
pvlan400, ID 400	Primary VLAN
comm300, ID 300	Community VLAN, member of pvlan100
comm600, ID 600	Community VLAN, member of pvlan400
isolation-vlan-id 200	VLAN ID for isolated VLAN, member of pvlan100

Table 99: Components of the Topology for Configuring a Secondary VLAN Trunk on Switch 1 ***(Continued)***

Component	Description
isolation–vlan-id 500	VLAN ID for isolated VLAN, member of pvlan400
xe-0/0/0.0	Secondary VLAN trunk port for primary VLANs pvlan100 and pvlan400
xe-0/0/1.0	PVLAN trunk port for primary VLANs pvlan100 and pvlan400
xe-0/0/2.0	Isolated access port for pvlan100
xe-0/0/3.0	Community access port for comm300
xe-0/0/5.0	Isolated access port for pvlan400
xe-0/0/6.0	Community trunk port for comm600

Table 100: Components of the Topology for Configuring a Secondary VLAN Trunk on Switch 2

Component	Description
pvlan100, ID 100	Primary VLAN
pvlan400, ID 400	Primary VLAN
comm300, ID 300	Community VLAN, member of pvlan100
comm600, ID 600	Community VLAN, member of pvlan400
isolation-vlan-id 200	VLAN ID for isolated VLAN, member of pvlan100
isolation–vlan-id 500	VLAN ID for isolated VLAN, member of pvlan400
xe-0/0/0.0	Promiscuous access port for primary VLANs pvlan100

Table 100: Components of the Topology for Configuring a Secondary VLAN Trunk on Switch 2 *(Continued)*

Component	Description
xe-0/0/1.0	PVLAN trunk port for primary VLANs pvlan100 and pvlan400
xe-0/0/2.0	Secondary trunk port for isolated VLAN, member of pvlan100
xe-0/0/3.0	Community access port for comm300
xe-0/0/5.0	Isolated access port for pvlan400
xe-0/0/6.0	Community access port for comm600

Configuring the PVLANs on Switch 1

IN THIS SECTION

CLI Quick Configuration

To quickly create and configure the PVLANs on Switch 1, copy the following commands and paste them into a switch terminal window:

```
[edit]
set interfaces xe-0/0/0 unit 0 family ethernet-switching port-mode trunk
set interfaces xe-0/0/1 unit 0 family ethernet-switching port-mode trunk
set interfaces xe-0/0/1 unit 0 family ethernet-switching vlan members pvlan100
set interfacesxe-0/0/1 unit 0 family ethernet-switching vlan members pvlan400
```

```
set interfaces xe-0/0/2 unit 0 family ethernet-switching port-mode access
set interfaces xe-0/0/3 unit 0 family ethernet-switching port-mode access
set interfaces xe-0/0/5 unit 0 family ethernet-switching port-mode access
set interfaces xe-0/0/6 unit 0 family ethernet-switching port-mode trunk
set vlans pvlan100 vlan-id 100
set vlans pvlan400 vlan-id 400
set vlans pvlan100 pvlan
set vlans pvlan400 pvlan
set vlans pvlan100 interface xe-0/0/1.0 pvlan-trunk

set vlans pvlan400 interface xe-0/0/1.0 pvlan-trunk
set vlans comm300 vlan-id 300
set vlans comm300 primary-vlan pvlan100
set vlans comm300 interface xe-0/0/3.0
set vlans comm600 vlan-id 600
set vlans comm600 primary-vlan pvlan400
set vlans comm600 interface xe-0/0/6.0
set vlans pvlan100 pvlan isolation-vlan-id 200
set vlans pvlan400 pvlan isolation-vlan-id 500
set vlans pvlan100 interface xe-0/0/0.0 isolated
set vlans pvlan400 interface xe-0/0/0.0 isolated
set vlans comm600 interface xe-0/0/0.0
set vlans pvlan100 interface xe-0/0/2.0 isolated
set vlans pvlan400 interface xe-0/0/5.0 isolated
```

Procedure

Step-by-Step Procedure

To configure the private VLANs and secondary VLAN trunk ports:

1. Configure the interfaces and port modes:

```
[edit interfaces]
user@switch# set xe-0/0/0 unit 0 family ethernet-switching port-mode trunk
user@switch# set xe-0/0/1 unit 0 family ethernet-switching port-mode trunk
user@switch# set xe-0/0/1 unit 0 family ethernet-switching vlan members pvlan100
user@switch# set xe-0/0/1 unit 0 family ethernet-switching vlan members pvlan400
user@switch# set xe-0/0/2 unit 0 family ethernet-switching port-mode access
user@switch# set xe-0/0/3 unit 0 family ethernet-switching port-mode access
```

```
user@switch# set xe-0/0/5 unit 0 family ethernet-switching port-mode access
user@switch# set xe-0/0/6 unit 0 family ethernet-switching port-mode access
```

2. Create the primary VLANs:

```
[edit vlans]
user@switch# set pvlan100 vlan-id 100
user@switch# set pvlan400 vlan-id 400
```

NOTE: Primary VLANs must always be tagged VLANs, even if they exist on only one device.

3. Configure the primary VLANs to be private:

```
 [edit vlans]
user@switch# set pvlan100 pvlan
user@switch# set pvlan400 pvlan
```

4. Configure the PVLAN trunk port to carry the private VLAN traffic between the switches:

```
[edit vlans]
user@switch# set pvlan100 interface xe-0/0/1.0 pvlan-trunk
user@switch# set pvlan400 interface xe-0/0/1.0 pvlan-trunk
```

5. Create secondary VLAN comm300 with VLAN ID 300:

```
[edit vlans]
user@switch# set comm300 vlan-id 300
```

6. Configure the primary VLAN for comm300:

```
[edit vlans]
user@switch# set comm300 primary-vlan pvlan100
```

7. Configure the interface for comm300:

```
[edit vlans]
user@switch# set comm300 interface xe-0/0/3.0
```

8. Create secondary VLAN comm600 with VLAN ID 600:

```
[edit vlans]
user@switch# set comm600 vlan-id 600
```

9. Configure the primary VLAN for comm600:

```
[edit vlans]
user@switch# set comm600 primary-vlan pvlan400
```

10. Configure the interface for comm600:

```
[edit vlans]
user@switch# set comm600 interface xe-0/0/6.0
```

11. Configure the interswitch isolated VLANs:

```
[edit vlans]
user@switch# set pvlan100 pvlan isolation-vlan-id 200
user@switch# set pvlan400 pvlan isolation-vlan-id 500
```

NOTE: When you configure a secondary VLAN trunk port to carry an isolated VLAN, you must also configure an *isolation-vlan-id*. This is true even if the isolated VLAN exists only on one switch.

12. Enable trunk port xe-0/0/0 to carry secondary VLANs for the primary VLANs:

```
[edit vlans]
user@switch# set pvlan100 interface xe-0/0/0.0 isolated
user@switch# set pvlan400 interface xe-0/0/0.0 isolated
```

13. Configure trunk port xe-0/0/0 to carry comm600 (member of pvlan400):

```
[edit vlans]
user@switch# set comm600 interface xe-0/0/0.0
```

NOTE: You do not need to explicitly configure xe-0/0/0 to carry the isolated VLAN traffic (tags 200 and 500) because all the isolated ports in pvlan100 and pvlan400–including xe-0/0/0.0–are automatically included in the isolated VLANs created when you configured `isolation-vlan-id 200` and `isolation-vlan-id 500`.

14. Configure xe-0/0/2 and xe-0/0/6 to be isolated:

```
[edit vlans]
user@switch# set pvlan100 interface xe-0/0/2.0 isolated
user@switch# set pvlan400 interface xe-0/0/5.0 isolated
```

Results

Check the results of the configuration on Switch 1:

```
[edit]
user@switch#  show
interfaces {
    xe-0/0/0 {
        unit 0 {
            family ethernet-switching {
                port-mode trunk;
                vlan {
                    members pvlan100;
                    members pvlan400;
                }
            }
        }
    }
    xe-0/0/1 {
        unit 0 {
            family ethernet-switching {
                port-mode trunk;
```

```
                vlan {
                    members pvlan100;
                    members pvlan400;
                }
            }
        }
    }
    xe-0/0/2 {
        unit 0 {
            family ethernet-switching {
                port-mode access;
            }
        }
    }
    xe-0/0/3 {
        unit 0 {
            family ethernet-switching {
                port-mode access;
            }
        }
    }
    xe-0/0/5 {
        unit 0 {
            family ethernet-switching {
                port-mode access;
            }
        }
    }
    xe-0/0/6 {
        unit 0 {
            family ethernet-switching {
                port-mode trunk;
            }
        }
    }
}
vlans {
    comm300 {
        vlan-id 300;
            interface {
            xe-0/0/3.0;
        }
        primary-vlan pvlan100;
```

```
    }
    comm600 {
        vlan-id 600;
            interface {
            xe-0/0/6.0;
        }
        primary-vlan pvlan400;
    }
    pvlan100 {
        vlan-id 100;
        interface {
            xe-0/0/0.0;
            xe-0/0/2.0;
            xe-0/0/3.0;
            xe-0/0/1.0 {
                pvlan-trunk;
            }
        }
        no-local-switching;
        isolation-id 200;
    }
    pvlan400 {
        vlan-id 400;
        interface {
            xe-0/0/0.0;
            xe-0/0/5.0;
            xe-0/0/6.0;
            xe-0/0/1.0 {
                pvlan-trunk;
            }
        }
        no-local-switching;
        isolation-id 500;
    }
}
```

Configuring the PVLANs on Switch 2

IN THIS SECTION

The configuration for Switch 2 is almost identical to the configuration for Switch 1. The most significant difference is that xe-0/0/0 on Switch 2 is configured as a promiscuous trunk port or a promiscuous access port, as Figure 35 on page 645 shows. In the following configuration, xe-0/0/0 is configured as a promiscuous access port for primary VLAN pvlan100.

If traffic ingresses on VLAN-enabled port and egresses on a promiscuous access port, the VLAN tags are dropped on egress and the traffic is untagged at that point. For example, traffic for comm600 ingresses on the secondary VLAN trunk port configured on xe-0/0/0.0 on Switch 1 and carries tag 600 as it is forwarded through the secondary VLAN. When it egresses from xe-0/0/0.0 on Switch 2, it will be untagged if you configure xe-0/0/0.0 as a promiscuous access port as shown in this example. If you instead configure xe-0/0/0.0 as a promiscuous trunk port (port-mode trunk), the traffic for comm600 carries its primary VLAN tag (400) when it egresses.

CLI Quick Configuration

To quickly create and configure the PVLANs on Switch 2, copy the following commands and paste them into a switch terminal window:

```
[edit]
set interfaces xe-0/0/0 unit 0 family ethernet-switching port-mode access
set interfaces xe-0/0/1 unit 0 family ethernet-switching port-mode trunk
set interfaces xe-0/0/1 unit 0 family ethernet-switching vlan members pvlan100
set interfaces xe-0/0/1 unit 0 family ethernet-switching vlan members pvlan400

set interfaces xe-0/0/2 unit 0 family ethernet-switching port-mode trunk
set interfaces xe-0/0/3 unit 0 family ethernet-switching port-mode access
set interfaces xe-0/0/5 unit 0 family ethernet-switching port-mode access
set interfaces xe-0/0/6 unit 0 family ethernet-switching port-mode access
set vlans pvlan100 vlan-id 100
set vlans pvlan400 vlan-id 400
```

```
set vlans pvlan100 pvlan
set vlans pvlan400 pvlan
set vlans pvlan100 interface xe-0/0/1.0 pvlan-trunk

set vlans pvlan400 interface xe-0/0/1.0 pvlan-trunk
set vlans comm300 vlan-id 300
set vlans comm300 primary-vlan pvlan100
set vlans comm300 interface xe-0/0/3.0
set vlans comm600 vlan-id 600
set vlans comm600 primary-vlan pvlan400
set vlans comm600 interface xe-0/0/6.0
set vlans pvlan100 pvlan isolation-vlan-id 200
set vlans pvlan400 pvlan isolation-vlan-id 500
set vlans pvlan100 interface xe-0/0/0.0 promiscuous
set vlans pvlan100 interface xe-0/0/2.0 isolated
set vlans pvlan400 interface xe-0/0/5.0 isolated
```

Procedure

Step-by-Step Procedure

To configure the private VLANs and secondary VLAN trunk ports:

1. Configure the interfaces and port modes:

```
[edit interfaces]
user@switch# set xe-0/0/0 unit 0 family ethernet-switching port-mode access
```

```
user@switch# set xe-0/0/1 unit 0 family ethernet-switching port-mode trunk
user@switch# set xe-0/0/1 unit 0 family ethernet-switching vlan members pvlan100
user@switch# set xe-0/0/1 unit 0 family ethernet-switching vlan members pvlan400

user@switch# set xe-0/0/2 unit 0 family ethernet-switching port-mode trunk
user@switch# set xe-0/0/3 unit 0 family ethernet-switching port-mode access
user@switch# set xe-0/0/5 unit 0 family ethernet-switching port-mode access
user@switch# set xe-0/0/6 unit 0 family ethernet-switching port-mode access
```

2. Create the primary VLANs:

```
[edit vlans]
user@switch# set pvlan100 vlan-id 100
user@switch# set pvlan400 vlan-id 400
```

3. Configure the primary VLANs to be private:

```
 [edit vlans]
user@switch# set pvlan100 pvlan
user@switch# set pvlan400 pvlan
```

4. Configure the PVLAN trunk port to carry the private VLAN traffic between the switches:

```
[edit vlans]
user@switch# set pvlan100 interface xe-0/0/1.0 pvlan-trunk
user@switch# set pvlan400 interface xe-0/0/1.0 pvlan-trunk
```

5. Create secondary VLAN comm300 with VLAN ID 300:

```
[edit vlans]
user@switch# set comm300 vlan-id 300
```

6. Configure the primary VLAN for comm300:

```
[edit vlans]
user@switch# set comm300 primary-vlan pvlan100
```

7. Configure the interface for comm300:

```
[edit vlans]
user@switch# set comm300 interface xe-0/0/3.0
```

8. Create secondary VLAN comm600 with VLAN ID 600:

```
[edit vlans]
user@switch# set comm600 vlan-id 600
```

9. Configure the primary VLAN for comm600:

```
[edit vlans]
user@switch# set comm600 primary-vlan pvlan400
```

10. Configure the interface for comm600:

```
[edit vlans]
user@switch# set comm600 interface xe-0/0/6.0
```

11. Configuring the PVLANs on Switch 1
    Configure the interswitch isolated VLANs:

```
[edit vlans]
user@switch# set pvlan100 pvlan isolation-vlan-id 200
user@switch# set pvlan400 pvlan isolation-vlan-id 500
```

12. Configure access port xe-0/0/0 to be promiscuous for pvlan100:

```
[edit vlans]
user@switch# set pvlan100 interface xe-0/0/0.0 promiscuous
```

NOTE: A promiscuous access port can be a member of only one primary VLAN.

13. Configure xe-0/0/2 and xe-0/0/6 to be isolated:

```
[edit vlans]
user@switch# set pvlan100 interface xe-0/0/2.0 isolated
user@switch# set pvlan400 interface xe-0/0/5.0 isolated
```

Results

Check the results of the configuration on Switch 2:

```
[edit]
user@switch#  show
interfaces {
    xe-0/0/0 {
        unit 0 {
            family ethernet-switching {
                port-mode access;
                vlan {
                    members pvlan100;
                }
            }
        }
    }

        xe-0/0/1 {
        unit 0 {
            family ethernet-switching {
                port-mode trunk;
                vlan {
                    members pvlan100;
                    members pvlan400;
                }
            }
        }
    }
    xe-0/0/2 {
        unit 0 {
            family ethernet-switching {
                port-mode trunk;
            }
        }
    }
    xe-0/0/3 {
        unit 0 {
            family ethernet-switching {
                port-mode access;
            }
        }
```

```
    }
    xe-0/0/5 {
        unit 0 {
            family ethernet-switching {
                port-mode access;
            }
        }
    }
    xe-0/0/6 {
        unit 0 {
            family ethernet-switching {
                port-mode access;
            }
        }
    }
vlans {
    comm300 {
        vlan-id 300;
            interface {
            xe-0/0/3.0;
        }
        primary-vlan pvlan100;
    }
    comm600 {
        vlan-id 600;
            interface {
            xe-0/0/6.0;
        }
        primary-vlan pvlan400;
    }
    pvlan100 {
        vlan-id 100;
        interface {
            xe-0/0/0.0;
            xe-0/0/2.0;
            xe-0/0/3.0;
            xe-0/0/1.0 {
                pvlan-trunk;
            }
        }
        no-local-switching;
        isolation-id 200;
    }
```

```
    pvlan400 {
        vlan-id 400;
        interface {
            xe-0/0/5.0;
            xe-0/0/6.0;
            xe-0/0/1.0 {
                pvlan-trunk;
            }
        }
        no-local-switching;
        isolation-id 500;
    }
}
```

Verification

IN THIS SECTION

To confirm that the configuration is working properly, perform these tasks:

Verifying That the Private VLAN and Secondary VLANs Were Created

Purpose

Verify that the primary VLAN and secondary VLANs were properly created on Switch 1.

Action

Use the `show vlans` command:

```
user@switch>  show vlans private-vlan
```

```
Name               Role        Tag      Interfaces
pvlan100           Primary     100         xe-0/0/0.0, xe-0/0/1.0, xe-0/0/2.0, xe-0/0/3.0
__iso_pvlan100__   Isolated    200      xe-0/0/2.0
comm300            Community   300      xe-0/0/3.0

pvlan400           Primary     400      xe-0/0/0.0, xe-0/0/1.0, xe-0/0/5.0, xe-0/0/6.0
__iso_pvlan400__   Isolated    500      xe-0/0/5.0
comm600            Community   600     xe-0/0/6.0
```

Meaning

The output shows that the private VLANs were created and identifies the interfaces and secondary VLANs associated with them.

Verifying The Ethernet Switching Table Entries

Purpose

Verify that the Ethernet switching table entries were created for primary VLAN pvlan100.

Action

Show the Ethernet switching table entries for pvlan100.

```
user@switch> show ethernet-switching table vlan pvlan100 private-vlan
Ethernet-switching table: 0 unicast entries
  pvlan100             *                 Flood          - All-members
  pvlan100             00:10:94:00:00:02 Learn            xe-0/0/2.0
  __iso_pvlan100__     *                 Flood          - All-members
  __iso_pvlan100__     00:10:94:00:00:02 Replicated      - xe-0/0/2.0
```

RELATED DOCUMENTATION

Understanding Egress Firewall Filters with PVLANs

IRB Interfaces in Private VLANs on MX Series Routers

You can configure integrated routing and bridging (IRB) interfaces in a private VLAN (PVLAN) on a single MX router to span multiple MX routers. PVLANs limit the communication within a VLAN by restricting traffic flows through their member switch ports (which are called “private ports”) so that these ports communicate only with a specified uplink trunk port or with specified ports within the same VLAN. IRB provides simultaneous support for Layer 2 bridging and Layer 3 routing on the same interface. IRB enables you to route packets to another routed interface or to another bridge domain that has an IRB interface configured. You configure a logical routing interface by including the `irb` statement at the `[edit interfaces]` hierarchy level and include that interface in the bridge domain.

PVLANs are supported on MX80 routers, on MX240, MX480, and MX960 routers with DPCs in LAN mode, and on MX Series routers with MPC1, MPC2, and Adaptive Services PICs. This functionality is supported only on MX240, MX480, and MX960 routers that function in enhanced LAN mode (by entering the `network-services lan` statement at the `[edit chassis]` hierarchy level).

IRB in PVLANs replaces the external router used for routing across VLANs. The routing operations in the absence of IRB occur through external router connected to promiscuous port. This behavior takes care of all the routed frames for all the ports defined under the PVLAN domain. In this case, no layer 3 exchange occurs on MX Series routers in enhanced LAN mode for this PVLAN bridge domain. In the case of IRB, the Layer 3 interface is associated with the primary VLAN that is configured and is considered to be a single Layer 3 interface for the entire PVLAN domain. The ingress routed traffic from all ports in the PVLAN domain needs to be mapped to this IRB interface. The egress of the IRB interface take places under the PVLAN. For a PVLAN domain spanning multiple switches, only one IRB interface can be configured in one switch. This IRB interface represent whole PVLAN domain to interact with the Layer 3 domains. An IRB interface only associates with the primary bridge domain and all Layer 3 forwarding occurs only in the primary bridge domain. When a Layer 3 packet is received in an isolated port or a promiscuous port, the device first locates the secondary bridge domain, based on secondary bridge domain to find primary bridge domain identifier. If the destination MAC address is the local IRB MAC address, the microcode transmits the packet to IRB interface associated with primary bridge domain for further processing. The same procedure occurs for receiver Layer 3 packets in an interswitch link (ISL) port with the isolated or community VLAN tag.

For the ingress Layer 3 packet with Layer 3 forwarding logic sent to IRB interfaces associated with a PVLAN bridge domain, the device processes and determines the ARP entry to send packet to the related interface that might be an isolated port or a community port. The microcode appends or translates the packet VLAN ID to the isolation or community vlan ID based on the port type. The VLAN ID is removed if the related port is untagged. A special operational case exists for Layer 3 packets that are forwarded to remote isolated or community port through the ISL link. The Layer 3 packet might contain the primary bridge domain VLAN ID and the remote node performs the translation or pop operation when it sends

the packet out on the related port. This method of processing is different from Layer 2 domains. Because all forwarding base on ARP must be unicast traffic and in the remote node, the port that must be used to forward is known and the transmission of PVLAN ID occurs properly.

An ARP entry carries only the primary bridge domain information. When an ARP response is received from an isolated port or a promiscuous port, the system identifies the secondary bridge domain, and based on the secondary bridge domain, it attempts to retrieve the primary bridge domain identifier. ARP packets eventually reach the IRB interface associated with the primary bridge domain. The kernel considers this ARP packet as a normal bridge domain and creates and maintains the ARP entry only for the primary bridge domain. The same procedure is adopted for ARP request packets that are destined for the local IRB MAC address. The response is transmitted through the IRB interface and appropriate VLAN translation or a pop operation is performed, depending on the received interface.

Guidelines for Configuring IRB Interfaces in PVLANs on MX Series Routers

Keep the following points in mind when you configure IRB interfaces for PVLANs:

- All of the IP applications such as IP multicast, IPv4, IPv6, and VRRP that are compatible with IRB in normal bridge domains function properly when IRB for PVLAN bridge domains is configured.
- MC-LAG interfaces are not supported. All ports that are associated with PVLAN bridge domains cannot be mc-ae interfaces.
- IGMP snooping is not supported.
- A virtual switch instance that contains a bridge domain associated with logical interfaces is supported.
- Q-in-Q tunneling is not supported.
- Logical systems are not supported.
- Virtual private LAN service (VPLS) and Etherent VPN (EVPN) in virtual switch routing instances are not supported. A validation is performed if you attempt to configure Layer 3 interfaces in a secondary VLAN.
- MX Series Virtual Chassis configuration is not supported.

Forwarding of Packets Using IRB Interfaces in PVLANs

IN THIS SECTION

This topic describes how PVLAN packet forwarding operates with IRB interfaces on MX Series routers in enhanced LAN mode. The IRB interface operates as a Layer 3 gateway for all members of a bridging domain. All the members of bridging domain are assumed to be in the same subnet as the subnet of the IRB interface, which works as a gateway.

Consider a sample deployment scenario in which two routers, Router1 and Router2, are configured with a PVLAN. On Router1, the promiscuous port is P1, interswitch link is L1, isolated port is I1, and two community ports are C11 and C21. Similarly, on Router2, the promiscuous port is P2, interswitch link is L2, isolated port is I2, and two community ports are C12 and C22. In the example configuration, the two routers are interconnected through an ISL link, L1 with L2. A PVLAN domain is defined across these two routers encompassing a subdomain of isolated ports (I1, I2), and Community1 ports (C11, C12), and Community2 ports (C21, C22). Because all the ports are in the same subnet, without IRB, switching capability works across ports, across routers following the PVLAN rules. When the end-host needs to reach out cross the subnet, you must configure IRB on the bridging domain. From an end-host perspective, to reach out across the bridging domain, it needs to be configured with the IRB IP address as the default gateway address. All Layer 3 connectivity is established by processing ARP request and ARP responses. The following sections describe the different scenarios encountered for Layer 3 traffic support in PVLANs.

Incoming ARP Requests on PVLAN Ports

ARP requests enter a PVLAN port as broadcast packets. All packets that enter in the ingress direction of a PVLAN domain contain their bridge domain ID translated into the primary VLAN bridge domain ID. In

this case, the bridge domain ID contained in the ARP packet is also translated be to the bridge domain ID of the primary VLAN. When IRB is configured in a bridging domain, the IRB MAC address is added to the MAC table as an eligible destination MAC address on the primary VLAN bridge domain ID. The ARP request is flooded to all ports of the secondary bridging domain in which it was received and, in addition, a copy is sent to the IRB logical interface.

When an IRB logical interface receives this packet, it sends the packet to the host as an ARP packet with the primary BD and the Layer 2 logical interface on which it is received. The PVLAN domain learns the source MAC address of the ARP packet and the kernel learns the sender IP of the ARP packet, and triggers a next-hop installation. If the ARP request is destined for IRB IP address, then an ARP response is sent. If proxy ARP is enabled on IRB, IRB responds with an ARP reply if the destination IP address is known.

The preceding configuration case describes a scenario the ARP request came on Local PVLAN port. If the ARP request is received on a remote PVLAN port, then it is flooded on all the ports of the remote PVLAN domain. Because IRB is configured only on one router of the PVLAN domain, on the remote PVLAN, the flooding is on all the ports. As part of the flooding in the remote PVLAN domain, a copy of the packet is sent to the ISL port. The ISL port processes this packet as though it was received on the local isolated port or community port and the aforementioned method of processing takes place

Outgoing ARP Responses on PVLAN Ports

When a ARP request is received in the kernel, both the bridge domain ID and the receiving Layer 2 logical interface are transmitted. A next-hop installation is triggered to create a next-hop to the Layer 2 logical interface for the sender IP address with the IRB MAC Address as the destination MAC address and the sender MAC address as the source MAC address, with both these addresses appearing as Layer 2 rewrite during the next-hop. If the ARP request queries for the IRB IP address, then an ARP response is sent to the receiving Layer 2 logical interface. If the ARP request queries for an IP address other than the IRB IP address, it is processed as though proxy ARP is enabled on IRB or it is discarded. Because all ARP requests are processed as being received on the primary VLAN, the response is also sent with the primary VLAN. However, when it reaches the receiving Layer 2 logical interface, the appropriate VLAN translation takes place.

The preceding scenario describes an ARP response being sent on a local PVLAN port. If the ARP request is received from a remote PVLAN domain, the receiving Layer 2 logical interface is the ISL port. In this case, the ARP response is sent to the ISL port, on the remote PVLAN domain, the ARP response received on the ISL port is forwarded to the same port where the ARP request is received. This behavior is possible because the source MAC address of the ARP request is learned on the shared VLAN.

Outgoing ARP Requests on PVLAN Ports

When IRB has to advertise a ARP request, it uses the kernel flood next-hop for the primary VLAN and floods to all the ports in the local PVLAN domain. The receiving ISL port also floods the packet to the remote PVLAN domain. Although the ARP request is constructed with the primary VLAN, in the egress direction, appropriate VLAN translation or VLAN pop is performed using the specific port.

Incoming ARP Responses on PVLAN Ports

ARP responses are unicast packets with the destination MAC address as the IRB MAC Address. When such a packet is received on the local PVLAN domain where IRB is enabled, it is forwarded to the IRB logical interface. When the packet arrives at the IRB logical interface, it is propagated to the host. The kernel triggers a next-hop installation with the appropriate Layer 2 rewrite. This operation works properly for ARP responses received on the local PVLAN port. If the ARP response is received on a remote PVLAN port, it is forwarded similar to a normal Layer 2 packet because IRB is not enabled in such a scenario. When the ARP request is sent out from the local PVLAN domain, the receiving ISL port in the remote PVLAN domain might have learned the IRB MAC address on that port, and this address is used to forward the packet to the IRB logical interface.

Receipt of Layer 3 Packets on PVLAN Ports

The packet is received with the IRB MAC address as the destination MAC address and it is processed through the IRB logical interface. The packet is forwarded in the same manner as a regular IP packet.

Configuring IRB Interfaces in PVLAN Bridge Domains on MX Series Routers in Enhanced LAN Mode

You can configure integrated routing and bridging (IRB) interfaces in a private VLAN (PVLAN) on a single MX router to span multiple MX routers. PVLANs limit the communication within a VLAN by restricting traffic flows through their member switch ports (which are called “private ports”) so that these ports communicate only with a specified uplink trunk port or with specified ports within the same VLAN. IRB provides simultaneous support for Layer 2 bridging and Layer 3 routing on the same interface. IRB

enables you to route packets to another routed interface or to another bridge domain that has an IRB interface configured. You configure a logical routing interface and include that interface in the virtual switch instance that contains the bridge domain. You can specify the secondary VLANs as isolated or community VLANs in the bridge domain.

Before you begin configuring a PVLAN, make sure you have:

- Created and configured the necessary VLANs. See "Configuring VLAN and Extended VLAN Encapsulation" on page 299 and "Enabling VLAN Tagging" on page 277.
- Configured MX240, MX480, and MX960 routers to function in enhanced LAN mode by entering the `network-services lan` statement at the `[edit chassis]` hierarchy level.

  You must reboot the router when you configure or delete the enhanced LAN mode on the router. Configuring the `network-services lan` option implies that the system is running in the enhanced IP mode. When you configure a device to function in MX-LAN mode, only the supported configuration statements and operational show commands that are available for enabling or viewing in this mode are displayed in the CLI interface.

  If your system contains parameters that are not supported in MX-LAN mode in a configuration file, you cannot commit those unsupported attributes. You must remove the settings that are not supported and then commit the configuration. After the successful CLI commit, a system reboot is required for the attributes to become effective. Similarly, if you remove the `network-services lan` statement, the system does not run in MX-LAN mode. Therefore, all of the settings that are supported outside of the MX-LAN mode are displayed and are available for definition in the CLI interface. If your configuration file contains settings that are supported only in MX-LAN mode, you must remove those attributes before you commit the configuration. After the successful CLI commit, a system reboot is required for the CLI parameters to take effect. The Layer 2 Next-Generation CLI configuration settings are supported in MX-LAN mode. As a result, the typical format of CLI configurations might differ in MX-LAN mode.

To configure an IRB interface in a PVLAN bridge domain associated with a virtual switch instance:

1. Create a promiscuous port for the PVLAN.

```
[edit interfaces]
user@host# set interface interface-name unit logical-unit-number family bridge interface-mode
trunk
user@host# set interface interface-name unit logical-unit-number family bridge vlan-id vlan-id
```

2. Create the interswitch link (ISL) trunk port for the PVLAN.

```
[edit interfaces]
user@host# set interface interface-name unit logical-unit-number family bridge interface-mode
```

```
trunk inter-switch-link
user@host# set interface interface-name unit logical-unit-number family bridge vlan-id vlan-id
```

3. Create the isolated port for the PVLAN. The port is identified as an isolated port or a community port, based on the VLAN ID or the list of VLAN IDs to which the interface corresponds. For example, if you configure a port with a VLAN ID of 50, and if you specify a VLAN ID of 50 as the isolated VLAN or tag in the bridge domain, the port is considered as an isolation port.

```
[edit interfaces]
user@host# set interface interface-name unit logical-unit-number family bridge interface-mode
access
user@host# set interface interface-name unit logical-unit-number family bridge vlan-id vlan-id
```

4. Create the community port for the PVLAN. The port is identified as an isolated port or a community port, based on the VLAN ID or the list of VLAN IDs to which the interface corresponds. For example, if you configure a port with a VLAN ID of 50, and if you specify a VLAN ID of 50 as the community VLAN or tag in the bridge domain, the port is considered as a community port.

```
[edit interfaces]
user@host# set interface interface-name unit logical-unit-number family bridge interface-mode
access
user@host# set interface interface-name unit logical-unit-number family bridge vlan-id vlan-id
```

5. Create a virtual switch instance with a bridge domain and associate the logical interfaces.

```
[edit routing-instances]
user@host# set routing-instance-name instance-type virtual-switch
user@host# set routing-instance-name interface interface-name unit logical-unit-number
user@host# set routing-instance-name bridge-domains bridge-domain-name
```

6. Create an IRB interface and specify the IRB interface in the bridge domain associated with the virtual switch instance. IRB provides simultaneous support for Layer 2 bridging and Layer 3 IP routing on the same interface. IRB enables you to route local packets to another routed interface or to another bridge domain that has a Layer 3 protocol configured.

```
[edit]
user@host# set interfaces irb unit logical-unit-number family family-name address ip-address
[edit routing-instances instance-name bridge-domains bridge-domain-name]
user@host# set routing-interface irb unit logical-unit-number
```

7. Specify the primary, isolated, and community VLAN IDs, and associate the VLANs with the bridge domain.

```
[edit routing-instances instance-name bridge-domains bridge-domain-name]
user@host# set vlan-id vlan-id
user@host# set isolated-vlan vlan-id
user@host# set community-vlans [ number number-number ]
```

Example: Configuring an IRB Interface in a Private VLAN on a Single MX Series Router

IN THIS SECTION

For security reasons, it is often useful to restrict the flow of broadcast and unknown unicast traffic and to even limit the communication between known hosts. The private VLAN (PVLAN) feature on MX Series routers allows an administrator to split a broadcast domain into multiple isolated broadcast subdomains, essentially putting a VLAN inside a VLAN.

This example describes how to create an integrated routing and bridging (IRB) interface in a PVLAN bridge domain associated with a virtual switch instance on a single MX Series router:

NOTE: Configuring a voice over IP (VoIP) VLAN on PVLAN interfaces is not supported.

Requirements

This example uses the following hardware and software components:

- One MX Series router in enhanced LAN mode.
- Junos OS Release 15.1 or later for MX Series routers

Before you begin configuring a PVLAN, make sure you have:

- Created and configured the necessary VLANs. See "Configuring VLAN and Extended VLAN Encapsulation" on page 299 and "Enabling VLAN Tagging" on page 277.
- Configured MX240, MX480, and MX960 routers to function in enhanced LAN mode by entering the `network-services lan` statement at the `[edit chassis]` hierarchy level.

Overview and Topology

In a large office with multiple buildings and VLANs, you might need to isolate some workgroups or other endpoints for security reasons or to partition the broadcast domain. This configuration example shows a simple topology to illustrate how to create a PVLAN with one primary VLAN and four community VLANs, as well as two isolated ports.

Assume a sample deployment in which a primary VLAN named VP contains ports, p1, p2, t1, t2, i1, i2, cx1, and cx2. The port types of these configured ports are as follows:

- Promiscuous ports = p1, p2
- ISL ports = t1, t2
- Isolated ports = i1, i2
- Community VLAN = Cx
- Community ports = cx1, cx2

An IRB interface, irb.0, is configured and mapped to the bridge domain in the virtual switch instance.

Bridge domains are provisioned for each of the VLANs, namely, Vp, Vi, and Vcx. Assume the bridge domains to be configured as follows:

Vp—BD_primary_Vp (ports contained are p1, t1, i1, i2, cx1, cx2)

Vi—BD_isolate_Vi (ports contained are p1, t1, *i1, *i2)

Vcx—BD_community_Vcx (ports contained are p1, t1, cx1, cx2)

The bridge domains for community, primary, and isolated VLANs are automatically created by the system internally when you configure a bridge domain with a trunk interface, access interface, or interswitch link. The bridge domains contain the same VLAN ID corresponding to the VLANs. To use bridge domains for PVLANs, you must configure the following additional attributes:

Configuration

IN THIS SECTION

To configure an IRB interface in a PVLAN, perform these tasks:

CLI Quick Configuration

To quickly create and configure a PVLAN and include an IRB interface in a PVLAN bridge domain associated with a virtual switch instance, copy the following commands and paste them into the router terminal window:

Configuring an IRB Interface

```
set interfaces irb unit 0 family inet address 22.22.22.1/24
```

Configuring Promiscuous, ISL, Isolated, and Community Ports

```
set interfaces ge-0/0/9 unit 0 family bridge interface-mode trunk
set interfaces ge-0/0/9 unit 0 family bridge vlan-id 100
set interfaces ge-0/0/13 unit 0 family bridge interface-mode trunk
set interfaces ge-0/0/13 unit 0 family bridge vlan-id 100
set interfaces ge-0/0/10 unit 0 family bridge interface-mode access
set interfaces ge-0/0/10 unit 0 family bridge vlan-id 10
set interfaces ge-0/0/12 unit 0 family bridge interface-mode access
set interfaces ge-0/0/12 unit 0 family bridge vlan-id 10
```

```
set interfaces ge-0/0/1 unit 0 family bridge interface-mode access
set interfaces ge-0/0/1 unit 0 family bridge vlan-id 50
set interfaces ge-0/0/2 unit 0 family bridge interface-mode access
set interfaces ge-0/0/2 unit 0 family bridge vlan-id 50
set interfaces ge-0/0/3 unit 0 family bridge interface-mode access
set interfaces ge-0/0/3 unit 0 family bridge vlan-id 60
set interfaces ge-0/0/4 unit 0 family bridge interface-mode access
set interfaces ge-0/0/4 unit 0 family bridge vlan-id 60
```

Configuring a Virtual Switch Instance With Bridge Domain Interfaces

```
set routing-instances vs-1 instance-type virtual-switch
set routing-instances vs-1 interface ge-0/0/1.0
set routing-instances vs-1 interface ge-0/0/2.0
set routing-instances vs-1 interface ge-0/0/3.0
set routing-instances vs-1 interface ge-0/0/4.0
set routing-instances vs-1 interface ge-0/0/9.0
set routing-instances vs-1 interface ge-0/0/10.0
set routing-instances vs-1 interface ge-0/0/12.0
set routing-instances vs-1 interface ge-0/0/13.0
set routing-instances vs-1 bridge-domains bd1
```

Specify the IRB Interface and Primary, Isolated, and Community VLAN IDs in the Bridge Domain

```
set routing-instances vs1 bridge-domains bd1 vlan-id 100
set routing-instances vs1 bridge-domains bd1 isolated-vlan 10
set routing-instances vs1 bridge-domains bd1 community-vlans [50 60]
set routing-instances vs1 bridge-domains bd1 routing-interface irb.0
```

Procedure

Step-by-Step Procedure

To configure the interswitch link (ISL) for a PVLAN, the PVLAN port types, and secondary VLANs for the PVLAN:

1. Create an IRB interface.

```
[edit interfaces]
user@host# set interfaces irb unit 0 family inet address 22.22.22.1/24
```

2. Create a promiscuous port for the PVLAN.

```
[edit interfaces]
user@host# set ge-0/0/9 unit 0 family bridge interface-mode trunk
user@host# set ge-0/0/9 unit 0 family bridge vlan-id 100
```

3. Create the interswitch link (ISL) trunk port for the PVLAN.

```
[edit interfaces]
user@host# set ge-0/0/13 unit 0 family bridge interface-mode trunk inter-switch-link
user@host# set ge-0/0/13 unit 0 family bridge vlan-id 100
```

4. Create the isolated ports for the PVLAN.

```
[edit interfaces]
user@host# set ge-0/0/10 unit 0 family bridge interface-mode access
user@host# set ge-0/0/10 unit 0 family bridge vlan-id 10
user@host# set ge-0/0/12 unit 0 family bridge interface-mode access
user@host# set ge-0/0/12 unit 0 family bridge vlan-id 10
```

5. Create the community ports for the PVLAN.

```
[edit interfaces]
user@host# set ge-0/0/1 unit 0 family bridge interface-mode access
user@host# set ge-0/0/1 unit 0 family bridge vlan-id 50
user@host# set ge-0/0/2 unit 0 family bridge interface-mode access
user@host# set ge-0/0/2 unit 0 family bridge vlan-id 50
user@host# set ge-0/0/3 unit 0 family bridge interface-mode access
user@host# set ge-0/0/3 unit 0 family bridge vlan-id 60
user@host# set ge-0/0/4 unit 0 family bridge interface-mode access
user@host# set ge-0/0/4 unit 0 family bridge vlan-id 60
```

6. Create a virtual switch instance with a bridge domain and associate the logical interfaces.

```
[edit routing-instances]
user@host# set vs-1 instance-type virtual-switch
user@host# set vs-1 interface ge-0/0/1.0
user@host# set vs-1 interface ge-0/0/2.0
user@host# set vs-1 interface ge-0/0/3.0
user@host# set vs-1 interface ge-0/0/4.0
user@host# set vs-1 interface ge-0/0/9.0
user@host# set vs-1 interface ge-0/0/10.0
user@host# set vs-1 interface ge-0/0/12.0
user@host# set vs-1 interface ge-0/0/13.0
user@host# set vs-1 bridge-domains bd1
```

7. Specify the IRB interface, primary, isolated, and community VLAN IDs, and associate the VLANs with the bridge domain.

```
[edit routing-instances vs1 bridge-domains bd1]
user@host# set vlan-id 100
user@host# set isolated-vlan 10
user@host# set community-vlans [50 60]
user@host# set routing-interface irb.0
```

Results

Check the results of the configuration:

```
[edit]
[interfaces]
     ge-0/0/9 {
         unit 0 {
             family bridge {
     interface-mode trunk;
         vlan-id 100;             Promiscuous port by vlan id
             }
         }
     }

     ge-0/0/13 {
```

```
        unit 0 {
            family bridge {
    interface-mode trunk inter-switch-link;   ISL trunk
    vlan-id 100;
            }
        }
    }


    ge-0/0/10 {
        unit 0 {
            family bridge {
                  interface-mode access;
                  vlan-id 10;              isolated port by vlan ID
             }
        }
    }

    ge-0/0/12 {
        unit 0 {
            family bridge {
                  interface-mode access;
                  vlan-id 10;              isolated port by vlan ID
             }
        }
    }
    ge-0/0/1 {
        unit 0 {
            family bridge {
                  interface-mode access;
                  vlan-id 50;              community port by vlan ID
             }
        }
    }

    ge-0/0/2 {
        unit 0 {
            family bridge {
                  interface-mode access;
                  vlan-id 50;              community port by vlan ID
             }

        }
```

```
    }

    ge-0/0/3 {
        unit 0 {
            family bridge {
                  interface-mode access;
                  vlan-id 60;              community port by vlan ID
             }
        }
    }

    ge-0/0/4 {
        unit 0 {
            family bridge {
                  interface-mode access;
                  vlan-id 60;              community port by vlan ID
             }
        }
    }

    irb {
       unit 0 {
          family inet {
               address 22.22.22.1/24;
          }
    }
}
```

```
[edit]
routing-instances {
    vs-1 {
        instance-type virtual-switch;
        interface ge-0/0/1.0;
        interface ge-0/0/2.0;
        interface ge-0/0/3.0;
        interface ge-0/0/4.0;
        interface ge-0/0/9.0;
        interface ge-0/0/10.0;
        interface ge-0/0/12.0;
        interface ge-0/0/13.0;
```

```
 bridge-domains {
     bd1 {
             vlan-id 100;                /* primary vlan */
                   isolated-vlan 10;
                   community-vlans [50 60]
                   routing-interface irb.0 /* IRB interface */
      }
}
}
```

Verification

IN THIS SECTION

- Verifying That the Private VLAN and Secondary VLANs Were Created | **677**

To confirm that the configuration is working properly, perform these tasks:

Verifying That the Private VLAN and Secondary VLANs Were Created

Purpose

Verify that the primary VLAN and secondary VLANs were properly created on the switch.

Action

Use the `show bridge domain` command:

```
user@host> show bridge domain
Routing instance          Bridge domain            VLAN ID     Interfaces
default-switch                  bd1-primary-100          100
                                                                      ge-0/0/9.0

ge-0/0/10.0
                                                                    ge-0/0/12.0
```

```
                                                                ge-0/0/13.0
                                                                 ge-0/0/1.0
                                                                 ge-0/0/2.0
                                                                 ge-0/0/3.0

                                                                 ge-0/0/4.0

default-switch                    bd1-isolation-10          10
                                                                   ge-0/0/9.0
                                                                ge-0/0/10.0
                                                                 ge-0/0/12.0
                                                                 ge-0/0/13.0


default-switch                  bd1-comunity-50           50
                                                                   ge-0/0/9.0
                                                                    ge-0/0/13.0
                                                                ge-0/0/1.0
                                                                   ge-0/0/2.0

default-switch                   bd1-comunity-60           60
                                                                   ge-0/0/9.0
                                                                    ge-0/0/13.0
                                                                ge-0/0/3.0
                                                                 ge-0/0/4.0
```

Meaning

The output shows that the primary VLAN was created and identifies the interfaces and secondary VLANs associated with it.

CHAPTER 18

Configuring Layer 2 Bridging Interfaces

IN THIS CHAPTER

Layer 2 Bridging Interfaces Overview

Bridging operates at Layer 2 of the OSI reference model while routing operates at Layer 3. A set of logical ports configured for bridging can be said to constitute a bridging domain.

A bridging domain can be created by configuring a routing instance and specifying the instance-type as `bridge`.

Integrated routing and bridging (IRB) is the ability to:

- Route a packet if the destination MAC address is the MAC address of the router and the packet `ethertype` is IPv4, IPv6, or MPLS.
- Switch all multicast and broadcast packets within a bridging domain at layer 2.
- Route a copy of the packet if the destination MAC address is a multicast address and the `ethertype` is IPv4 or IPv6.
- Switch all other unicast packets at Layer 2.
- Handle supported Layer 2 control packets such as STP and LACP.
- Handle supported Layer 3 control packets such as OSPF and RIP.

RELATED DOCUMENTATION

Configuring Layer 2 Bridging Interfaces | 680

Ethernet Interfaces User Guide for Routing Devices

Configuring Layer 2 Bridging Interfaces

Integrated routing and bridging interfaces are logical Layer 3 VLAN interfaces that route traffic between bridge domains (VLANs). So, an IRB logical interface is usually associated with a bridge domain or VLAN. The IRB logical interface also functions as the gateway IP address for the other devices on the same sub-network that are associated with the same VLAN. IRB interfaces support Layer 2 bridging and Layer 3 routing on the same interface. As a result, IRB interfaces enable the router to act both as a router and as a Layer 2 switch at the same time.

NOTE: If the status of all Layer 2 logical interfaces in the bridge domain is down, the status of the `irb` logical interface is also down.

To configure an IRB logical interface:

1. In configuration mode, at the [`edit bridge-domains`] hierarchy level, configure the bridge domain by specifying the name of the bridge and the VLAN ID.

```
[edit bridge-domains]
user@host# set bridge-domain-name vlan-id vlan-id
```

2. Configure an interface in trunk mode and include the interface in the appropriate bridge domain using the `vlan-id-list` command at the [`edit interfaces`] hierarchy level.

```
[edit interfaces]
user@host# set interfacetype-fpc/pic/port vlan-tagging
user@host# set interfacetype-fpc/pic/port unit logical-unit-number family bridge interface-
mode trunk
user@host# set interfacetype-fpc/pic/port unit logical-unit-number family bridge vlan-id-list
vlan-id
```

3. Configure the IRB interface at the [`edit interfaces`] hierarchy level and specify the associated IP address.

```
[edit interfaces]
user@host# set interfaces irb unit logical-unit-number family inet address address
```

4. Configure the IRB interface as the routing interface for the bridge domain at the [`edit bridge-domains`] hierarchy level.

```
[edit bridge-domains]
user@host# set bridge-domain- name vlan-id vlan-id routing-interface irb.logical-interface-
number
```

RELATED DOCUMENTATION

Layer 2 Bridging Interfaces Overview | 680

Example: Configuring the MAC Address of an IRB Interface

IN THIS SECTION

This example shows how to configure the media access control (MAC) address of an integrated routing and bridging (IRB) interface for devices with Modular Port Concentrator (MPC) cards . An IRB interface is a Layer 3 routing interface that is used in a bridge domain or virtual private LAN service (VPLS) routing.

Requirements

This example requires MX Series routers with MPC cards.

Overview

IN THIS SECTION

The assignment of MAC addresses to IRB logical interfaces is supported . The IRB logical interfaces provide support for simultaneous Layer 2 bridging and Layer 3 routing within the same bridge domain. Packets that arrive on an interface of the bridge domain are either switched or routed, based on the destination MAC address of the packet. The packets with the router's Layer 2 virtual MAC address, which is manually configured, are switched to Layer 2 interfaces.

Configuring a MAC address of an IRB logical interface allows the use of a transparent firewall between two VLANs on the same switch. When both VLANs are on the same subnet and traffic from one VLAN needs to go through the firewall to the host on the other VLAN, then the VLAN tag is changed to communicate with the host on the other VLAN.

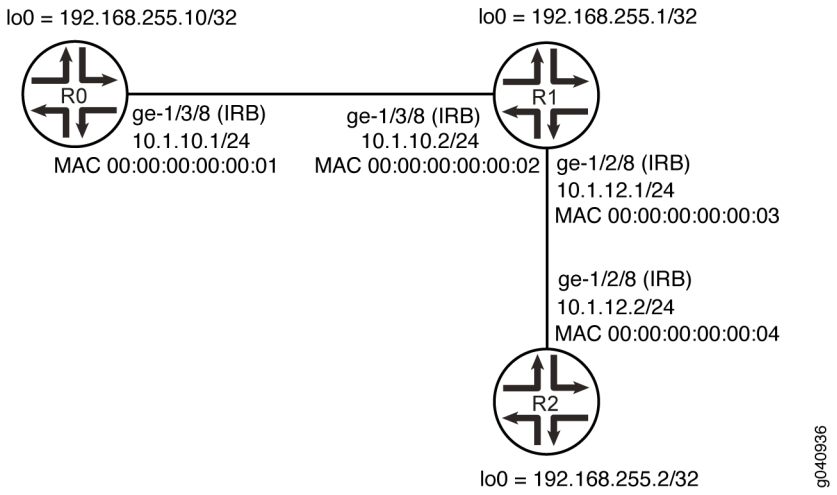
Before the introduction of this feature, if the MAC address of an IRB logical interface was the same for both VLANs, the firewall dropped the traffic. This new feature allows you to configure distinct MAC addresses for different VLANs, which facilitates the exchange of traffic between two VLANs on the same switch.

In case of VPLS multihoming, if there is a failover of the primary provider edge (PE) router to a secondary PE router, the MAC address of an IRB changes. The hosts connected to the customer edge (CE) router must change their Address Resolution Protocol (ARP) for IRB's IP and MAC address. This feature allows you to configure the same MAC address for IRB interfaces in both the primary and secondary PE routers and eliminates the need for changing the ARP binding of the IRB logical interface in CE routers, in case of a failover.

Figure 36 on page 684 shows the sample topology.

Topology

Figure 36: Configuring the MAC Address of an IRB Interface



In this example you configure MAC address of IRB logical interfaces.

Configuration

IN THIS SECTION

CLI Quick Configuration

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

Router R0

```
set interfaces ge-1/3/8 vlan-tagging
set interfaces ge-1/3/8 encapsulation flexible-ethernet-services
set interfaces ge-1/3/8 unit 10 encapsulation vlan-bridge
set interfaces ge-1/3/8 unit 10 vlan-id 10
set interfaces irb unit 10 family inet address 10.1.10.1/24
set interfaces irb unit 10 family mpls
set interfaces irb unit 10 mac 00:00:00:00:00:01
set interfaces lo0 unit 10 family inet address 192.168.255.10/32
set protocols rsvp interface irb.10
set protocols mpls label-switched-path R0-1-R2 to 192.168.255.2
set protocols mpls label-switched-path R0-1-R2 install 192.168.255.2/32 active
set protocols mpls label-switched-path R0-1-R2 no-cspf
set protocols mpls interface irb.10
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 192.168.255.10
set protocols bgp group ibgp neighbor 192.168.255.2
set protocols ospf area 0.0.0.0 interface irb.10
set protocols ospf area 0.0.0.0 interface lo0.10 passive
set protocols ldp interface irb.10
set protocols ldp interface lo0.10
set routing-options autonomous-system 400
set bridge-domains lsbd1 vlan-id 10
set bridge-domains lsbd1 interface ge-1/3/8.10
set bridge-domains lsbd1 routing-interface irb.10
```

Router R1

```
set interfaces ge-1/3/8 vlan-tagging
set interfaces ge-1/3/8 encapsulation flexible-ethernet-services
set interfaces ge-1/3/8 unit 10 encapsulation vlan-bridge
set interfaces ge-1/3/8 unit 10 vlan-id 10
set interfaces ge-1/2/8 vlan-tagging
set interfaces ge-1/2/8 encapsulation flexible-ethernet-services
set interfaces ge-1/2/8 unit 40 encapsulation vlan-bridge
```

```
set interfaces ge-1/2/8 unit 40 vlan-id 40
set interfaces irb unit 20 family inet address 10.1.10.2/24
set interfaces irb unit 20 family mpls
set interfaces irb unit 20 mac 00:00:00:00:00:02
set interfaces irb unit 30 family inet address 10.1.12.1/24
set interfaces irb unit 30 family mpls
set interfaces irb unit 30 mac 00:00:00:00:00:03
set interfaces lo0 unit 20 family inet address 192.168.255.1/32
set protocols rsvp interface irb.20
set protocols rsvp interface irb.30
set protocols mpls interface irb.30
set protocols mpls interface irb.20
set protocols ospf area 0.0.0.0 interface irb.20
set protocols ospf area 0.0.0.0 interface irb.30
set protocols ospf area 0.0.0.0 interface lo0.20 passive
set protocols ldp interface irb.20
set protocols ldp interface irb.30
set protocols ldp interface lo0.20
set routing-options autonomous-system 400
set bridge-domains lsbd2 vlan-id 10
set bridge-domains lsbd2 interface ge-1/3/8.10
set bridge-domains lsbd2 routing-interface irb.20
set bridge-domains lsbd3 vlan-id 40
set bridge-domains lsbd3 interface ge-1/2/8.40
set bridge-domains lsbd3 routing-interface irb.30
```

Router R2

```
set interfaces ge-1/2/8 vlan-tagging
set interfaces ge-1/2/8 encapsulation flexible-ethernet-services
set interfaces ge-1/2/8 unit 40 encapsulation vlan-bridge
set interfaces ge-1/2/8 unit 40 vlan-id 40
set interfaces irb unit 40 family inet address 10.1.12.2/24
set interfaces irb unit 40 family mpls
set interfaces irb unit 40 mac 00:00:00:00:00:04
set interfaces lo0 unit 30 family inet address 192.168.255.2/32
set protocols rsvp interface irb.40
set protocols mpls label-switched-path R2-1-R0 to 192.168.255.10
set protocols mpls label-switched-path R2-1-R0 no-cspf
set protocols mpls interface irb.40
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 192.168.255.2
```

```
set protocols bgp group ibgp neighbor 192.168.255.10
set protocols ospf area 0.0.0.0 interface irb.40
set protocols ospf area 0.0.0.0 interface lo0.30 passive
set protocols ldp interface irb.40
set protocols ldp interface lo0.30
set routing-options autonomous-system 400
set bridge-domains lsbd4 vlan-id 40
set bridge-domains lsbd4 interface ge-1/2/8.40
set bridge-domains lsbd4 routing-interface irb.40
```

Configuring the MAC Address of an IRB Interface

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

NOTE: Repeat this procedure for Juniper Networks Routers R1 and R2, modifying the appropriate interface names, addresses, and any other parameters for each router.

To configure the MAC address of an IRB interface on Router R0:

1. Configure the physical interfaces.

```
[edit interfaces ge-1/3/8]
user@R0# set vlan-tagging
user@R0# set encapsulation flexible-ethernet-services
user@R0# set unit 10 encapsulation vlan-bridge
user@R0# set unit 10 vlan-id 10
```

2. Configure the IRB logical interface.

```
[edit interfaces irb]
user@R0# set unit 10 family inet address 10.1.10.1/24
user@R0# set unit 10 family mpls
user@R0# set unit 10 mac 00:00:00:00:00:01
[edit interfaces]
user@R0# set lo0 unit 10 family inet address 192.168.255.10/32
```

3. Configure the RSVP protocol.

```
[edit protocols rsvp]
user@R0# set interface irb.10
```

4. Configure the MPLS protocol.

```
[edit protocols mpls]
user@R0# set label-switched-path R0-1-R2 to 192.168.255.2
user@R0# set label-switched-path R0-1-R2 install 192.168.255.2/32 active
user@R0# set label-switched-path R0-1-R2 no-cspf
user@R0# set interface irb.10
user@R0# set interface irb.10
```

5. Configure the BGP protocol.

```
[edit protocols BGP]
user@R0# set group ibgp type internal
user@R0# set group ibgp local-address 192.168.255.10
user@R0# set group ibgp neighbor 192.168.255.2
```

6. Configure the OSPF protocol.

```
[edit protocols ospf]
user@R0# set area 0.0.0.0 interface irb.10
user@R0# set area 0.0.0.0 interface lo0.10 passive
```

7. Configure the LDP protocol.

```
[edit protocols ldp]
user@R0# set interface irb.10
user@R0# set interface lo0.10
```

8. Configure the autonomous system (AS) number.

```
[edit routing-options]
user@R0# set autonomous-system 400
```

9. Configure the bridge domains.

```
[edit]
user@R0# set bridge-domains lsbd1 vlan-id 10
user@R0# set bridge-domains lsbd1 interface ge-1/3/8.10
user@R0# set bridge-domains lsbd1 routing-interface irb.10
```

Results

From configuration mode, enter the **show interfaces**, **show protocols** and **show bridge-domains**, commands and confirm your configuration. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R0# show interfaces
ge-1/3/8 {
    unit 10 {
        encapsulation vlan-bridge;
        vlan-id 10;
    }
}
irb {
    unit 10 {
        family inet {
            mtu 1450;
            address 10.1.10.1/24;
        }
        family mpls;
        mac 00:00:00:00:00:01;
    }
}
lo0 {
    unit 10 {
        family inet {
            address 192.168.255.10/32;
```

```
        }
    }
}
user@R0# show protocols
rsvp {
    interface irb.10;
}
    mpls {
        label-switched-path R0-1-R2 {
            to 192.168.255.2;
            install 192.168.255.2/32 active;
            no-cspf;
        }
        interface irb.10;
    }
    bgp {
        group ibgp {
            type internal;
            local-address 192.168.255.10;
            neighbor 192.168.255.2;
        }
    }
    ospf {
        area 0.0.0.0 {
            interface irb.10;
            interface lo0.10 {
                passive;
            }
        }
    }
    ldp {
        interface irb.10;
        interface lo0.10;
    }
user@R0# show bridge-domains
lsbd1 {
    vlan-id 10;
    interface ge-1/3/8.10;
    routing-interface irb.10;
}
```

If you are done configuring the devices, commit the configuration.

```
user@host# commit
```

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying the MAC Address of the IRB Interface

Purpose

Verify that the specified MAC address is assigned to the IRB interface.

Action

From operational mode, run the `show interfaces irb` command on the device.

user@host# **show interfaces irb**

```
Physical interface: irb, Enabled, Physical link is Up
  Interface index: 132, SNMP ifIndex: 505
  Type: Ethernet, Link-level type: Ethernet, MTU: 1514
  Device flags   : Present Running
  Interface flags: SNMP-Traps
  Link type      : Full-Duplex
  Link flags     : None
  Current address: 80:71:1f:c2:58:f0, Hardware address: 80:71:1f:c2:58:f0
  Last flapped   : Never
    Input packets : 0
    Output packets: 0
```

```
Logical interface irb.10 (Index 326) (SNMP ifIndex 634)
  Flags: SNMP-Traps 0x0 Encapsulation: ENET2
  MAC: 00:00:00:00:00:01
  Bandwidth: 1000mbps
  Routing Instance: LS1/default Bridging Domain: lsbd1+10
  Input packets : 55202
  Output packets: 69286
  Protocol inet, MTU: 1450
    Flags: Sendbcast-pkt-to-re, Is-Primary, User-MTU
    Addresses, Flags: Is-Preferred Is-Primary
      Destination: 10.1.10/24, Local: 10.1.10.1, Broadcast: 10.1.10.255
    Addresses, Flags: Is-Preferred
      Destination: 10.1.12/24, Local: 10.1.12.1, Broadcast: 10.1.12.255
  Protocol mpls, MTU: 1500, Maximum labels: 3
    Flags: Is-Primary
  Protocol multiservice, MTU: 1500

Logical interface irb.20 (Index 358) (SNMP ifIndex 635)
  Flags: SNMP-Traps 0x0 Encapsulation: ENET2
  MAC: 00:00:00:00:00:02
  Bandwidth: 1000mbps
  Routing Instance: LS2/default Bridging Domain: lsbd2+10
  Input packets : 66044
  Output packets: 68464
  Protocol inet, MTU: 1450
    Flags: Sendbcast-pkt-to-re, Is-Primary, User-MTU
    Addresses, Flags: Is-Preferred Is-Primary
      Destination: 10.1.10/24, Local: 10.1.10.2, Broadcast: 10.1.10.255
    Addresses, Flags: Is-Preferred
      Destination: 10.1.12/24, Local: 10.1.12.2, Broadcast: 10.1.12.255
  Protocol mpls, MTU: 1500, Maximum labels: 3
    Flags: Is-Primary
  Protocol multiservice, MTU: 1500

Logical interface irb.30 (Index 360) (SNMP ifIndex 636)
  Flags: SNMP-Traps 0x0 Encapsulation: ENET2
  MAC: 00:00:00:00:00:03
  Bandwidth: 1000mbps
  Routing Instance: LS2/default Bridging Domain: lsbd3+40
  Input packets : 26948
  Output packets: 53605
  Protocol inet, MTU: 1500
```

```
      Flags: Sendbcast-pkt-to-re
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: 10.1.12/24, Local: 10.1.12.2, Broadcast: 10.1.12.255
      Addresses, Flags: Is-Preferred
        Destination: 10.1.10/24, Local: 10.1.10.1, Broadcast: 10.1.10.255
    Protocol mpls, MTU: 1500, Maximum labels: 3
    Protocol multiservice, MTU: 1500

  Logical interface irb.40 (Index 355) (SNMP ifIndex 632)
    Flags: SNMP-Traps 0x0 Encapsulation: ENET2
    MAC: 00:00:00:00:00:04
    Bandwidth: 1000mbps
    Routing Instance: LS3/default Bridging Domain: lsbd4+40
    Input packets : 40575
    Output packets: 31128
    Protocol inet, MTU: 1500
      Flags: Sendbcast-pkt-to-re, Is-Primary
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: 10.1.12/24, Local: 10.1.12.1, Broadcast: 10.1.12.255
    Protocol mpls, MTU: 1500, Maximum labels: 3
      Flags: Is-Primary
    Protocol multiservice, MTU: 1500
```

Meaning

The output shows the manually configured MAC address in the MAC field.

NOTE: If you did not configure the MAC address for a logical interface, the output does not include this value. However, the device uses the MAC address of the physical interface during data transmission.

RELATED DOCUMENTATION

- mac
- Active-Active Bridging and VRRP over IRB Functionality Overview

19

CHAPTER

Configuring Layer 2 Virtual Switch Instances

IN THIS CHAPTER

Layer 2 Virtual Switch Instances

IN THIS SECTION

Understanding Layer 2 Virtual Switches Instances

Benefit of Using Layer 2 Virtual Switch Instances:

- Splitting Layer 2 traffic using virtual switch instances allows you to more logically organize your Layer 2 traffic into multiple “virtual” Layer 2 networks.

At Layer 2, you can group one or more VLANs into a single routing instance to form a virtual switch instance. A virtual switch instance is composed of VLANs. The virtual switch instance isolates a LAN segment and contains most Layer 2 functions, such as spanning-tree protocol instances and VLAN ID spaces, into it’s own smaller, logical network. Splitting Layer 2 traffic using virtual switch instances allows you to more logically organize your Layer 2 traffic into multiple “virtual” Layer 2 networks.

A default virtual switch, called default-switch, is automatically created when a virtual switch is configured. All Layer 2 traffic not assigned to a VLAN in a virtual switch automatically becomes part of the default virtual switch.

You can configure a virtual switch to participate only in Layer 2 bridging and optionally to perform Layer 3 routing. In addition, you can configure spanning-tree protocols (STPs) within the virtual switch to prevent forwarding loops. For more information about how to configure Layer 2 logical ports on an interface, see the Junos OS Network Interfaces Library for Routing Devices.

You can associate one or more logical interfaces configured as trunk interfaces with a virtual switch. A trunk interface, or Layer 2 trunk port, enables you to configure a *logical interface* to represent multiple VLANs on the physical interface. For more information about how to configure trunk interfaces, see the Junos OS Network Interfaces Library for Routing Devices.

You can also configure Layer 2 forwarding and learning properties for the virtual switch.

Configuring a Layer 2 Virtual Switch on an EX Series Switch

A Layer 2 virtual switch, which isolates a LAN segment with its spanning-tree protocol instance and separates its VLAN ID space, filters and forwards traffic only at the data link layer. Each VLAN consists of a set of logical ports that participate in Layer 2 learning and forwarding. A virtual switch represents a Layer 2 network.

Two main types of interfaces are used in virtual switch hierarchies:

- Layer 2 logical interface—This type of interface uses the VLAN-ID as a virtual circuit identifier and the scope of the VLAN-ID is local to the interface port. This type of interface is often used in service-provider-centric applications.
- Access or trunk interface—This type of interface uses a VLAN-ID with global significance. The access or trunk interface is implicitly associated with VLANs based on VLAN membership. Access or trunk interfaces are typically used in enterprise-centric applications.

NOTE: The difference between access interfaces and trunk interfaces is that access interfaces can be part of one VLAN only and the interface is normally attached to an end-user device (packets are implicitly associated with the configured VLAN). In contrast, trunk interfaces multiplex traffic from multiple VLANs and usually interconnect switches.

To configure a Layer 2 virtual switch, include the following statements:

```
[edit]
routing-instances {
    routing-instance-name (
        instance-type virtual-switch;
        vlans vlan-name{
            vlan-id (all | none | number);
                [...configure optional VLAN parameters]
        }
    }
}
```

To enable a virtual switch, you must specify `virtual-switch` as the `instance-type`.

The VLANs that are specified with the `vlan-id` statement are included in the virtual switch.

You can configure other optional VLAN parameters in the virtual switch.

Configuring a Layer 2 Virtual Switch with a Layer 2 Trunk Port

You can associate one or more Layer 2 trunk interfaces with a virtual switch.

A virtual switch configured with a Layer 2 trunk port also supports IRB within a VLAN. IRB provides simultaneous support for Layer 2 bridging and Layer 3 IP routing on the same interface. Only an interface configured with the `interface-mode (access | trunk)` statement can be associated with a virtual switch. An access interface enables you to accept packets with no VLAN identifier.

In addition, you can configure Layer 2 learning and forwarding properties for the virtual switch.

To configure a virtual switch with a Layer 2 trunk interface, include the following statements:

```
[edit]
routing-instances {
    routing-instance-name {
        instance-type virtual-switch;
        interface interface-name;
        vlans name{
            vlan-id (all | none | number);
                [...configure optional VLAN parameters]
        }
    }
}
```

20

CHAPTER

Configuring Link Layer Discovery Protocol

IN THIS CHAPTER

LLDP Overview

The Link Layer Discovery Protocol (LLDP) is an industry-standard, vendor-neutral method to allow networked devices to advertise capabilities, identity, and other information onto a LAN. The Layer 2 protocol, detailed in IEEE 802.1AB-2005, replaces several proprietary protocols implemented by individual vendors for their equipment.

LLDP allows network devices that operate at the lower layers of a protocol stack (such as Layer 2 bridges and switches) to learn some of the capabilities and characteristics of LAN devices available to higher layer protocols, such as IP addresses. The information gathered through LLDP operation is stored in a network device and is queried with SNMP. Topology information can also be gathered from this database.

Some of the information that can be gathered by LLDP (only minimal information is mandatory) is:

- System name and description
- Port name and description
- VLAN name and identifier
- IP network management address
- Capabilities of the device (for example, switch, router, or server)
- MAC address and physical layer information
- Power information
- Link aggregation information

LLDP frames are sent at fixed intervals on each port that runs LLDP. LLDP protocol data units (LLDP PDUs) are sent inside Ethernet frames and identified by their destination Media Access Control (MAC) address (`01:80:C2:00:00:0E`) and Ethertype (`0x88CC`). Mandatory information supplied by LLDP is chassis ID, port ID, and a time-to-live value for this information.

RELATED DOCUMENTATION

Configuring LLDP | **700**

Tracing LLDP Operations | **711**

Example: Configuring LLDP | **704**

LLDP Operational Mode Commands | **710**

Configuring LLDP

You configure LLDP by including the `lldp` statement and associated parameters at the `[edit protocols]` hierarchy level. The complete set of LLDP statements follows:

```
lldp {
    advertisement-interval seconds;
    (disable | enable);
    hold-multiplier number;
    chassis-id {
    chassis-id-type (chassis-component | interface-alias | interface-name | locally-assigned |
mac-address | network-address | port-component);
    chassis-id-value chassis-id-value;
    }
    interface (all | [interface-name]) {
        dest-mac-type;
        (disable | enable);
        power-negotiation <(disable | enable)>;
        tlv-filter;
        tlv-select;
        trap-notification (disable | enable);
    }
    dest-mac-type destination-mac-address;
    lldp-configuration-notification-interval seconds;
    lldp-tx-fast-init;
    management-address ip-management-address; | management-interface interface-name;
    mau-type;

    neighbour-port-info-display (port-description | port-id);
    port-description-type (interface-alias | interface-description);
    port-id-subtype (interface-name | locally-assigned);
    ptopo-configuration-maximum-hold-time seconds;
    ptopo-configuration-trap-interval seconds;
    tlv-filter;
    tlv-select;
    traceoptions {
        file filename <files number> <size maximum-file-size> <world-readable | no-world-
readable>;
        flag flag <disable>;
    }
```

```
    transmit-delay seconds;
    vlan-name-tlv-option (name | vlan-id);
}
```

The following statements have default values:

- advertisement-interval—The default value is 30 seconds. The allowable range is from 5 through 32768 seconds.
- chassis-id-type—The default value is MAC address.
- chassis-id-value—The default value is system-mac.
- hold-multiplier—The default values is 4. The allowable range is from 2 through 10.
- ptopo-configuration-maximum-hold-time—The default value is 300 seconds. The allowable range is from 1 through 2147483647 seconds.
- transmit-delay—The default values is 2 seconds. The allowable range is from 1 through 8192 seconds.

The following statements must be explicitly configured:

- lldp-configuration-notification-interval—The allowable range is from 5 through 3600 seconds. The default value is 5.
- ptopo-configuration-trap-interval—The allowable range is from 0 through 3600 seconds. The default value is 0.

By default, LLDP is disabled, and user must configure it using [set protocols lldp interface (all | *interface-name*)] to use the LLDP services. If it is enabled for all interfaces, you can disable LLDP on specific interfaces.

NOTE: The interface-name must be the physical interface (for example, ge-1/0/0) and not a logical interface (unit).

Starting in Junos OS Release 19.4R2, you can configure the LLDP on redundant Ethernet (reth) interfaces. Use the set protocol lldp interface *<reth-interface>* command to configure LLDP on reth interface.

Starting in Junos OS Release 22.1R1, you can configure alternate LLDP destination mac addresses. If no configuration is provided, then the packets are sent to the nearest-bridge mac-address, which is 01:80:c2:00:00:0e. Use the set protocols lldp dest-mac-type *<mac-type>* statement to configure packets sent out of all interfaces. And use the set protocols lldp interface *<intf-name>* dest-mac-type *<mac-type>* statement for packets sent out of a specific interface.

- To configure LLDP on all interfaces:

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

- To configure LLDP on a specific interface:

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

To disable LLDP, include the `disable` option:

- To disable LLDP on all interfaces:

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

- To disable LLDP on a specific interface:

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

The advertisement interval determines the frequency that an LLDP interface sends LLDP advertisement frames. The default value is 30 seconds. The allowable range is from 5 through 32768 seconds. You adjust this parameter by including the `advertisement-interval` statement at the `[edit protocols lldp]` hierarchy level.

The hold multiplier determines the multiplier to apply to the advertisement interval. The resulting value in seconds is used to cache learned LLDP information before discard. The default value is 4. When used with the default advertisement interval value of 30 seconds, this makes the default cache lifetime 120 seconds. The allowable range of the hold multiplier is from 2 through 10. You adjust this parameter by including the `hold-multiplier` statement at the `[edit protocols lldp]` hierarchy level.

The transmit delay determines the delay between any two consecutive LLDP advertisement frames. The default value is 2 seconds. The allowable range is from 1 through 8192 seconds. You adjust this parameter by including the `transmit-delay` statement at the `[edit protocols lldp]` hierarchy level.

The physical topology configuration maximum hold time determines the time interval for which an agent device maintains physical topology database entries. The default value is 300 seconds. The allowable

range is from 1 through 2147483647 seconds. You adjust this parameter by including the `ptopo-configuration-maximum-hold-time` statement at the `[edit protocols lldp]` hierarchy level.

The LLDP configuration notification interval determines the period for which trap notifications are sent to the SNMP Master Agent when changes occur in the database of LLDP information. The allowable range is from 5 through 3600 seconds. You adjust this parameter by including the `lldp-configuration-notification-interval` statement at the `[edit protocols lldp]` hierarchy level.

The physical topology configuration trap interval determines the period for which trap notifications are sent to the SNMP Master Agent when changes occur in the global physical topology statistics. This capability is disabled by default. The allowable range is from 0 (disabled) through 3600 seconds. The LLDP agent sends traps to the SNMP Master Agent if this interval has a value greater than 0 and there is any change during the `lldp-configuration-notification-interval` trap interval. You adjust this parameter by including the `ptopo-configuration-trap-interval` statement at the `[edit protocols lldp]` hierarchy level.

You can specify the destination MAC address for LLDP. The options are `nearest-bridge`, `nearest-customer-bridge`, and `nearest-non-tpmr-bridge`.

By default, the management interface of the device is used in the management address TLV of the LLDP PDU. You can configure the management address or the management interface for LLDP.

If you configure the management address, that address is sent in the management address TLV. If you configure the management interface, the IP address of the management interface is sent in the TLV. If the management interface does not have an IP address, the default IP address of the device's management interface is sent in the TLV. If the device's management interface does not have an IP address, the MAC address of the management interface is sent in the TLV.

By default, LLDP generates the SNMP index of the interface for the port ID Type, Length, and Value (TLV). When the `interface-name` statement is configured on the remote LLDP neighbor, the `show lldp neighbors` command output displays the interface name in the `Port ID` field rather than the SNMP index of the interface, which is displayed by default. If you change the default behavior of generating the SNMP index of the interface as the Port ID TLV, you can reenable the default behavior by including the `locally-assigned` statement at the `[edit protocols lldp port-id-subtype]` hierarchy level.

Change History Table

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

Release	Description
19.4R2	Starting in Junos OS Release 19.4R2, you can configure the LLDP on redundant Ethernet (reth) interfaces. Use the `set protocol lldp interface` *<reth-interface>* command to configure LLDP on reth interface.
15.1R7	

RELATED DOCUMENTATION

LLDP Overview | 699

Tracing LLDP Operations | 711

Example: Configuring LLDP | 704

Example: Configuring LLDP

The following example configures LLDP on interface ge-1/1/1 but disables LLDP on all other interfaces, explicitly configures the default values for all automatically enabled features, and configures a value of 30 seconds for the LLDP configuration notification interval and a value of 30 seconds for the physical topology trap interval.

```
[edit]
protocols {
    lldp {
        advertisement-interval 30;
        hold-multiplier 4;
        interface all {
            disable;
        }
        interface ge-1/1/1;
        lldp-configuration-notification-interval 30;
        ptopo-configuration-maximum-hold-time 300;
        ptopo-configuration-trap-interval 30;
        transmit-delay 2;
    }
}
```

You verify operation of LLDP with several show commands:

- show lldp <detail>
- show lldp neighbors *interface-name*
- show lldp statistics *interface-name*
- show lldp local-information
- show lldp remote-global-statisitics

You can clear LLDP neighbor information or statistics globally or on an interface:

- `clear lldp neighbors` *interface-name*
- `clear lldp statistics` *interface-name*

You can display basic information about LLDP with the `show lldp detail` command:

```
user@host> show lldp detail
LLDP                     : Enabled
Advertisement interval   : 30 Second(s)
Transmit delay           : 2 Second(s)
Hold timer               : 4 Second(s)
Notification interval    : 30 Second(s)
Config Trap Interval     : 300 Second(s)
Connection Hold timer    : 60 Second(s)


Interface      LLDP       Neighbor count
ge-1/1/1       Enabled    0

LLDP basic TLVs supported:
Chassis identifier, Port identifier, Port description, System name, System description, System
capabilities, Management address.

LLDP 802 TLVs supported:
Link aggregation, Maximum frame size, MAC/PHY Configuration/Status, Port VLAN ID, Port VLAN name.
```

For more details about the output of these commands, see the CLI Explorer.

RELATED DOCUMENTATION

LLDP Overview | **699**

Configuring LLDP | **700**

Tracing LLDP Operations | **711**

Configuring the Transmission of Maximum VLAN Name TLVs in LLDP

IN THIS SECTION

This topic discusses the support of transmitting maximum number of VLAN Name TLVs in Link Layer Discovery Protocol (LLDP) protocol data units (PDUs).

Generally, an LLDP PDU can incorporate a maximum of 5 VLAN Name TLVs. To advertise more than 5 VLAN Name TLVs, run the `set protocols lldp interface <name> transmit-max-vlan-name-tlv` command. This command is a one-time switch which enables the support of transmitting maximum number of VLAN Name TLVs. However, to disable the support and reset the system to limit up to 5 VLAN Name TLVs, run the `delete protocols lldp interface <name> transmit-max-vlan-name-tlv` command.

NOTE: The *<name>* in the above commands indicates the interface name.

The maximum number of VLAN Name TLVs that can be carried is calculated dynamically. The number is based on the maximum transmission unit (MTU) capacity of the PDU while considering the other necessary TLVs that need to be carried in the PDU.

Considerations for Enabling Transmission of Maximum VLAN Name TLVs in LLDP PDUs

Consider the following conditions before enabling the transmission of maximum VLAN Name TLVs in LLDP PDUs:

- In a default scenario, wherein VLANs configured on the interface have IDs 2 to 1000, with a default MTU value of 1500, 68 VLAN Name TLVs are transmitted.
- If MTU of the interface is increased to 9192 which is the maximum, then 523 VLAN Name TLVs are transmitted. The number of VLAN Name TLVs may vary based on the size of VLAN ID. That is, if the

VLAN IDs are more than a single digit, then the maximum number of VLAN Name TLVs, which are transmitted decreases.

- If the `vlan-name-tlv-option` *`name`* is selected to be transmitted in the VLAN Name TLV, the number of transmitted TLVs changes based on the size of the names of the VLANs. By default, VLAN Name is transmitted as “vlan-ID#”. Therefore, the number of VLAN Name TLVs may increase, or decrease depending on the VLAN names being larger or smaller compared to the default name.
- The VLANs that are transmitted is decided by the VLAN Name TLV key. By default, it is decided by the VLAN ID. For example, if VLANs are configured from 101 to 200, the first 68 VLANs are transmitted.
- Each LLDP PDU transmits the same VLAN Name TLVs until new VLANs are added or deleted, or if the MTU is changed. If new VLANs are added or deleted, then the next set of TLVs are formed based on the conditions mentioned above.
- The maximum VLAN Name length which can be accommodated in the VLAN Name TLV is 34 bytes. Therefore, if the VLAN Name length is more than 32 bytes, that vlan is eliminated form the list.
- As an example, the table below contains the maximum number of VLAN Name TLVs which can be sent in a given scenario. In the table, the configured VLAN IDs start from 101 and go till 200, which means that the default VLAN Name has a length of 10 bytes, and the maximum VLAN Name that is configured is with the length of 32 bytes. The MTU is set to the default value of 1500.

Table 101: Maximum Number of VLAN Name TLVs

Variations	TLVs	Data Size	Maximum Number of VLAN Name TLVs	
			Having `vlan-name-tlv-option` set as vlan-id (each of length - 15 bytes)	Having `vlan-name-tlv-option` set as Name and VLAN size of 32 bytes (each VLAN Name TLV of length - 39 bytes)
Example 1	Mandatory TLVs: • Chassis id- 7 bytes • Port Id- 4 bytes • TTL- 2 bytes • End tlv- 0 bytes	13 bytes	65	27
	Port Description TLV	8 bytes		
	System Name TLV	25 bytes		
	System Description TLV	171 bytes		
	System Capabilities TLV	7 bytes		
	Management Address TLV	8 bytes		
	Mac-PHY status TLV	9 bytes		

Table 101: Maximum Number of VLAN Name TLVs *(Continued)*

Variations	TLVs	Data Size	Maximum Number of VLAN Name TLVs	
			Having `vlan-name-tlv-option` set as vlan-id (each of length - 15 bytes)	Having `vlan-name-tlv-option` set as Name and VLAN size of 32 bytes (each VLAN Name TLV of length - 39 bytes)
	Link Aggregation subtype 3 and Subtype 7	9 bytes + 9 bytes		
	Max Frame size TLV	6 bytes		
	2 Juniper Specific TLVs	16 bytes each		
	Port Vlan-ID TLV	6 bytes		
Example 2 - Includes TLVs in Example 1 with the addition of...	3 DCBX TLVs	25 bytes + 25 bytes + 6 bytes	60	25
	LLDP MED Capabilities TLV	7 bytes		
Example 3 - Includes TLVs in Example 2 with the addition of...	5 Management Address TLVs of type IPV6	36 bytes	50	20
	Power via MDI	12 bytes		
	Extended power via MDI	7 bytes		

RELATED DOCUMENTATION

- show lldp
- show lldp neighbors-vlan-name-tlv-list interface
- LLDP Overview | 699

LLDP Operational Mode Commands

Table 102 on page 710 summarizes the command-line interface (CLI) commands you can use to monitor and troubleshoot the Link Layer Discovery Protocol (LLDP) protocol. Commands are listed in alphabetical order.

Table 102: LLDP Operational Mode Commands

Task	Command
Clear LLDP neighbor information.	*clear lldp neighbors*
Clear LLDP statistics.	*clear lldp statistics*
Display basic LLDP information.	*show lldp*
Display LLDP local information.	*show lldp local-information*
Display LLDP neighbor information.	*show lldp neighbors*
Display LLDP remote global statistics.	*show lldp remote-global-statistics*
Display LLDP statistics.	*show lldp statistics*

RELATED DOCUMENTATION

- LLDP Overview | 699
- Configuring LLDP | 700

Tracing LLDP Operations

To trace LLDP operational traffic, you can specify options in the global `traceoptions` statement included at the `[edit routing-options]` hierarchy level, and you can specify LLDP-specific options by including the `traceoptions` statement:

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

You can include this statement at the following hierarchy levels:

- `[edit protocols lldp]`
- `[edit routing-instances routing-instance-name protocols lldp]`

You can specify the following LLDP-specific options in the LLDP `traceoptions` statement:

- `all`—Trace all operations.
- `config`—Log configuration events.
- `interface`—Trace interface update events.
- `protocol`—Trace protocol information.
- `rtsock`—Trace real-time socket events.
- `vlan`—Trace VLAN update events.

NOTE: Use the trace flag `all` with caution. This flag may cause the CPU to become very busy.

For general information about tracing and global tracing options, see the statement summary for the global `traceoptions` statement in the Junos OS Routing Protocols Library for Routing Devices.

RELATED DOCUMENTATION

21

CHAPTER

Configuring Layer 2 Protocol Tunneling

IN THIS CHAPTER

Layer 2 Protocol Tunneling (L2PT)

SUMMARY

Use Layer 2 protocol tunneling (L2PT) to tunnel supported Layer 2 protocols across a network to devices that are not part of the local broadcast domain.

IN THIS SECTION

Understanding Layer 2 Protocol Tunneling

IN THIS SECTION

Juniper Networks Ethernet switches and routers use Layer 2 protocol tunneling (L2PT) to send Layer 2 protocol data units (PDUs) across the network and deliver them to devices that are not part of the local broadcast domain. This feature is useful when you want to run Layer 2 protocols on a network that includes switches located at remote sites that are connected across a service provider network.

You can also use L2PT to tunnel protocols between two locally-connected user-to-network interfaces (UNIs) in the same broadcast domain, but in that case, the device floods protocol packets in the VLAN instead of rewriting the packets with the tunnel MAC address.

Use Feature Explorer to confirm platform and release support for specific features.

Review the "Platform-Specific L2PT Behavior" on page 733 section for notes related to your platform.

Benefits of L2PT

- Enables you to run supported Layer 2 protocols in a tunnel across a service provider network to remote sites.
- Provides a single spanning-tree protocol domain for subscribers across a service provider network.

How L2PT Works

L2PT works by encapsulating Layer 2 PDUs, tunneling them across a service provider network, and decapsulating them for delivery to their destination switches. The ingress service provider edge (PE) device encapsulates Layer 2 PDUs by rewriting the PDUs' destination media access control (MAC) addresses before forwarding them onto the service provider network. The devices in the service provider network treat these encapsulated PDUs as multicast Ethernet packets. Upon receipt of these PDUs, the egress PE devices decapsulate them by replacing the destination MAC addresses with the address of the Layer 2 protocol that is being tunneled before forwarding the PDUs to their destination devices.

When a PE port configured for Layer 2 protocol tunneling receives a control packet for a supported Layer 2 protocol, the PE device rewrites the multicast destination MAC address with the predefined multicast tunnel MAC address 01:00:0C:CD:CD:D0. The PE device then sends the modified packet onto the provider network. The packet travels across the provider network transparently across the service provider network with the tunnel MAC address. All devices on the provider network treat these packets as multicast Ethernet packets and deliver them to all PE devices for the customer. The egress PE devices receive all the control PDUs with the tunnel MAC address, identify the packet type by doing deeper packet inspection, and replace the destination MAC address with the appropriate destination MAC address. The egress PE devices send out the modified PDUs to the customer PE devices, and the original MAC address is restored when the packets reach the destination ports.

The L2PT protocol is valid for all types of packets, such as untagged, tagged, and Q-in-Q tagged packets.

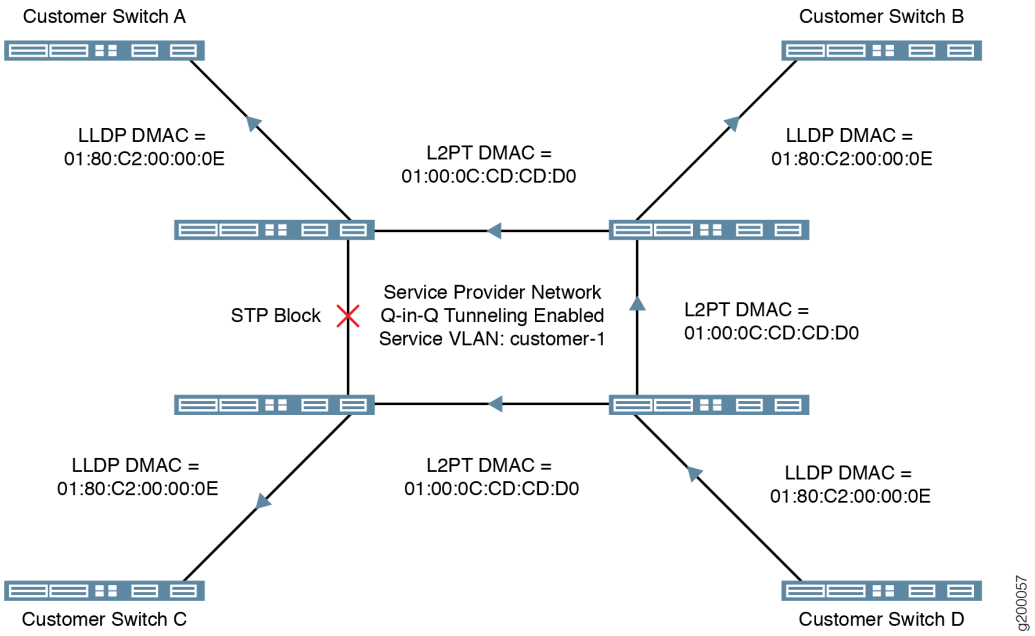
If a PE device receives a packet on a tunnel interface that already has a destination MAC address of 01:00:0C:CD:CD:D0, the device puts the port into an error state and shuts down the port. You can clear this error condition on an interface using the CLI by entering the `clear error mac-rewrite interface` *`interface-name`* command on the following devices that support L2PT:

- MX Series and ACX Series routers

- EX Series switches that use Enhanced Layer 2 Software (ELS)
- QFX Series switches

Figure 37 on page 716 illustrates an example of the L2PT process with EX Series switches in a service provider network that are configured to tunnel LLDP packets on a service VLAN with Q-in-Q tunneling enabled.

Figure 37: L2PT LLDP Example



1. Customer Switch D sends an LLDP PDU to the service provider network that is ultimately intended for the other switches in the customer network.
2. The receiving provider switch rewrites the LLDP destination MAC address with the L2PT destination MAC address, and sends the frame with the encapsulated LLDP PDU to the other switches in the service provider network.
3. When the other service provider switches receive the frame, they detect the L2PT destination MAC address, restore the LLDP destination MAC address, and forward it to Customer Switches A, B, and C.

VLAN Configuration Requirements for Configuring L2PT on Switches

On switches, you enable L2PT on a per-VLAN basis. When you enable L2PT for a particular Layer 2 protocol on a VLAN, all access interfaces are considered to be customer-facing interfaces and all trunk

interfaces are considered to be service provider network-facing interfaces. You cannot configure the specified protocol on the access interfaces. L2PT only acts on logical interfaces with family `ethernet-switching`. The switch floods L2PT PDUs to all trunk and access ports within a given S-VLAN.

NOTE: Access interfaces in an L2PT-enabled VLAN should not receive L2PT-tunneled PDUs. If an access interface does receive L2PT-tunneled PDUs, there might be a loop in the network, and the device will shut down the interface.

If the switch receives untagged or priority-tagged Layer 2 control PDUs to be tunneled, then you must configure the switch to map untagged and priority-tagged packets to an L2PT-enabled VLAN. For more information on assigning untagged and priority-tagged packets to VLANs, see "Understanding Q-in-Q Tunneling and VLAN Translation " on page 941 and "Configuring Q-in-Q Tunneling on EX Series Switches" on page 963.

Layer 2 Control Protocol Tunneling in Layer 2 VPN

Layer 2 circuit cross-connect (CCC) establishes transparent connections between two logical interfaces of the same kind. Therefore, packets received on the first interface should be transmitted out the second interface, and vice versa. On PTX devices, this works as expected for data packets. However, control packets such as LACP do not get transmitted over the Layer 2 circuit, and instead go to Routing Engine for processing.

In a Layer 2 VPN network with a CCC configured between two provider edge (PE) devices, you can enable certain Layer 2 control packets such as LACP or LLDP between the two customer edge (CE) devices as well as between a CE and its directly connected PE device. If you configure the relevant protocol on the interface between PE and its local CE device, they are consumed locally. Else they will be tunneled.

Layer 2 Control Protocol (L2CP) Transparent Tunneling

Layer 2 Control Protocol (L2CP) Transparent Tunneling forwards L2CP packets transparently in hardware unless a specific protocol is configured on the incoming interface. This feature prevents unnecessary discarding of L2CP Bridge Protocol Data Units (BPDUs), improving network performance and the transit of L2 protocol traffic.

Benefits of L2CP Transparent Tunneling

- **Transparent Forwarding** - By default, network devices forward L2CP packets in hardware, which accelerates L2 protocol traffic, reduces delays, streamlines packet flow, and prevents unnecessary discards.

- **Protocol-Specific Handling** - The network devices manages BPDUs according to configured protocols and redirects them to the CPU only when explicitly defined. This approach optimizes CPU utilization.
- **MEF Compliance** - The system provides a Metro Ethernet Forum (MEF) standard forwarding profile. Using a dedicated CLI command enables this profile. This configuration ensures compliance with industry standards and provides flexibility in packet handling.

Configuring MEF Standard Forwarding Profile

The Metro Ethernet Forum (MEF) specifies the rules for processing L2CP Ethernet frame when the packets arrive at the L2CP decision point on the user network interface (UNI). The rules provide the mechanism for transparently passing the L2CP packets.

To configure the MEF standard forwarding profile for L2CP packets, use the command `MEF-forwarding-profile` at the `[edit system packet-forwarding-options]`. This command configures the system to handle L2CP packets in compliance with MEF standards. Configuring this command restarts the management process (mgd).

Configure Layer 2 Protocol Tunneling

NOTE: This topic applies to routers, QFX Series switches, and EX Series switches that support the Enhanced Layer 2 Software (ELS) configuration style. Use Feature Explorer to check if your EX Series switch supports ELS.

To configure Layer 2 protocol tunneling (L2PT) on EX Series switches that do not use ELS, see "Configure Layer 2 Protocol Tunneling on EX Series Switches Without ELS Support" on page 724.

To configure L2PT, you enable MAC address rewriting for Layer 2 protocol tunneling, which installs a destination multicast tunnel MAC address in the MAC table. At the same time, you select the Layer 2 protocol to be tunneled from the list of available options for the type of switch you are configuring (see *protocol*).

By default, the device rewrites the multicast destination MAC address with the predefined multicast tunneling MAC address 01:00:0C:CD:CD:D0 in the MAC table. You can optionally specify a different multicast MAC address.

Use the following guidelines when you configure L2PT:

- Layer 2 protocol tunneling must be configured on the interfaces at both ends of the tunnel.

- You can enable Layer 2 protocol tunneling for untagged interfaces and single-identifier tagged interfaces only, not for double-identifier tagged interfaces.

  For single-identifier tagged ports, configure a *logical interface* with the native VLAN identifier. This configuration associates the untagged control packets with a logical interface.

  When you enable L2PT for a protocol on one user-to-network interface (UNI) in a bridge domain or VLAN, you should also configure all UNIs in the bridge domain or VLAN to tunnel the same protocol for consistent behavior. In that case, those UNIs can receive non-tunneled packets, and tunneled packets are forwarded through the network-to-network interfaces (NNIs).

If you are configuring a QFX Series switch or an EX Series switch, you must configure and enable Q-in-Q tunneling (802.1Q VLAN encapsulation) before you can configure L2PT. To configure L2PT on a specific interface, you must first configure a Q-in-Q on that interface or group of interfaces. This requires configuring the tag protocol ID (TPID). L2PT supports only the default TPID of 0x8100.

To configure Q-in-Q tunneling:

- For QFX Series switches, see "Configuring Q-in-Q Tunneling on QFX Series Switches" on page 952.
- For EX9200 switches, see:
  - "Configuring VLAN Encapsulation" on page 269
  - "Configuring Inner and Outer TPIDs and VLAN IDs" on page 367
  - "Stacking a VLAN Tag" on page 359.
- For other EX Series switches that support ELS, see "Configuring Q-in-Q Tunneling on EX Series Switches with ELS Support" on page 954.

NOTE: L2PT supports only the default tag protocol ID (TPID) of 0x8100.

1. (Optional) Specify the rewrite MAC address.

   By default, MAC address rewriting installs the destination multicast tunnel MAC address 01:00:0C:CD:CD:D0 in the MAC table. You can optionally specify a different multicast MAC address. You can set any non-reserved multicast MAC address.

```
[edit protocols]
user@device# set layer2-control mac-rewrite tunnel-destination-mac mac-address
```

2. To configure L2PT for a single Layer 2 protocol on a specified interface:

```
[edit protocols]
user@device# set layer2-control mac-rewrite interface interface-name protocol protocol-name
```

> **NOTE**: If you enable an L2 protocol for MAC rewrite on an interface, all traffic belonging to the protocol is tunneled. For this reason, you cannot also enable that protocol on that interface.

3. If multiple logical interfaces are enabled on the physical interface where you just configured L2PT, you must also include this configuration:

```
[edit protocols]
user@device# set layer2-control mac-rewrite interface interface-name enable-all-ifl
```

4. (Optional) Configure L2PT for multiple Layer 2 protocols.

   If you want an interface to support tunneling more than one Layer 2 protocol, you must enter the `mac-rewrite` statement separately to select each of the protocols you want to tunnel.

   For example, on an EX9200 switch, the following commands configure a UNI (`xe-1/1/3`) for Q-in-Q tunneling and MAC address rewriting for STP:

```
set interfaces xe-1/1/3 flexible-vlan-tagging
set interfaces xe-1/1/3 encapsulation extended-vlan-bridge
set interfaces xe-1/1/3 unit 10 encapsulation vlan-bridge
set interfaces xe-1/1/3 unit 10 vlan-id 10
set interfaces xe-1/1/3 native-vlan-id 10
set interfaces xe-1/1/3 unit 10 input-vlan-map push
set interfaces xe-1/1/3 unit 10 input-vlan-map vlan-id 100
set interfaces xe-1/1/3 unit 10 output-vlan-map pop
set protocols layer2-control mac-rewrite interface xe-1/1/3 protocol stp
set vlans v10 interface xe-1/1/3.10
```

   On an ELS EX Series switch or a QFX Series switch, the following commands configure a UNI (`ge-0/0/0`) for Q-in-Q tunneling and MAC address rewriting for STP and LLDP:

```
set interfaces ge-0/0/0 flexible-vlan-tagging
set interfaces ge-0/0/0 encapsulation extended-vlan-bridge
set interfaces ge-0/0/0 unit 10 vlan-id 10
```

```
set interfaces ge-0/0/0 native-vlan-id 10
set interfaces ge-0/0/0 unit 10 input-vlan-map push
set interfaces ge-0/0/0 unit 10 output-vlan-map pop
set protocols layer2-control mac-rewrite interface ge-0/0/0 protocol stp
set protocols layer2-control mac-rewrite interface ge-0/0/0 protocol lldp
set vlans v10 interface ge-0/0/0.10
```

5. (Optional) If you want to tunnel protocols to or from two locally-connected UNIs on the *same* switch, configure L2PT.

   In this case, although you still configure the `mac-rewrite` statement to specify the protocol being tunneled, the switch simply floods the protocol packets within the VLAN instead of rewriting the MAC address. You use the same configuration for both interfaces, and you don't need to use a loopback cable.

   For example, the following commands configure two UNIs (`ge-0/0/0` and `ge-0/0/1`) in VLAN v20 for Q-in-Q tunneling on a switch, and the two ports on the switch exchange LACP and LLDP packets:

```
set vlans v20 vlan-id 20
set interfaces ge-0/0/0 unit 20 vlan-id 20
set interfaces ge-0/0/0 unit 20 family ethernet-switching vlan members v20
set interfaces ge-0/0/0 flexible-vlan-tagging
set interfaces ge-0/0/0 native-vlan-id 20
set interfaces ge-0/0/0 encapsulation extended-vlan-bridge
set interfaces ge-0/0/0 unit 20 input-vlan-map push
set interfaces ge-0/0/0 unit 20 output-vlan-map pop
set interfaces ge-0/0/1 unit 20 vlan-id 20
set interfaces ge-0/0/1 unit 20 family ethernet-switching vlan members v20
set interfaces ge-0/0/1 flexible-vlan-tagging
set interfaces ge-0/0/1 native-vlan-id 20
set interfaces ge-0/0/1 encapsulation extended-vlan-bridge
set interfaces ge-0/0/1 unit 20 input-vlan-map push
set interfaces ge-0/0/1 unit 20 output-vlan-map pop
set protocols layer2-control mac-rewrite interface ge-0/0/0 protocol lacp
set protocols layer2-control mac-rewrite interface ge-0/0/0 protocol lldp
set protocols layer2-control mac-rewrite interface ge-0/0/1 protocol lacp
set protocols layer2-control mac-rewrite interface ge-0/0/1 protocol lldp
set vlans v20 interface ge-0/0/0.20
set vlans v20 interface ge-0/0/1.20
```

6. To check the protocols configured for L2PT on an interface, enter the `show mac-rewrite interface` CLI command with the interface name.

For example:

```
user@device> show mac-rewrite interface ge-0/0/0

Interface       Protocols
 ge-0/0/0        LLDP STP
```

If you don't specify an interface name, the `show mac-rewrite interface` command displays all interfaces with L2PT configured.

For example:

```
user@switch> show mac-rewrite interface

Interface       Protocols
 ge-0/0/0        LACP LLDP
 ge-0/0/1        LACP LLDP
```

7. To detect and clear an interface configured with L2PT that appears to be blocked due to a MAC rewrite error, see "Clearing a MAC Rewrite Error on an Interface with Layer 2 Protocol Tunneling" on page 723.

   You have configured L2TP on your device.

8. Repeat the configuration on the device at the other end of the tunnel.

Use the following table to review platform-specific behaviors for your platforms.

Table 103: Platform-Specific Behavior for the L2PT

Platform	Difference
QFX Series	• (QFX5130-32CD, QFX5130E-32CD, QFX5130-48C, QFX5130-48CM, QFX5700, and QFX5700E) If you configure MAC rewrite for any protocol on any interface, that protocol only works on that interface. It does not work on other interfaces. • (QFX5130-32CD, QFX5130E-32CD, QFX5130-48C, QFX5130-48CM, QFX5700, and QFX5700E) You cannot configure L2PT over VXLAN and L2PT MAC rewrite together. You must configure one or the other. • (QFX5130-32CD, QFX5130E-32CD, QFX5130-48C, QFX5130-48CM) You can configure a maximum of 8 unique combinations of MAC rewrite protocols on interfaces on these devices. As long as multiple interfaces share the same combination of protocols, the device counts that as one combination.

SEE ALSO

Using the Enhanced Layer 2 Software CLI | 15

Configuring VLAN Encapsulation | 269

Stacking a VLAN Tag | 359

Clearing a MAC Rewrite Error on an Interface with Layer 2 Protocol Tunneling

On devices with Layer 2 protocol tunneling (L2PT) configured, customer-facing ports should not receive packets with the L2PT MAC address as the destination address unless you have a network topology or configuration error. Under these conditions, when an interface with L2PT enabled receives an L2PT packet, the interface state becomes disabled due to a MAC rewrite error, and you must subsequently re-enable it to continue operation.

1. To check whether an interface with L2PT enabled has become disabled due to a MAC rewrite error condition, use the `show interfaces` operational command:

```
user@switch> show interfaces interface-name
```

   If the interface status includes `Disabled, Physical link is Down` or `Enabled, Physical link is Down` and the `MAC-REWRITE Error` field is `Detected`, then the device detected a MAC rewrite error that contributed to the interface being down. When the device did not detect any MAC rewrite errors, the `MAC-REWRITE Error` field is `None`.

   For example, the following output shows the device detected a MAC rewrite error on the given interface:

```
user@switch> show interfaces ge-0/0/2
Physical interface: ge-0/0/2, Disabled, Physical link is Down
  Interface index: 150, SNMP ifIndex: 531
  Link-level type: Ethernet, MTU: 1514, LAN-PHY mode, Speed: 1000mbps, BPDU Error: None,
  Loop Detect PDU Error: None, Ethernet-Switching Error: None, Source filtering: Disabled
  Ethernet-Switching Error: None, MAC-REWRITE Error: Detected, Loopback: Disabled,
  Flow control: Enabled, Auto-negotiation: Enabled, Remote fault: Online, Media type: Fiber
  Device flags   : Present Running
```

2. On routers, QFX Series switches, and EX Series switches that use the Enhanced Layer 2 Software configuration style, you can clear a MAC rewrite error from the Junos CLI.

   To clear a MAC rewrite error from an interface that has L2PT enabled, use the `clear error mac-rewrite` operational command:

```
user@switch> clear error mac-rewrite interface-name
```

Configure Layer 2 Protocol Tunneling on EX Series Switches Without ELS Support

NOTE: This task applies only to switches that do not support the Enhanced Layer 2 Software (ELS) configuration style. Use Feature Explorer to check if your switch supports ELS.

Tunneled Layer 2 PDUs do not normally arrive at high rate. If the tunneled Layer 2 PDUs do arrive at high rate, there might be a problem in the network. Typically, you would want to shut down the interface that is receiving a high rate of tunneled Layer 2 PDUs to isolate the problem. You can use the `shutdown-threshold` statement to do so. However, if you do not want to completely shut down the interface, you can use the `drop-threshold` statement to configure the switch to drop tunneled Layer 2 PDUs that exceed a certain threshold.

There are no default settings for `drop-threshold` and `shutdown-threshold`, so unless you explicitly configure these values, the switch doesn't enforce any thresholds. As a result, the switch tunnels all Layer 2 PDUs regardless of the speed at which they are received, although the number of packets tunneled per second might be limited by other factors.

You can specify a drop threshold value without specifying a shutdown threshold value, and you can specify a shutdown threshold value without specifying a drop threshold value. If you specify both threshold values, then the drop threshold value must be less than or equal to the shutdown threshold value. If the drop threshold value is greater than the shutdown threshold value and you try to commit the configuration, the commit will fail.

NOTE: You can't configure both L2PT and VLAN translation with the `mapping` statement on the same VLAN. However, you can configure L2PT on one VLAN on a switch and VLAN translation on a different VLAN that doesn't have L2PT configured.

If the switch receives untagged Layer 2 control PDUs to be tunnelled, then you must configure the switch to map untagged (native) packets to an L2PT-enabled VLAN. Otherwise, the switch discards untagged Layer 2 control PDU packets. For more information, see "Understanding Q-in-Q Tunneling and VLAN Translation " on page 941 and "Configuring Q-in-Q Tunneling on EX Series Switches" on page 963.

To configure L2PT on an EX Series switch without ELS support:

1. Because L2PT operates under the Q-in-Q tunneling configuration, you must enable Q-in-Q tunneling before you can configure L2PT.

   Enable Q-in-Q tunneling on VLAN customer-1:

   ```
   [edit]
   user@switch# set vlans customer-1 dot1q-tunneling
   ```

2. Enable L2PT for the Layer 2 protocol you want to tunnel, on the VLAN:

- To enable L2PT for a specific protocol (here, STP):

```
[edit]
user@switch# set vlans customer-1 dot1q-tunneling layer2-protocol-tunneling stp
```

- To enable L2PT for all supported protocols:

```
[edit]
user@switch# set vlans customer-1 dot1q-tunneling layer2-protocol-tunneling all
```

3. (Optional) Configure the drop threshold.

   If you also configure the shutdown threshold, ensure that you configure the drop threshold value to be less than or equal to the shutdown threshold value. If the drop threshold value is greater than the shutdown threshold value and you to try to commit the configuration changes, the commit will fail.

```
[edit]
user@switch# set vlans customer-1 dot1q-tunneling layer2-protocol-tunneling stp drop-
threshold 50
```

4. (Optional) Configure the shutdown threshold:

   If you also configure the drop threshold, ensure that you configure the shutdown threshold value to be greater than or equal to the drop threshold value. If the shutdown threshold value is less than the drop threshold value and you to try to commit the configuration changes, the commit will fail.

```
[edit]
user@switch# set vlans customer-1 dot1q-tunneling layer2-protocol-tunneling stp shutdown-
threshold 100
```

NOTE: After an interface becomes disabled, you must explicitly reenable it using the `clear ethernet-switching layer2-protocol-tunneling error` command. Otherwise, the interface remains disabled.

Example: Configure Layer 2 Protocol Tunneling on EX Series Switches Without ELS Support

IN THIS SECTION

NOTE: This example uses Junos OS for EX Series switches that do not support the Enhanced Layer 2 Software (ELS) configuration style. Use Feature Explorer to check if your EX Series switch supports ELS.

In this example, learn how to configure L2PT to tunnel Layer 2 traffic across two sites in a customer network. These sites are connected across a service provider network.

Requirements

This example uses the following hardware and software components:

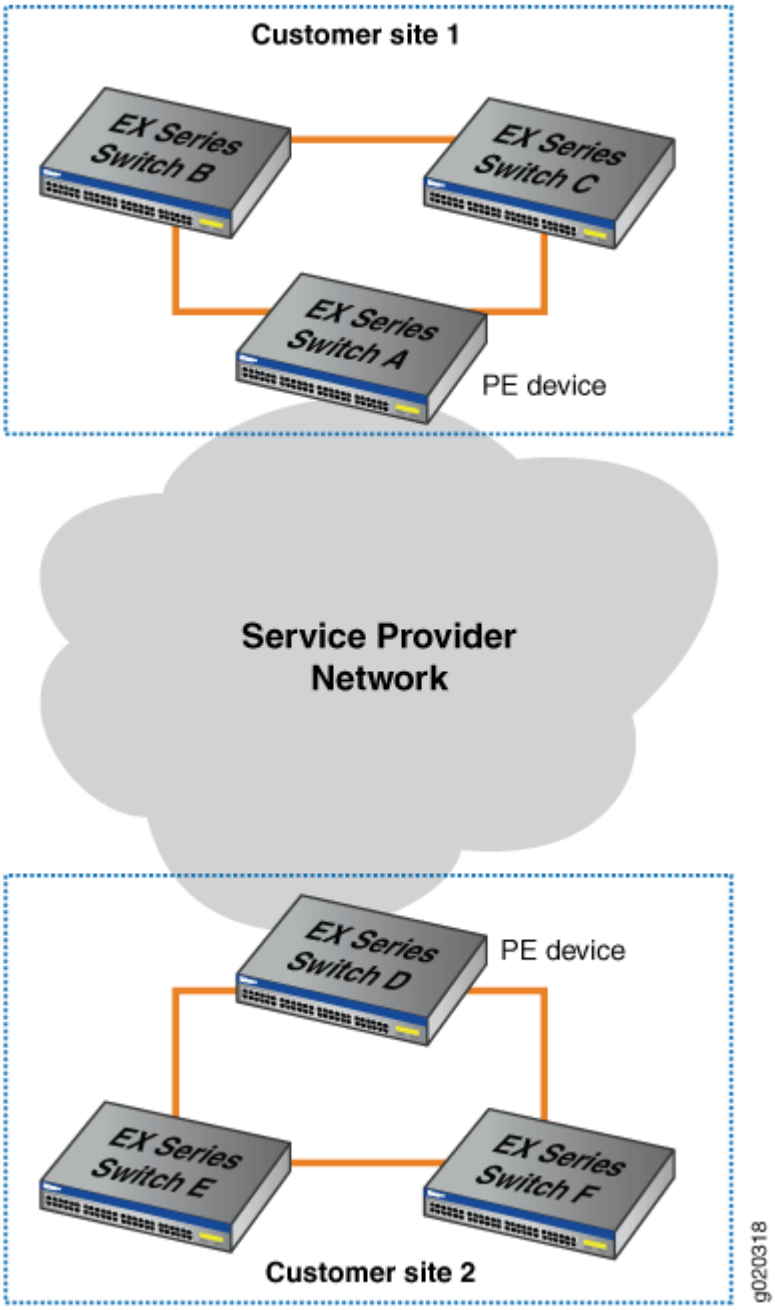
- Six EX Series switches, with three each at two customer sites, with one of the switches at each site designated as the provider edge (PE) device
- Junos OS Release 10.0 or later for EX Series switches

Overview and Topology

L2PT enables you to send Layer 2 PDUs across a service provider network and deliver them to EX Series switches that are not part of the local broadcast domain.

Figure 38 on page 728 shows a customer network that includes two sites that are connected across a service provider network. Site 1 contains three switches connected in a Layer 2 network, with Switch A designated as a provider edge (PE) device in the service provider network. Site 2 contains a Layer 2 network with a similar topology to that of Site 1, with Switch D designated as a PE device.

Figure 38: L2PT Topology



When you enable L2PT on a VLAN, you also must enable Q-in-Q tunneling. Q-in-Q tunneling ensures that Switches A, B, C, D, E, and F are part of the same broadcast domain.

This example uses STP as the Layer 2 protocol being tunneled, but you could substitute any of the supported protocols for STP. You can also use the `all` keyword to enable L2PT for all supported Layer 2 protocols.

Tunneled Layer 2 PDUs do not normally arrive at a high rate. If the tunneled Layer 2 PDUs do arrive at a high rate, you might have a problem in the network. Typically, you would want to shut down the interface that is receiving a high rate of tunneled Layer 2 PDUs so that the problem can be isolated. Alternately, if you do not want to completely shut down the interface, you can configure the switch to drop tunneled Layer 2 PDUs that exceed a certain threshold.

The `drop-theshold` configuration statement enables you to specify the maximum number of Layer 2 PDUs of the specified protocol that can be received per second on the interfaces in a specified VLAN before the switch begins dropping the Layer 2 PDUs. The drop threshold must be less than or equal to the shutdown threshold. If the drop threshold is greater than the shutdown threshold and you try to commit the configuration, the commit will fail.

The `shutdown-threshold` configuration statement enables you to specify the maximum number of Layer 2 PDUs of the specified protocol that can be received per second on the interfaces in a specified VLAN before the specified interface is disabled. The shutdown threshold must be greater than or equal to the drop threshold. You can specify a drop threshold without specifying a shutdown threshold, and you can specify a shutdown threshold without specifying a drop threshold. If you do not specify these thresholds, then no thresholds are enforced. As a result, the switch tunnels all Layer 2 PDUs regardless of the speed at which they are received, although the number of packets tunneled per second might be limited by other factors.

In this example, we will configure both a drop threshold and a shutdown threshold to show how this is done.

If L2PT-encapsulated packets are received on an access interface, the switch reacts as it does when there is a loop between the service provider network and the customer network and shuts down (disables) the access interface.

Once an interface is disabled, you must explicitly reenable it using the `clear ethernet-switching layer2-protocol-tunneling error` command or else the interface will remain disabled.

Configuration

IN THIS SECTION

To configure L2PT, perform these tasks:

Procedure

CLI Quick Configuration

To quickly configure L2PT, copy the following commands and paste them into the switch terminal window of each PE device (in Figure 38 on page 728, Switch A and Switch D are the PE devices):

```
[edit]
set vlans customer-1 dot1q-tunneling
set vlans customer-1 dot1q-tunneling layer2-protocol-tunneling stp
set vlans customer-1 dot1q-tunneling layer2-protocol-tunneling stp drop-threshold 50
set vlans customer-1 dot1q-tunneling layer2-protocol-tunneling stp shutdown-threshold 100
```

Step-by-Step Procedure

To configure L2PT, perform these tasks on each PE device (in Figure 38 on page 728, Switch A and Switch D are the PE devices):

1. Enable Q-in-Q tunneling on VLAN `customer-1`:

```
[edit]
user@switch# set vlans customer-1 dot1q-tunneling
```

2. Enable L2PT for STP on VLAN `customer-1`:

```
[edit]
user@switch# set vlans customer-1 dot1q-tunneling layer2-protocol-tunneling stp
```

3. Configure the drop threshold as 50:

```
[edit]
user@switch# set vlans customer-1 dot1q-tunneling layer2-protocol-tunneling stp drop-
threshold 50
```

4. Configure the shutdown threshold as 100:

```
[edit]
user@switch# set vlans customer-1 dot1q-tunneling layer2-protocol-tunneling stp shutdown-
threshold 100
```

Results

Check the results of the configuration:

```
[edit]
user@switch# show vlans customer-1 dot1q-tunneling
layer2-protocol-tunneling {
    stp {
        drop-threshold 50;
        shutdown-threshold 100;
    }
}
```

Verification

IN THIS SECTION

To verify that L2PT is working correctly, perform this task:

Verify That L2PT Is Working Correctly

Purpose

Verify that Q-in-Q tunneling and L2PT are enabled.

Action

Check to see that Q-in-Q tunneling and L2PT are enabled on each PE device (Switch A and Switch D are the PE devices):

```
user@switchA> show vlans extensive customer-1
VLAN: customer-1, Created at: Thu Jun 25 05:07:38 2009
802.1Q Tag: 100, Internal index: 4, Admin State: Enabled, Origin: Static
Dot1q Tunneling status: Enabled
Layer2 Protocol Tunneling status: Enabled
Protocol: Port Mode, Mac aging time: 300 seconds
Number of interfaces: Tagged 0 (Active = 0), Untagged  3 (Active = 0)
      ge-0/0/7.0, untagged, access
      ge-0/0/8.0, untagged, access
      ge-0/0/9.0, untagged, access
```

Check to see that L2PT is tunneling STP on VLAN `customer-1` and that `drop-threshold` and `shutdown-threshold` have been configured:

```
user@switchA> show ethernet-switching layer2-protocol-tunneling vlan customer-1

Layer2 Protocol Tunneling VLAN information:
VLAN          Protocol      Drop       Shutdown
                            Threshold  Threshold
customer-1    stp           50         100
```

Check the state of the interfaces on which L2PT has been enabled, including what kind of operation (encapsulation or decapsulation) they are performing:

```
user@switchA> show ethernet-switching layer2-protocol-tunneling interface

Layer2 Protocol Tunneling information:
Interface     Operation       State      Description
ge-0/0/0.0    Encapsulation   Shutdown   Shutdown threshold exceeded
ge-0/0/1.0    Decapsulation   Shutdown   Loop detected
ge-0/0/2.0    Decapsulation   Active
```

Meaning

The `show vlans extensive customer-1` command shows that Q-in-Q tunneling and L2PT have been enabled. The `show ethernet-switching layer2-protocol-tunneling vlan customer-1` command shows that L2PT is tunneling STP on VLAN `customer-1`,the drop threshold is set to 50, and the shutdown threshold is set to 100. The `show ethernet-switching layer2-protocol-tunneling interface` command shows the type of operation being performed on each interface, the state of each interface and, if the state is `Shutdown`, the reason why the interface is shut down.

Platform-Specific L2PT Behavior

Use Feature Explorer to confirm platform and release support for specific features.

Use the following table to review platform-specific behavior for your platforms.

See the "Additional Platform Information" on page 733 section for more information.

Platform	Difference
ACX Series (Junos OS Evolved)	• These devices do not support L2PT on access or trunk interfaces.
MX Series	• MX Series routers must have enhanced queuing Dense Port Concentrators (DPCs) to support L2PT.

Additional Platform Information

IN THIS SECTION

Use Feature Explorer to confirm platform and release support for specific features.

Use the following sections to review platform-specific behaviors for your platforms.

ACX Series Routers

L2PT on ACX Series routers supports tunneling the Layer 2 PDUs listed in Table 104 on page 734 with the indicated Ethernet encapsulation type and MAC address.

Table 104: L2PT Protocols Supported on ACX Series Routers

Protocol	Ethernet Encapsulation	MAC Address
802.1X (IEEE 802.1X authentication)	Ether (0x888E)	01:80:C2:00:00:03
802.3ah (IEEE 802.3ah Operation, Administration, and Maintenance (OAM) link fault management (LFM))	Ether (0x8809)	01:80:C2:00:00:02
Cisco Discovery Protocol (CDP)	LLC (0xAAAA03)	01:00:0C:CC:CC:CC
Ethernet local management interface (E-LMI)	Ether (0x88EE)	01:80:C2:00:00:07
Link Aggregation Control Protocol (LACP)	Ether (0x8809)	01:80:C2:00:00:02
Link Layer Discovery Protocol (LLDP)	Ether (0x88CC)	01:80:C2:00:00:0E
Multiple MAC Registration Protocol (MMRP)	Ether (0x88F5)	01:80:C2:00:00:20
MVRP VLAN Registration Protocol (MVRP)	Ether (0x88F6)	01:80:c2:00:00:21
Spanning Tree Protocol (STP), Rapid Spanning Tree Protocol (RSTP), and Multiple Spanning Tree Protocol (MSTP)	LLC (0x424203)	01:80:C2:00:00:00
VLAN Trunking Protocol (VTP)	LLC (0xAAAA03)	01:00:0C:CC:CC:CC

MX Series Routers

MX Series routers that support this feature support tunneling the Layer 2 PDUs shown in Table 105 on page 735.

Table 105: L2PT Protocols Supported on MX Series Routers

Protocol	MAC Address
Cisco Discovery Protocol (CDP)	01:00:0C:CC:CC:CC
Per-VLAN Spanning Tree Protocol (PVSTP)	01:00:0C:CC:CC:CD
Spanning Tree Protocol (STP), Rapid Spanning Tree Protocol (RSTP), and Multiple Spanning Tree Protocol (MSTP)	01:80:C2:00:00:00
VLAN Trunking Protocol (VTP)	01:00:0C:CC:CC:CC

L2PT and MAC rewrite are supported in VPLS, but only certain hardware configurations are supported. Table 106 on page 735 shows the Modular Port Concentrators (MPCs) and enhanced Dense Port Concentrators (DPCs) supported when configuring L2PT and VPLS.

Table 106: MAC Rewrite and VPLS Configurations

CE-Facing Interface	PE-Core Facing Interface	Layer 2 Protocol Tunneling
MPC	MPC	Yes
MPC	Enhanced DPC	Yes
Enhanced DPC	MPC	Yes
Enhanced DPC	Enhanced DPC	No

All MPCs support L2PT. MX Series routers with certain enhanced DPCs or enhanced queuing DPCs support L2PT. See Table 107 on page 736 for a list of the supported DPCs.

NOTE: L2PT is not supported on Rev-A DPCs on MX Series routers because of microcode space limitations.

Table 107: DPCs Supported for L2PT

DPC Name	DPC Model Number
Gigabit Ethernet	
Gigabit Ethernet Enhanced DPC with SFP	DPCE-R-40GE-SFP
Gigabit Ethernet Enhanced Ethernet Services DPC with SFP	DPCE-X-40GE-SFP
Gigabit Ethernet Enhanced Queuing Ethernet Services DPC with SFP	DPCE-X-Q-40GE-SFP
Gigabit Ethernet Enhanced Queuing IP Services DPCs with SFP	DPCE-R-Q-20GE-SFP
Gigabit Ethernet Enhanced Queuing IP Services DPCs with SFP	DPCE-R-Q-40GE-SFP
10-Gigabit Ethernet	
10-Gigabit Ethernet Enhanced DPCs with XFP	DPCE-R-2XGE-XFP
10-Gigabit Ethernet Enhanced DPCs with XFP	DPCE-R-4XGE-XFP
10-Gigabit Ethernet Enhanced Ethernet Services DPC with XFP	DPCE-X-4XGE-XFP
10-Gigabit Ethernet Enhanced Queuing Ethernet Services DPC with XFP	DPCE-X-Q-4XGE-XFP
10-Gigabit Ethernet Enhanced Queuing IP Services DPC with XFP	DPCE-R-Q-4XGE-XFP
Multi-Rate Ethernet	
Multi-Rate Ethernet Enhanced DPC with SFP and XFP	DPCE-R-20GE-2XGE

Table 107: DPCs Supported for L2PT *(Continued)*

DPC Name	DPC Model Number
Multi-Rate Ethernet Enhanced Ethernet Services DPC with SFP and XFP	DPCE-X-20GE-2XGE
Multi-Rate Ethernet Enhanced Queuing IP Services DPC with SFP and XFP	DPCE-R-Q-20GE-2XGE
Tri-Rate Ethernet	
Tri-Rate Enhanced DPC	DPCE-R-40GE-TX
Tri-Rate Enhanced Ethernet Services DPC	DPCE-X-40GE-TX

NOTE: When a device sends a RADIUS access request, the `Chargeable-User-Identity` parameter is an empty field. For more information about configuring RADIUS, see the *Junos Subscriber Access Configuration Guide*.

EX Series and QFX Series Switches

NOTE: QFX Series and EX Series switches that use the Enhanced Layer 2 Software (ELS) configuration style share the same configuration hierarchy to set up L2PT. The configuration hierarchy is different for EX Series switches that do not support ELS. Use Feature Explorer to check if your EX Series switch supports ELS.

For details on the configuration options to enable tunneling the supported protocols on each type of switch, see either of the following configuration statements:

- QFX Series switches and EX Series ELS switches: *protocol* statement in the `[edit protocols layer2-control mac-rewrite interface interface-name]` hierarchy.
- Non-ELS switches: *layer2-protocol-tunneling* statement in the `[edit vlans vlan-name dot1q-tunneling]` hierarchy.

Table 108 on page 738 lists the Layer 2 protocols that can be tunneled on QFX Series and EX Series switches. All switches that support L2PT can tunnel the listed protocols unless otherwise noted in the second column.

Table 108: L2PT Protocols Supported on EX Series and QFX Series Switches

Layer 2 Protocol That Can Be Tunneled	Support Notes and Exceptions
802.1X authentication	Not supported on EX2300 multigigabit model switches.
802.3ah Operation, Administration, and Maintenance (OAM) link fault management (LFM)	If you enable L2PT for untagged OAM LFM packets, do not configure LFM on the corresponding access interface.
Cisco Discovery Protocol (CDP)	You can't configure CDP on EX Series and QFX Series switches. However, L2PT can tunnel CDP PDUs.
Ethernet local management interface (E-LMI)	Not supported on EX2300 multigigabit model switches.
Generic Attribute Registration Protocol (GARP) VLAN Registration Protocol (GVRP)	Supported.
Link Aggregation Control Protocol (LACP)	If you enable L2PT for untagged LACP packets, do not configure Link Aggregation Control Protocol (LACP) on the corresponding access interface.
Link Layer Discovery Protocol (LLDP)	Supported.
Multiple MAC Registration Protocol (MMRP)	Not supported on EX2300 multigigabit model switches.
MVRP VLAN Registration Protocol (MVRP)	Supported.
Per-VLAN Spanning Tree and Per-VLAN Spanning Tree Plus (PVST+) Protocols	Only supported on EX9200 switches. Use this option to enable tunneling VSTP instead of the `vstp` option.
Spanning Tree Protocol (STP), Rapid Spanning Tree Protocol (RSTP), and Multiple Spanning Tree Protocol (MSTP)	Supported.

Table 108: L2PT Protocols Supported on EX Series and QFX Series Switches *(Continued)*

Layer 2 Protocol That Can Be Tunneled	Support Notes and Exceptions
Unidirectional Link Detection (UDLD)	Not supported on EX2300 multigigabit model switches. You can't configure UDLD on EX Series and QFX Series switches. However, L2PT can tunnel UDLD PDUs.
VLAN Spanning Tree Protocol (VSTP)	EX9200 switches support tunneling VSTP packets but do not have a separate option to enable tunneling VSTP. The option that enables tunneling PVST and PVST+ (`pvstp`) also enables tunneling VSTP.
VLAN Trunking Protocol (VTP)	You can't configure VTP on EX Series and QFX Series switches. However, L2PT can tunnel VTP PDUs.

The egress PE switches use the encapsulated MAC address to identify the tunneled Layer 2 control protocol and do the destination MAC address rewrite. Table 109 on page 739 lists the supported protocols and their corresponding encapsulation types and MAC addresses on EX Series and QFX Series switches:

Table 109: Protocol Destination MAC Addresses

Protocol	Ethernet Encapsulation	MAC Address
802.1X	Ether-II	01:80:C2:00:00:03
802.3ah	Ether-II	01:80:C2:00:00:02
CDP	LLC/SNAP	01:00:0C:CC:CC:CC
E-LMI	Ether-II	01:80:C2:00:00:07
GVRP	LLC/SNAP	01:80:C2:00:00:21
LACP	Ether-II	01:80:C2:00:00:02

Table 109: Protocol Destination MAC Addresses *(Continued)*

Protocol	Ethernet Encapsulation	MAC Address
LLDP	Ether-II	01:80:C2:00:00:0E
MMRP	Ether-II	01:80:C2:00:00:20
MVRP	Ether-II	01:80:C2:00:00:21
PVSTP	LLC/SNAP	01:00:0C:CC:CC:CD
STP, RSTP, MSTP	LLC/SNAP	01:80:C2:00:00:00
UDLD	LLC/SNAP	01:00:0C:CC:CC:CC
VSTP	LLC/SNAP	01:00:0C:CC:CC:CD
VTP	LLC/SNAP	01:00:0C:CC:CC:CC

Change History Table

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

Release	Description
Junos OS Evolved Release 22.4R1	Starting in Junos OS Evolved Release 22.4R1, on PTX10001-36MR, PTX10004, PTX10008, and PTX10016 devices, we support bridge protocol data unit (BPDU) transparency on CCC interfaces. All Layer 2 control frames received at a local provider edge (PE) device in a Layer 2 VPN will be tunneled to the remote PE devices, unless you have configured the respective protocol on the local PE device's interface that connects to its CE device. In prior releases, you were required to use the `l2circuit-control-passthrough` configuration statement under the `forwarding-options` hierarchy level to allow tunneling to remote PE. This configuration statement is no-longer needed and the option is removed from configuration hierarchy. We've implemented this feature per "MEF 6.1.1 Layer 2 Control Protocol Handling Amendment."

Layer 2 Control Protocol (L2CP) Transparent Tunneling

SUMMARY

IN THIS SECTION

Layer 2 Control Protocol (L2CP) Transparent Tunneling forwards L2CP packets transparently in hardware unless a specific protocol is configured on the incoming interface. This feature prevents unnecessary discarding of L2CP Bridge Protocol Data Units (BPDUs), improving network performance and the transit of L2 protocol traffic.

Benefits of L2CP Transparent Tunneling

- **Transparent Forwarding** - By default, network devices forward L2CP packets in hardware, which accelerates L2 protocol traffic, reduces delays, streamlines packet flow, and prevents unnecessary discards.
- **Protocol-Specific Handling** - The network devices manages BPDUs according to configured protocols and redirects them to the CPU only when explicitly defined. This approach optimizes CPU utilization.
- **MEF Compliance** - The system provides a Metro Ethernet Forum (MEF) standard forwarding profile. Using a dedicated CLI command enables this profile. This configuration ensures compliance with industry standards and provides flexibility in packet handling.

Configuring MEF Standard Forwarding Profile

The Metro Ethernet Forum (MEF) specifies the rules for processing L2CP Ethernet frame when the packets arrive at the L2CP decision point on the user network interface (UNI). The rules provide the mechanism for transparently passing the L2CP packets.

To configure the MEF standard forwarding profile for L2CP packets, use the command `MEF-forwarding-profile` at the `[edit system packet-forwarding-options]`. This command configures the system to handle L2CP packets in compliance with MEF standards. Configuring this command restarts the management process (mgd).

22

CHAPTER

Configuring Virtual Routing Instances

IN THIS CHAPTER

Virtual Routing Instances

IN THIS SECTION

Understanding Virtual Routing Instances on EX Series Switches

Virtual routing instances allow administrators to divide a Juniper Networks EX Series Ethernet Switch into multiple independent virtual routers, each with its own routing table. Splitting a device into many virtual routing instances isolates traffic traveling across the network without requiring multiple devices to segment the network.

You can use virtual routing instances to isolate customer traffic on your network and to bind customer-specific instances to customer-owned interfaces.

Virtual routing and forwarding (VRF) is often used in conjunction with Layer 3 subinterfaces, allowing traffic on a single physical interface to be differentiated and associated with multiple virtual routers. Each logical Layer 3 subinterface can belong to only one routing instance.

EX Series switches support IPv4 and IPv6 unicast and multicast VRF traffic. See Feature Explorer for details on VRF support by switch per Junos OS release.

SEE ALSO

Understanding Layer 3 Subinterfaces

Configuring Virtual Routing Instances on EX Series Switches

Use virtual routing and forwarding (VRF) to divide an EX Series switch into multiple virtual routing instances. VRF allows you to isolate traffic traversing the network without using multiple devices to segment your network. VRF is supported on all Layer 3 interfaces.

Before you begin, make sure to set up your VLANs. See *Configuring VLANs for EX Series Switches*, Configuring VLANs for EX Series Switches with ELS Support (CLI Procedure), or Configuring VLANs for EX Series Switches (J-Web Procedure).

To configure virtual routing instances:

1. Create a routing instance:

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

NOTE: EX Series switches only support the virtual-router instance type.

2. Bind each routing instance to the corresponding physical interfaces:

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

3. Create the logical interfaces that are bound to the routing instance.
   - To create a logical interface with an IPv4 address:

```
[edit interfaces]user@switch# set interface-name unit logical-unit-number family inet
address ip-address
```

   - To create a logical interface with an IPv6 address:

```
[edit interfaces]user@switch# set interface-name unit logical-unit-number family inet6
address ipv6-address
```

NOTE: Do not create a logical interface using the **family ethernet-switching** option in this step. Binding an interface using the **family ethernet-switching** option to a routing instance can cause the interface to shutdown.

4. Enable VLAN tagging on each physical interface that was bound to the routing instance:

```
[edit interfaces]user@switch# set interface-name vlan-tagging
```

Example: Using Virtual Routing Instances to Route Among VLANs on EX Series Switches

IN THIS SECTION

Virtual routing instances allow each EX Series switch to have multiple routing tables on a device. With virtual routing instances, you can segment your network to isolate traffic without setting up additional devices.

This example describes how to create virtual routing instances:

Requirements

This example uses the following hardware and software components:

- One EX Series switch
- Junos OS Release 9.2 or later for EX Series switches

Before you create the virtual routing instances, make sure you have:

- Configured the necessary VLANs. See *Configuring VLANs for EX Series Switches*, Configuring VLANs for EX Series Switches with ELS Support (CLI Procedure), or Configuring VLANs for EX Series Switches (J-Web Procedure).

Overview and Topology

In a large office, you may need multiple VLANs to properly manage your traffic. This configuration example shows a simple topology wherein a LAN is segmented into two VLANs, each of which is associated with an interface and a virtual routing instance, on the EX Series switch. This example also shows how to use policy statements to import routes from one of the virtual routing instances to the other.

Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly create and configure virtual routing instances, copy the following commands and paste them into the switch terminal window:

```
[edit]
 set interfaces ge-0/0/3 vlan-tagging
 set interfaces ge-0/0/3 unit 0 vlan-id 1030 family inet address 10.1.1.1/24
 set interfaces ge-0/0/3 unit 1 vlan-id 1031 family inet address 10.1.1.1/24
set interfaces ge-0/0/1 unit 0 family inet address 10.11.1.1/24
set interfaces ge-0/0/2 unit 0 family inet address 10.12.1.1/24
set routing-instances r1 instance-type virtual-router
set routing-instances r1 interface ge-0/0/1.0
set routing-instances r1 interface ge-0/0/3.0
set routing-instances r1 routing-options instance-import import-from-r2
set routing-instances r2 instance-type virtual-router
set routing-instances r2 interface ge-0/0/2.0
set routing-instances r2 interface ge-0/0/3.1
set routing-instances r2 routing-options instance-import import-from-r1
set policy-options policy-statement import-from-r1 term 1 from instance r1
set policy-options policy-statement import-from-r1 term 1 then accept
set policy-options policy-statement import-from-r2 term 1 from instance r2
set policy-options policy-statement import-from-r2 term 1 then accept
```

Procedure

Step-by-Step Procedure

To configure virtual routing instances:

1. Create a VLAN-tagged interface:

```
[edit]user@switch# set interfaces ge-0/0/3 vlan-tagging
```

2. Create one or more subinterfaces on the interfaces to be included in each routing instance:

```
[edit]user@switch# set interfaces ge-0/0/3 unit 0 vlan-id 1030 family inet address
10.1.1.1/24
user@switch# set interfaces ge-0/0/3 unit 1 vlan-id 1031 family inet address 10.1.1.1/24
user@switch# set interfaces ge-0/0/1 unit 0 family inet address 10.11.1.1/24
user@switch# set interfaces ge-0/0/2 unit 0 family inet address 10.12.1.1/24
```

3. Create two virtual routing instances:

```
[edit]user@switch# set routing-instances r1 instance-type virtual-router
user@switch# set routing-instances r2 instance-type virtual-router
```

4. Set the interfaces for the virtual routing instances:

```
[edit]user@switch# set routing-instances r1 interface ge-0/0/1.0
user@switch# set routing-instances r1 interface ge-0/0/3.0
user@switch# set routing-instances r2 interface ge-0/0/2.0
user@switch# set routing-instances r2 interface ge-0/0/3.1
```

5. Apply a policy to routes being imported into each of the virtual routing instances:

```
[edit]user@switch# set routing-instances r1 routing-options instance-import import-from-r2
user@switch# set routing-instances r2 routing-options instance-import import-from-r1
```

6. Create a policy that imports routes from routing instances r1 to r2 and another policy that imports routes from routing instances r2 to r1:

```
[edit]user@switch# set policy-options policy-statement import-from-r1 term 1 from instance r1
user@switch# set policy-options policy-statement import-from-r1 term 1 then accept
user@switch# set policy-options policy-statement import-from-r2 term 1 from instance r2
user@switch# set policy-options policy-statement import-from-r2 term 1 then accept
```

Results

Check the results of the configuration:

```
user@switch> show configuration
interfaces {
            ge-0/0/1 {
                unit 0 {
                    family inet {
                        address 10.11.1.1/24;
                    }
                }
            }
            ge-0/0/2 {
                unit 0 {
                    family inet {
                        address 10.12.1.1/24;
                    }
                }
            }
            ge-1/0/3 {
                vlan-tagging;
                unit 0 {
                    vlan-id 1030;
                    family inet {
                        address 10.1.1.1/24;
                    }
                }
                unit 1 {
                    vlan-id 1031;
                    family inet {
                        address 10.1.1.1/24;
                    }
```

```
                }
            }
        }
        policy-options {
            policy-statement import-from-r1 {
                term 1 {
                    from instance r1;
                    then accept;
                }
            }
            policy-statement import-from-r2 {
                term 1 {
                    from instance r2;
                    then accept;
                }
            }
        }
        routing-instances {
            r1 {
                instance-type virtual-router;
                interface ge-0/0/1.0;
                interface ge-0/0/3.0;
                routing-options {
                    instance-import import-from-r2;
                }
            }
            r2 {
                instance-type virtual-router;
                interface ge-0/0/2.0;
                interface ge-0/0/3.1;
                routing-options {
                    instance-import import-from-r1;
                }
            }
        }
    }
}
```

Verification

IN THIS SECTION

- Verifying That the Routing Instances Were Created | **751**

To confirm that the configuration is working properly, perform these tasks:

Verifying That the Routing Instances Were Created

Purpose

Verify that the virtual routing instances were properly created on the switch.

Action

Use the `show route instance` command:

```
user@switch> show route instance
Instance             Type
         Primary RIB                                 Active/holddown/hidden
master               forwarding
         inet.0                                      6/0/0
         iso.0                                       1/0/0
         inet6.0                                     2/0/0

...

r1                   virtual-router
         r1.inet.0                                   7/0/0

r2                   virtual-router
         r2.inet.0                                   7/0/0
```

Meaning

Each routing instance created is displayed, along with its type, information about whether it is active or not, and its primary routing table.

Verifying That Virtual Routing Instances Are Working on EX Series Switches

IN THIS SECTION

Purpose

After creating a virtual routing instance, make sure it is set up properly.

Action

1. Use the `show route instance` command to list all of the routing instances and their properties:

```
user@switch> show route instance
Instance             Type
         Primary RIB                                 Active/holddown/hidden
master               forwarding
         inet.0                                      3/0/0

__juniper_private1__ forwarding
         __juniper_private1__.inet.0                 1/0/3

__juniper_private2__ forwarding

instance1            forwarding

r1                   virtual-router
         r1.inet.0                                   1/0/0

r2                   virtual-router
         r2.inet.0                                   1/0/0
```

2. Use the `show route forwarding-table` command to view the forwarding table information for each routing instance:

```
user@switch> show route forwarding-table
Routing table: r1.inet
Internet:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct   539     2
0.0.0.0/32         perm     0                    dscd   537     1
10.1.1.0/24       ifdn      0                    rslv   579     1 ge-0/0/3.0
10.1.1.0/32       iddn      0 10.1.1.0           recv   577     1 ge-0/0/3.0
10.1.1.1/32       user      0                    rjct   539     2
10.1.1.1/32       intf      0 10.1.1.1           locl   578     2
10.1.1.1/32       iddn      0 10.1.1.1           locl   578     2
10.1.1.255/32     iddn      0 10.1.1.255         bcst   576     1 ge-0/0/3.0
233.252.0.1/32       perm     0 233.252.0.1          mcst   534     1
255.255.255.255/32 perm     0                    bcst   535     1
```

Meaning

The output confirms that the virtual routing instances are created and the links are up and displays the routing table information.

23

CHAPTER

Configuring Layer 3 Logical Interfaces

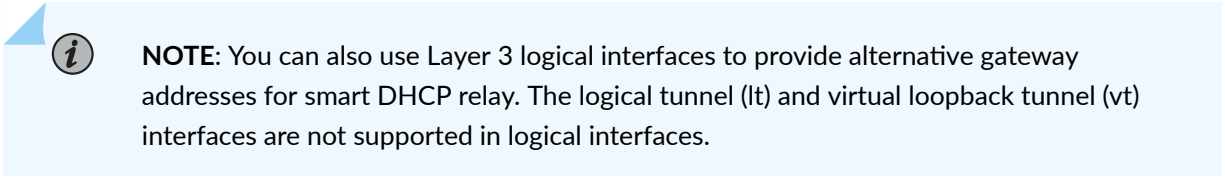
IN THIS CHAPTER

Layer 3 Logical Interfaces

IN THIS SECTION

Understanding Layer 3 Logical Interfaces

A Layer 3 *logical interface* is a logical division of a physical interface that operates at the network level and therefore can receive and forward 802.1Q VLAN tags. You can use Layer 3 logical interfaces to route traffic among multiple VLANs along a single trunk line that connects a Juniper Networks switch to a Layer 2 switch. Only one physical connection is required between the switches. .



NOTE: You can also use Layer 3 logical interfaces to provide alternative gateway addresses for smart DHCP relay. The logical tunnel (lt) and virtual loopback tunnel (vt) interfaces are not supported in logical interfaces.

To create Layer 3 logical interfaces on a switch, enable VLAN tagging, partition the physical interface into logical partitions, and bind the VLAN ID to the logical interface.

We recommend that you use the VLAN ID as the logical interface number when you configure the logical interface. QFX Series and EX4600 switches support a maximum of 4089 VLANs, which includes the default VLAN. You can assign VLAN ID in the range of 1 through 4094.

VLAN tagging places the VLAN ID in the frame header, allowing each physical interface to handle multiple VLANs. When you configure multiple VLANs on an interface, you must also enable tagging on that interface. Junos OS on switches supports a subset of the 802.1Q standard for receiving and forwarding routed or bridged Ethernet frames with single VLAN tags and running Virtual Router Redundancy Protocol (VRRP) over 802.1Q-tagged interfaces.

Configuring a Layer 3 Logical Interface

Devices use Layer 3 logical interfaces to divide a physical interface into multiple logical interfaces, each corresponding to a VLAN. Layer 3 logical interfaces route traffic between subnets.

To configure Layer 3 logical interfaces, enable VLAN tagging and partition one or more physical ports into multiple logical interfaces, each corresponding to a VLAN ID.

Before you begin, make sure you set up your VLANs. See "Configuring VLANs on Switches" on page 152.

To configure Layer 3 logical interfaces:

1. Enable VLAN tagging:

```
[edit interfaces interface-name]
user@switch# set vlan-tagging
```

2. Bind each VLAN ID to a logical interface:

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

Verifying That Layer 3 Logical Interfaces Are Working

IN THIS SECTION

Purpose

After configuring Layer 3 logical interfaces, verify that they are set up properly and transmitting data.

Action

1. To determine if you have successfully created the logical interfaces and the links are up:

```
[edit interfaces]
user@switch> show interfaces interface-name terse
Interface               Admin Link Proto    Local                 Remote
ge-0/0/0                up    up
ge-0/0/0.0              up    up   inet     10.0.1.1/24
ge-0/0/0.1              up    up   inet     10.0.2.2/24
ge-0/0/0.2              up    up   inet     10.0.3.3/24
ge-0/0/0.3              up    up   inet     10.0.4.4/24
ge-0/0/0.4              up    up   inet     10.0.5.5/24
ge-0/0/0.32767          up    up
```

2. Use the `ping` command from a device on one subnet to an address on another subnet to determine if packets were transmitted correctly on the logical interface VLANs:

```
user@switch> ping ip-address
PING 10.1.1.1 (10.1.1.1): 56 data bytes
64 bytes from 10.1.1.1: icmp_seq=0 ttl=64 time=0.157 ms
64 bytes from 10.1.1.1: icmp_seq=1 ttl=64 time=0.238 ms
64 bytes from 10.1.1.1: icmp_seq=2 ttl=64 time=0.255 ms
64 bytes from 10.1.1.1: icmp_seq=3 ttl=64 time=0.128 ms
--- 10.1.1.1 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
```

Meaning

The output confirms that the logical interfaces have been created and the links are up.

24

CHAPTER

Configuring Routed VLAN Interfaces

IN THIS CHAPTER

Routed VLAN Interfaces

IN THIS SECTION

You can configure a routed VLAN interface (RVI) for a private VLAN (PVLAN) on an EX Series Virtual Chassis. Instead of a router connected to a promiscuous port routing Layer 3 traffic between isolated and community members, you can alternatively use an RVI.

To set up routing within a PVLAN, one RVI must be configured for the primary VLAN on one switch or Virtual Chassis in the PVLAN domain. This RVI serves the entire PVLAN domain regardless of whether the domain consists of one or more switches. After you configure the RVI, Layer 3 packets received by the secondary VLAN interfaces are mapped to and routed by the RVI.

When setting up the RVI, you must also enable proxy Address Resolution Protocol (ARP) so that the RVI can handle ARP requests received by the secondary VLAN interfaces.

Configuring a Routed VLAN Interface in a Private VLAN on an EX Series Switch

Before you begin, configure the PVLAN as described in "Creating a Private VLAN on a Single EX Series Switch (CLI Procedure)" on page 485 or "Creating a Private VLAN Spanning Multiple EX Series Switches (CLI Procedure)" on page 491.

To configure an RVI for a PVLAN:

1. Create a logical Layer 3 RVI on a subnet for the primary VLAN's broadcast domain:

```
[edit interfaces]
user@switch# set vlan unit logical-unit-number family inet address inet-address
```

2. Enable unrestricted proxy ARP on the RVI:

```
[edit interfaces]
user@switch# set vlan unit logical-unit-number proxy-arp unrestricted
```

3. Disable sending protocol redirect messages on the RVI:

```
[edit interfaces]
user@switch# set vlan unit logical-unit-number family inet no-redirects
```

4. Link the primary VLAN to the RVI:

```
[edit vlans]
user@switch# set vlan-name l3-interface vlan.logical-unit-number
```

The value of *logical-unit-number* is the same value that you supplied for *logical-unit-number* in the previous steps.

Configuring a Routed VLAN Interface in a Private VLAN on an EX Series Switch (ELS Procedure)

Before you begin, configure the PVLAN as described in "Creating a Private VLAN on a Single Switch with ELS Support (CLI Procedure)" on page 480 or "Creating a Private VLAN Spanning Multiple EX Series Switches with ELS Support (CLI Procedure)" on page 488.

On a switch with ELS, the RVI is called the Intergrated Routing and Bridging (IRB) interface.

To configure an IRB for a PVLAN:

1. Create an IRB on a subnet for the primary VLAN's broadcast domain.

```
[edit interfaces]
user@switch# set irb unit logical-unit-number family inet address inet-address
```

2. Enable unrestricted proxy ARP on the IRB.

```
[edit interfaces]
user@switch# set irb unit logical-unit-number proxy-arp unrestricted
```

3. Disable sending protocol redirect messages on the IRB.

```
[edit interfaces]
user@switch# set irb unit logical-unit-number family inet no-redirects
```

4. Link the primary VLAN to the IRB.

```
[edit vlans]
user@switch# set vlan-name l3-interface irb.logical-unit-number
```

The value of *logical-unit-number* is the same value that you supplied for *logical-unit-number* in the previous steps.

Verifying Routed VLAN Interface Status and Statistics on EX Series Switches

IN THIS SECTION

Purpose

Determine status information and traffic statistics for routed VLAN interfaces (RVIs) by using the following commands:

Action

Display RVI interfaces and their current states:

```
user@switch> show interfaces vlan terse
Interface               Admin Link Proto    Local                 Remote
```

```
vlan                    up    up
vlan.111                up    up   inet     111.111.111.1/24
```

Display Layer 2 VLANs, including any tags assigned to the VLANs and the interfaces associated with the VLANs:

```
user@switch>                        show vlans
Name           Tag     Interfaces
default
                       None
employee-vlan  20
                       ge-1/0/0.0, ge-1/0/1.0, ge-1/0/2.0
marketing      40
                       ge-1/0/10.0, ge-1/0/20.0, ge-1/0/30.0
support        111
                       ge-0/0/18.0
mgmt
                       bme0.32769, bme0.32771*
```

Display Ethernet switching table entries for the VLAN that is attached to the RVI:

```
user@switch>                       show ethernet-switching table
Ethernet-switching table: 1 entries, 0 learned
  VLAN              MAC address       Type         Age Interfaces
  support           00:19:e2:50:95:a0 Static         - Router
```

Display an RVI's ingress-counting statistics with either the **show interfaces vlan detail** command or the **show interfaces vlan extensive** command. Ingress counting is displayed as **Input bytes** and **Input packets** under **Transit Statistics**.

```
user@switch> show interfaces vlan.100 detail

Logical interface vlan.100 (Index 65) (SNMP ifIndex 503) (HW Token 100) (Generation 131)
Flags: SNMP-Traps 0x4000  Encapsulation: ENET2
Traffic statistics:
    Input bytes:       17516756
    Output bytes:      411764
    Input packets:     271745
    Output packets:    8256
Local statistics:
    Input bytes:       3240
```

```
    Output bytes:      411764
    Input packets:     54
    Output packets:    8256
Transit statistics:
    Input bytes:       17513516      0 bps
    Output bytes:      0                0 bps
    Input packets:     271745        0 pps
    Output packets:    0                0 pps
Protocol inet, Generation: 148, Route table: 0
    Flags: None
    Addresses, Flags: iS-Preferred  Is-Primary
        Destination: 50.1.1/24, Local: 50.1.1.1, Broadcast: 50.1.1.255, Generation: 136
```

Meaning

- `show interfaces vlan` displays a list of interfaces, including RVI interfaces, and their current states (up, down).
- `show vlans` displays a list of VLANs, including any tags assigned to the VLANs and the interfaces associated with the VLANs.
- `show ethernet-switching table` displays the Ethernet switching table entries, including VLANs attached to the RVI.
- `show interfaces vlan detail` displays RVI ingress counting as Input Bytes and Input Packets under Transit Statistics.

Change History Table

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

Release	Description
14.1X53-D10	Starting with Junos OS Release 14.1X53-D10, you can configure a routed VLAN interface (RVI) for a private VLAN (PVLAN) on an EX8200 switch or EX8200 Virtual Chassis.

CHAPTER 25

Configuring Integrated Routing and Bridging

IN THIS CHAPTER

Integrated Routing and Bridging

IN THIS SECTION

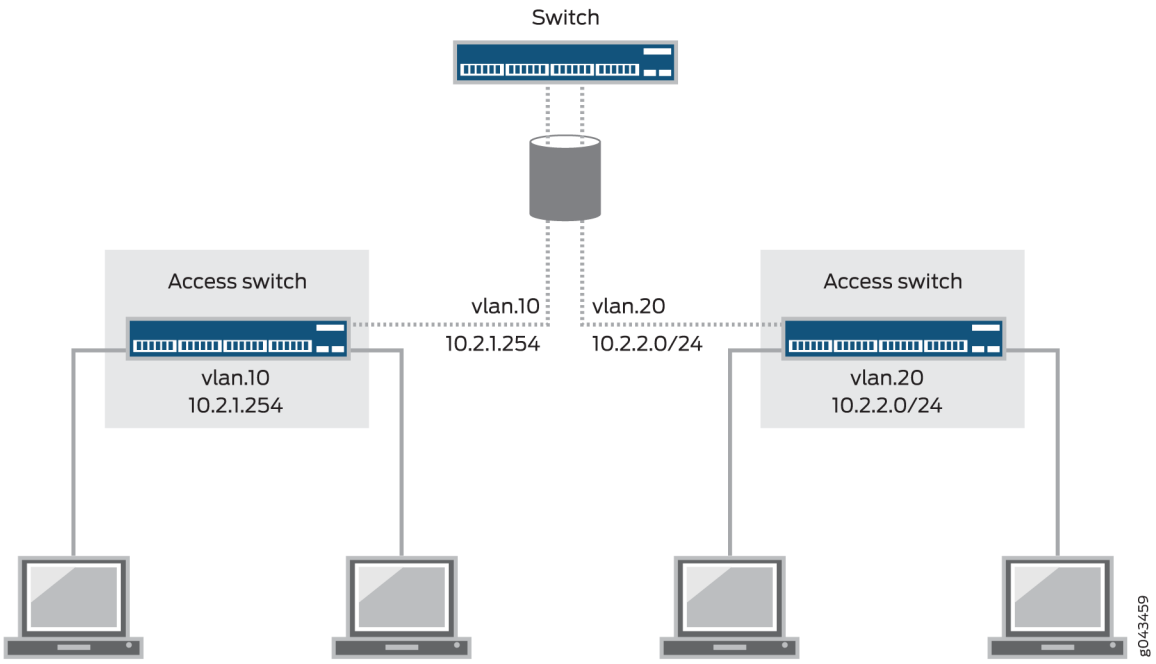
Understanding Integrated Routing and Bridging

IN THIS SECTION

To segment traffic on a LAN into separate broadcast domains, you create separate virtual LANs (VLANs). VLANs limit the amount of traffic flowing across the entire LAN, reducing the possible number of collisions and packet retransmissions within the LAN. For example, you might want to create a VLAN that includes the employees in a department and the resources that they use often, such as printers, servers, and so on.

Figure 39 on page 766 illustrates a switch routing VLAN traffic between two access layer switches using one of these interfaces.

Figure 39: An IRB Interface or RVI on a Switch Providing Routing Between Two Access Switches



Of course, you also want to allow these employees to communicate with people and resources in other VLANs. To forward packets between VLANs, you normally need a router that connects the VLANs. However, you can accomplish this forwarding on a switch without using a router by configuring an integrated routing and bridging (IRB) interface. (These interfaces are also called *routed VLAN interfaces*, or *RVIs*). Using this approach reduces complexity and avoids the costs associated with purchasing, installing, managing, powering, and cooling another device.

An IRB is a special type of Layer 3 virtual interface named `vlan`. Like normal Layer 3 interfaces, the `vlan` interface needs a logical unit number with an IP address. In fact, to be useful an IRB needs at least two logical units and two IP addresses—you must create units with addresses in each of the subnets associated with the VLANs between which you want traffic to be routed. That is, if you have two VLANs (for example, VLAN `red` and VLAN `blue`) with corresponding subnets, your IRB must have a logical unit with an address in the subnet for `red` and a logical unit with an address in the subnet for `blue`. The switch automatically creates direct routes to these subnets and uses these routes to forward traffic between

VLANs. Packets arriving on a Layer 2 interface that are destined for the device's MAC address are classified as Layer 3 traffic while packets that are not destined for the device's MAC address are classified as Layer 2 traffic. Packets destined for the device's MAC address are sent to the IRB interface. Packets from the device's routing engine are sent out the IRB interface.

NOTE: If you specify a VLAN identifier list in the VLAN configuration, you cannot configure an IRB interface for the VLAN.

NOTE: If you are using a version of Junos OS that supports Enhanced Layer 2 Software (ELS), you can also create a Layer 3 virtual interface named `irb` instead of `vlan`—that is, both statements are supported by ELS

IRB interfaces supporting the Enhanced Layer 2 Software (ELS) configuration style and RVIs that support non-ELS switches provide the same functionality. Where the functionality for both features is the same, this topic uses the term *these interfaces* to refer collectively to both IRB interfaces and RVIs. Where differences exist between the two features, this topic calls out the IRB interfaces and RVIs separately.

Table 110 on page 767 shows values you might use when configuring an IRB:

Table 110: Sample IRB Values

Property	Settings
VLAN names and tags (IDs)	`blue`, ID `100` `red`, ID `200`
Subnets associated with VLANs	`blue`: `192.0.2.0/25` (addresses `192.0.2.1` through `192.0.2.126`) `red`: `192.0.2.128/25` (addresses `192.0.2.129` through `192.0.2.254`)
IRB name	interface `irb`
IRB units and addresses	logical unit 100: `192.0.2.1/25` logical unit 200: `192.0.2.129/25`

For the sake of consistency and to avoid confusion, Table 110 on page 767 shows IRB logical unit numbers that match the IDs of the corresponding VLANs. However, you do not have to assign logical

unit numbers that match the VLAN IDs—you can use any values for the units. To bind the logical units of the IRB to the appropriate VLANs, you use the *l3-interface* statement.

Because IRBs operate at Layer 3, you can use Layer 3 services such as firewall filters or CoS rewriting with them.

Table 111 on page 768 shows the number of IRBs/RVIs that each QFX platform supports.

Table 111: Number of Supported IRBs/RVIs by Platform

Platform	Number of Supported IRBs/RVIs
QFX3500	1200
QFX3000-G	1024
QFX3000-M	1024

IRB Interfaces on SRX Series Devices

On SRX1400, SRX1500, SRX3400, SRX3600, SRX4100, SRX4200, SRX4600, SRX5600, and SRX5800 devices, Juniper supports an IRB interface that allows you to terminate management connections in transparent mode. However, you cannot route traffic on that interface or terminate IPsec VPNs.

On SRX300, SRX320, SRX340, SRX345, and SRX550M devices, the following features are not supported on *IRB interface*:

- IS-IS (family ISO)
- Encapsulations (Ether CCC, VLAN CCC, VPLS, PPPoE, and so on) on VLAN interfaces
- CLNS
- DVMRP
- VLAN interface MAC change
- G-ARP
- Change VLAN-Id for VLAN interface

You can configure only one IRB logical interface for each VLAN.

Starting with Junos OS Release 15.1X49-D60 and Junos OS Release 17.3R1, interface statistics are supported on the IRB logical interface for SRX300, SRX320, SRX340, SRX345, and SRX550M devices.

To verify the IRB logical interface statistics, enter the `show interfaces irb.<index> extensive` and `show interfaces irb.<index>statistics` commands.

When Should I Use an IRB Interface or RVI?

Configure an IRB interface or an RVI for a VLAN if you need to:

- Allow traffic to be routed between VLANs.
- Provide Layer 3 IP connectivity to the switch.
- Monitor individual VLANs for billing purposes. Service providers often need to monitor traffic for this purpose, but this capability can be useful for enterprises where various groups share the cost of the network.

How Does an IRB Interface or RVI Work?

For an IRB interface, the switch provides the name irb, and for an RVI, the switch provides the name vlan. Like all Layer 3 interfaces, these interfaces require a logical unit number with an IP address assigned to it. In fact, to be useful, the implementation of these interfaces in an enterprise with multiple VLANs requires at least two logical units and two IP addresses—you must create units with addresses in each of the subnets associated with the VLANs between which you want traffic to be routed. That is, if you have two VLANs (for example, VLAN **red** and VLAN **blue**) with corresponding subnets, your interfaces must have a logical unit with an address in the subnet for **red** and a logical unit with an address in the subnet for **blue**. The switch automatically creates direct routes to these subnets and uses these routes to forward traffic between VLANs.

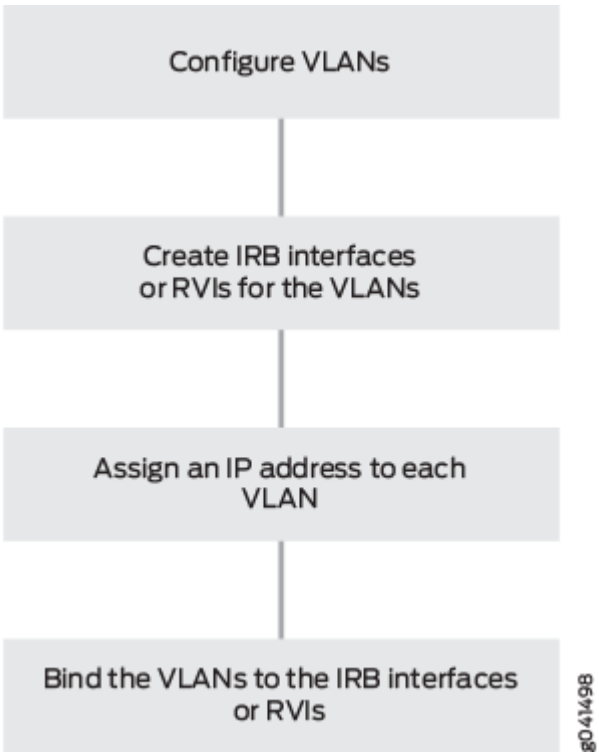
The interface on the switch detects both MAC addresses and IP addresses, then routes data to other Layer 3 interfaces on routers or other switches. These interfaces detect both IPv4 and IPv6 unicast and multicast virtual routing and forwarding (VRF) traffic. Each *logical interface* can belong to only one routing instance and is further subdivided into logical interfaces, each with a logical interface number appended as a suffix to the names irb and vlan—for example, irb.10 and vlan.10.

Creating an IRB Interface or RVI

You create an IRB *logical interface* in a similar manner as a Layer 3 interface, but the IRB interface does not support traffic forwarding or routing. The IRB interface cannot be assigned to a security zone; however, you can configure certain services on a per-zone basis to allow host-inbound traffic for management of the device. This allows you to control the type of traffic that can reach the device from interfaces bound to a specific zone.

There are four basic steps in creating an IRB interface or RVI as shown in Figure 40 on page 770.

Figure 40: Creating an IRB Interface or RVI



The following explanations correspond to the four steps for creating a VLAN, as depicted in Figure 40 on page 770.

- Configure VLANs—Virtual LANs are groups of hosts that communicate as if they were attached to the same broadcast stream. VLANs are created with software and do not require a physical router to forward traffic. VLANs are Layer 2 constructs.
- Create IRB interfaces or RVIs for the VLANs—The switch's IRB interfaces and RVIs use Layer 3 logical interfaces (unlike routers, which can use either physical or logical interfaces).
- Assign an IP address to each VLAN—An IRB interface or RVI cannot be activated unless it is associated with a physical interface.
- Bind the VLANs to the logical interfaces—There is a one-to-one mapping between a VLAN and an IRB interface or RVI, which means that only one of these interfaces can be mapped to a VLAN.

For specific instructions for creating an IRB interface, see "Configuring Integrated Routing and Bridging Interfaces on Switches (CLI Procedure)" on page 775, and for an RVI, see Configuring Routed VLAN Interfaces on Switches (CLI Procedure).

Viewing IRB Interface and RVI Statistics

Some switches automatically track IRB interface and RVI traffic statistics. Other switches allow you to configure tracking. Table 112 on page 771 illustrates the IRB interface- and RVI-tracking capability on various switches.

Table 112: Tracking IRB Interface and RVI Usage

Switch	Input (ingress)	Output (Egress)
EX4300	Automatic	Automatic
EX3200, EX4200	Automatic	–
EX8200	Configurable	Automatic
EX2200, EX3300, EX4500, EX6200	–	–

You can view input (ingress) and output (egress) totals with the following commands:

- For IRB interfaces, use the `show interfaces irb extensive` command. Look at the input and output values in the Transit Statistics field for IRB interface activity values.
- For RVI, use the `show interfaces vlan extensive` command. Look at the input and output values in the Logical Interface Transit Statistics field for RVI activity values.

IRB Interfaces and RVI Functions and Other Technologies

IRB interfaces and RVIs are similar to switch virtual interfaces (SVIs) and bridge-group virtual interfaces (BVIs), which are supported on other vendors' devices. They can also be combined with other functions:

- VRF is often used in conjunction with Layer 3 subinterfaces, allowing traffic on a single physical interface to be differentiated and associated with multiple virtual routers. For more information about VRF, see "Understanding Virtual Routing Instances on EX Series Switches " on page 744.
- For redundancy, you can combine an IRB interface or RVI with implementations of the Virtual Router Redundancy Protocol (VRRP) in both bridging and virtual private LAN service (VPLS) environments. For more information about VRRP, see *Understanding VRRP*.

SEE ALSO

Ethernet Switching and Layer 2 Transparent Mode Overview | 5

Example: Configuring VLANs on Security Devices | 153

Configuring IRB Interfaces on Switches

Integrated routing and bridging (IRB) interfaces enable a switch to recognize which packets are being sent to local addresses so that they are bridged whenever possible and are routed only when needed. Whenever packets can be switched instead of routed, several layers of processing are eliminated. Switching also reduces the number of address look-ups.

NOTE: In versions of Junos OS that do not support Enhanced Layer 2 Software (ELS), this type of interface is called a routed VLAN interface (RVI).

NOTE: When you upgrade from Junos OS Release 15.1X53 to Junos OS Release 17.3R1, you must define an IRB interface at both the `[edit vlans l3-interface]` and `[edit interfaces irb]` hierarchies, otherwise there will be a commit error.

To configure the routed VLAN interface:

1. Create the VLAN by assigning it a name and a VLAN ID:

```
[edit]
user@switch# set vlans support vlan-id 111
```

2. Assign an interface to the VLAN by specifying the logical interface (with the `unit` statement) and specifying the VLAN name as the member:

```
[edit]
user@switch# set interfaces ge-0/0/18 unit 0 family ethernet-switching vlan members
support
```

3. Create the subnet for the VLAN's broadcast domain:

```
[edit]
user@switch# set interfaces irb unit 111 family inet address 10.0.0.X/8
```

Where the value of X can be any number between the range 1 to 254.

4. Bind a Layer 3 interface with the VLAN:

```
[edit]
user@switch# set vlans support l3-interface irb.111
```

NOTE: If you are using a version of Junos OS that does not support ELS, you create a Layer 3 virtual interface named `vlan`

NOTE: Layer 3 interfaces on trunk ports allow the interface to transfer traffic between multiple VLANs. Within a VLAN, traffic is bridged, while across VLANs, traffic is routed.

You can display the configuration settings:

```
user@switch> show interfaces irb terse
Interface               Admin Link Proto    Local                 Remote
vlan                    up    up
irb.111                 up    up   inet     10.0.0.0/8
```

```
user@switch>                        show vlans
Name           Tag     Interfaces
default
                       None
employee-vlan  20
                       ge-1/0/0.0, ge-1/0/1.0, ge-1/0/2.0
marketing      40
                       ge-1/0/10.0, ge-1/0/20.0, ge-1/0/30.0
support        111
                       ge-0/0/18.0
mgmt
                       bme0.32769, bme0.32771*
```

```
user@switch>                        show ethernet-switching table
Ethernet-switching table: 1 entries, 0 learned
```

```
VLAN             MAC address       Type         Age Interfaces
support          00:19:e2:50:95:a0 Static         - Router
```

Configuring Integrated Routing and Bridging for VLANs

Integrated routing and bridging (IRB) provides simultaneous support for Layer 2 bridging and Layer 3 routing on the same interface. IRB enables you to route packets to another routed interface or to another VLAN that has an IRB interface configured. You configure a logical routing interface by specifying `irb` as an interface name at the `[edit interfaces]` hierarchy level and including that interface in the VLAN.

NOTE: You can include only one Layer 3 interface in a VLAN.

To configure a VLAN with IRB support, include the following statements:

```
[edit]
vlans {
    vlan-name {
        domain-type bridge;
        interface interface-name;
         l3-interface (VLAN) interface-name;
        vlan-id (none | number);
        vlan-tags outer number inner number;
    }
}
```

For each VLAN that you configure, specify a ***vlan-name***. You must also specify the value **bridge** for the `domain-type` statement.

For the `vlan-id` statement, you can specify either a valid VLAN identifier or the **none** option.

NOTE: If you configure a Layer 3 interface to support IRB in a VLAN, you cannot use the **all** option for the `vlan-id` statement.

The `vlan-tags` statement enables you to specify a pair of VLAN identifiers; an **outer** tag and an **inner** tag.

NOTE: For a single VLAN, you can include either the `vlan-id` statement or the `vlan-tags` statement, but not both.

To include one or more logical interfaces in the VLAN, specify the ***interface-name*** for each Ethernet interface to include that you configured at the `[edit interfaces]` hierarchy level.

NOTE: A maximum of 4096 active logical interfaces are supported for a VLAN or on each mesh group in a VPLS routing instance configured for Layer 2 bridging.

To associate a Layer 3 interface with a VLAN, include the `l3-interface` *`interface-name`* statement and specify an ***interface-name*** you configured at the `[edit interfaces irb]` hierarchy level. You can configure only one Layer 3 interface for each VLAN.

IRB interfaces are supported for multicast snooping.

In multihomed VPLS configurations, you can configure VPLS to keep a VPLS connection up if only an IRB interface is available by configuring the **irb** option for the `connectivity-type` statement at the `[edit routing-instances` *`routing-instance-name`* `protocols vpls]` hierarchy level. The `connectivity-type` statement has the **ce** and **irb** options. The **ce** option is the default and specifies that a CE interface is required to maintain the VPLS connection. By default, if only an IRB interface is available, the VPLS connection is brought down.

NOTE: When you configure IRB interfaces in more than one logical system on a device, all of the IRB logical interfaces share the same MAC address.

Configuring Integrated Routing and Bridging Interfaces on Switches (CLI Procedure)

Integrated routing and bridging (IRB) interfaces allow a switch to recognize packets that are being sent to local addresses so that they are bridged (switched) whenever possible and are routed only when necessary. Whenever packets can be switched instead of routed, several layers of processing are eliminated.

An interface named irb functions as a logical router on which you can configure a Layer 3 logical interface for each virtual LAN (VLAN). For redundancy, you can combine an IRB interface with implementations of the Virtual Router Redundancy Protocol (VRRP) in both bridging and virtual private LAN service (VPLS) environments.

Jumbo frames of up to 9216 bytes are supported on an IRB interface. To route jumbo data packets on the IRB interface, you must configure the jumbo MTU size on the member physical interfaces of the VLAN that you have associated with the IRB interface, as well as on the IRB interface itself (the interface named irb).

CAUTION: Setting or deleting the jumbo MTU size on the IRB interface (the interface named irb) while the switch is transmitting packets might result in dropped packets.

To configure the IRB interface:

1. Create a Layer 2 VLAN by assigning it a name and a VLAN ID:

```
[edit]
user@switch# set vlans vlan-name vlan-id vlan-id
```

2. Assign an interface to the VLAN by naming the VLAN as a trunk member on the logical interface, thereby making the interface part of the VLAN's broadcast domain:

```
[edit]
user@switch# set interfaces interface-name unit logical-unit-number family ethernet-switching
vlan members vlan-name
```

3. Create a logical Layer 3 IRB interface (its name will be irb.*logical-interface-number*, where the value for *logical-interface-number* is the value you supplied for *vlan-id* in Step 1; in the following command, it is the *logical-unit-number*) on a subnet for the VLAN's broadcast domain:

```
[edit]
user@switch# set interfaces irb unit logical-unit-number family inet address inet-address
```

4. Link the Layer 2 VLAN to the logical Layer 3 IRB interface:

```
[edit]
user@switch# set vlans vlan-name l3-interface  irb.logical-interface-number
```

NOTE: Layer 3 interfaces on trunk ports allow the interface to transfer traffic between multiple Layer 2 VLANs. Within a VLAN, traffic is switched, while across VLANs, traffic is routed.

Using an IRB Interface in a Private VLAN on a Switch

IN THIS SECTION

VLANs limit broadcasts to specified users. Private VLANs (PVLANs) take this concept a step further by splitting the broadcast domain into multiple isolated broadcast subdomains and essentially putting secondary VLANs inside a primary VLAN. PVLANs restrict traffic flows through their member switch ports (called “private ports”) so that these ports communicate only with a specified uplink trunk port or with specified ports within the same VLAN. PVLANs are useful for restricting the flow of broadcast and unknown unicast traffic and for limiting the communication between known hosts. Service providers use PVLANs to keep their customers isolated from one another.

Just like regular VLANs, PVLANs are isolated at Layer 2 and normally require that a Layer 3 device be used if you want to route traffic. Starting with Junos OS 14.1X53-D30, you can use an integrated routing and bridging (IRB) interface to route Layer 3 traffic between devices connected to a PVLAN. Using an IRB interface in this way can also allow the devices in the PVLAN to communicate at Layer 3 with devices outside the PVLAN.

Configuring an IRB Interface in a Private VLAN

Use the following guidelines when configuring an IRB interface in a PVLAN:

- You can create only one IRB interface in a PVLAN, regardless of how many switches participate in the PVLAN.
- The IRB interface must be a member of the primary VLAN in the PVLAN.
- Each host device that you want to connect at Layer 3 must use the IP address of the IRB as its default gateway address.
- • Because the host devices are isolated at Layer 2, you must configure the following statement for the IRB interface to allow ARP resolution to occur:

  ```
  set interfaces irb unit unit-number proxy-arp unrestricted
  ```

IRB Interface Limitation in a PVLAN

If your PVLAN includes multiple switches, an issue can occur if the Ethernet switching table is cleared on a switch that does not have an IRB interface. If a Layer 3 packet transits the switch before its

destination MAC address is learned again, it is broadcast to all the Layer 3 hosts connected to the PVLAN.

RELATED DOCUMENTATION

Understanding Private VLANs | 428

Creating a Private VLAN on a Single QFX Switch without ELS Support | 483

Example: Configuring Routing Between VLANs on One Switch Using an IRB Interface

IN THIS SECTION

To segment traffic on a LAN into separate broadcast domains, you create separate virtual LANs (VLANs). For example, you might want to create a VLAN that includes the employees in a department and the resources that they use often, such as printers, servers, and so on.

Of course, you also want to allow these employees to communicate with people and resources in other VLANs. To forward packets between VLANs you normally you need a router that connects the VLANs. However, you can accomplish this on a Juniper Networks switch without using a router by configuring an integrated routing and bridging (IRB) interface (also known as a routed VLAN interface—or RVI—in versions of Junos OS that do not support Enhanced Layer 2 Software). Using this approach reduces complexity and avoids the costs associated with purchasing, installing, managing, powering, and cooling another device.

Requirements

This example uses the following hardware and software components:

- One switch

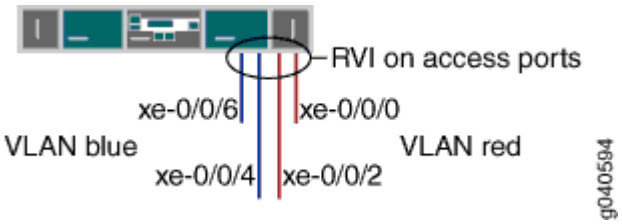
- Junos OS Release 11.1 or later

Overview and Topology

IN THIS SECTION

This example uses an IRB to route traffic between two VLANs on the same switch. The topology is shown in Figure 41 on page 779.

Figure 41: IRB with One Switch



This example shows a simple configuration to illustrate the basic steps for creating two VLANs on a single switch and configuring an IRB to enable routing between the VLANs. One VLAN, called `blue`, is for the sales and marketing group, and a second, called `red`, is for the customer support team. The sales and support groups each have their own file servers and wireless access points. Each VLAN must have a unique name, tag (VLAN ID), and distinct IP subnet. Table 113 on page 779 lists the components of the sample topology.

Topology

Table 113: Components of the Multiple VLAN Topology

Property	Settings
VLAN names and tag IDs	`blue`, ID `100` `red`, ID `200`

Table 113: Components of the Multiple VLAN Topology *(Continued)*

Property	Settings
Subnets associated with VLANs	blue: 192.0.2.0/25 (addresses 192.0.2.1 through 192.0.2.126) red: 192.0.2.128/25 (addresses 192.0.2.129 through 192.0.2.254)
Interfaces in VLAN blue	Sales server port: xe-0/0/4 Sales wireless access points: xe-0/0/6
Interfaces in VLAN red	Support server port: xe-0/0/0 Support wireless access points: xe-0/0/2
IRB name	interface irb
IRB units and addresses	logical unit 100: 192.0.2.1/25 logical unit 200: 192.0.2.129/25

This configuration example creates two IP subnets, one for the blue VLAN and the second for the red VLAN. The switch bridges traffic within the VLANs. For traffic passing between two VLANs, the switch routes the traffic using an IRB on which you have configured addresses in each IP subnet.

To keep the example simple, the configuration steps show only a few interfaces and VLANs. Use the same configuration procedure to add more interfaces and VLANs. By default, all interfaces are in access mode, so you do not have to configure the port mode.

Configure Layer 2 switching for two VLANs

IN THIS SECTION

Procedure

CLI Quick Configuration

To quickly configure Layer 2 switching for the two VLANs (`blue` and `red`) and to quickly configure Layer 3 routing of traffic between the two VLANs, copy the following commands and paste them into the switch terminal window:

NOTE: The following example uses a version of Junos OS that supports Enhanced Layer 2 Software (ELS). When you use ELS, you create a Layer 3 virtual interface named *irb*. If you are using a version of Junos OS that does not support ELS, you create a Layer 3 virtual interface named `vlan`.

```
[edit]
set interfaces xe-0/0/4  unit 0 description “Sales server port”
set interfaces xe-0/0/4 unit 0 family ethernet-switching vlan members blue
set interfaces xe-0/0/6 unit 0 description “Sales wireless access point port”
set interfaces xe-0/0/6 unit 0 family ethernet-switching vlan members blue
set interfaces xe-0/0/0  unit 0 description “Support servers”
set interfaces xe-0/0/0 unit 0 family ethernet-switching vlan members red
set interfaces xe-0/0/2 unit 0 description “Support wireless access point port”
set interfaces xe-0/0/2 unit 0 family ethernet-switching vlan members red
set interfaces irb unit 100 family inet address 192.0.2.1/25
set interfaces irb unit 200 family inet address 192.0.2.129/25
 set vlans blue l3-interface irb.100
set vlans blue vlan-id 100
 set vlans red vlan-id 200
set vlans red l3-interface irb.200
```

Step-by-Step Procedure

To configure the switch interfaces and the VLANs to which they belong:

1. Configure the interface for the sales server in the blue VLAN:

```
[edit interfaces xe-0/0/4 unit 0]
user@switch# set description “Sales server port”
user@switch# set family ethernet-switching vlan members blue
```

2. Configure the interface for the wireless access point in the blue VLAN:

```
[edit interfaces xe-0/0/6 unit 0]
user@switch# set description “Sales wireless access point port”
user@switch# set family ethernet-switching vlan members blue
```

3. Configure the interface for the support server in the red VLAN:

```
[edit interfaces xe-0/0/0 unit 0]
user@switch# set description “Support server port”
user@switch# set family ethernet-switching vlan members red
```

4. Configure the interface for the wireless access point in the red VLAN:

```
[edit interfaces xe-0/0/2 unit 0]
user@switch# set description “Support wireless access point port”
user@switch# set family ethernet-switching vlan members red
```

Step-by-Step Procedure

Now create the VLANs and the IRB. The IRB will have logical units in the broadcast domains of both VLANs.

1. Create the red and blue VLANs by configuring the VLAN IDs for them:

```
[edit vlans]
user@switch# set blue vlan-id 100
user@switch# set red vlan-id 200
```

2. Create the interface named `irb` with a logical unit in the sales broadcast domain (blue VLAN):

```
[edit interfaces]
user@switch# set irb unit 100 family inet address 192.0.2.1/25
```

   The unit number is arbitrary and does not have to match the VLAN tag ID. However, configuring the unit number to match the VLAN ID can help avoid confusion.

3. Add a logical unit in the support broadcast domain (red VLAN) to the `irb` interface:

```
[edit interfaces]
user@switch# set irb unit 200 family inet address 192.0.2.129/25
```

4. Complete the IRB configuration by binding the red and blue VLANs (Layer 2) with the appropriate logical units of the `irb` interface (Layer 3):

```
[edit vlans]
user@switch# set blue l3-interface irb.100
user@switch# set red l3-interface irb.200
```

Configuration Results

Display the results of the configuration:

```
user@switch> show configuration
interfaces {
    xe-0/0/4 {
        unit 0 {
            description “Sales server port”;
            family ethernet-switching {
                vlan members blue;
            }
        }
    }
    xe-0/0/6 {
        unit 0 {
            description “Sales wireless access point port”;
            family ethernet-switching {
                vlan members blue;
            }
        }
    }
    xe-0/0/0 {
        unit 0 {
            description “Support server port”;
            family ethernet-switching {
                vlan members red;
```

```
            }
        }
    }
    xe-0/0/2 {
        unit 0 {
            description “Support wireless access point port”;
            family ethernet-switching {
                vlan members red;
            }
        }
    }
    irb {
        unit 100 {
            family inet address 192.0.2.1/25;
        }
        unit 200 {
            family inet address 192.0.2.129/25;
            }
        }
    }
}
vlans {
    blue {
        vlan-id 100;
        interface xe-0/0/4.0:
        interface xe-0/0/6.0;
        l3-interface irb 100;
    }
    red {
        vlan-id 200;
        interface xe-0/0/0.0:
        interface xe-0/0/2.0;
        l3-interface irb 200;
    }
}
```

TIP: To quickly configure the blue and red VLAN interfaces, issue the `load merge terminal` command, copy the hierarchy, and paste it into the switch terminal window.

Verification

IN THIS SECTION

To verify that the blue and redVLANs have been created and are operating properly, perform these tasks:

Verifying That the VLANs Have Been Created and Associated with the Correct Interfaces

Purpose

Verify that the VLANs blue and red have been created on the switch and that all connected interfaces on the switch are members of the correct VLAN.

Action

List all VLANs configured on the switch:

```
user@switch> show vlans
Name          Tag      Interfaces
default                  xe-0/0/0.0, xe-0/0/2.0, xe-0/0/4.0, xe-0/0/6.0,
blue          100      xe-0/0/4.0, xe-0/0/6,
red           200      xe-0/0/0.0, xe-0/0/2.0, *
mgmt                   me0.0*
```

Meaning

The show vlans command lists all VLANs configured on the switch and which interfaces are members of each VLAN. This command output shows that the blue and red VLANs have been created. The blue VLAN has a tag ID of 100 and is associated with interfaces xe-0/0/4.0 and xe-0/0/6.0. VLAN red has a tag ID of 200 and is associated with interfaces xe-0/0/0.0 and xe-0/0/2.0.

Verifying That Traffic Can Be Routed Between the Two VLANs

Purpose

Verify routing between the two VLANs.

Action

Verify that the IRB logical units are up:

```
user@switch> show interfaces terse
irb.100                 up    up   inet     192.0.2.1/25
irb.200                 up    up   inet     192.0.2.129/25
```

NOTE: At least one port (access or trunk) with an appropriate VLAN assigned to it must be up for the `irb` interface to be up.

Verify that switch has created routes that use the IRB logical units:

```
user@switch> show route
192.0.2.0/25        *[Direct/0] 1d 03:26:45
                     > via irb.100
192.0.2.1/32        *[Local/0] 1d 03:26:45
                       Local via irb.100
192.0.2.128/25      *[Direct/0] 1d 03:26:45
                     > via irb.200
192.0.2.129/32      *[Local/0] 1d 03:26:45
                       Local via irb.200
```

List the Layer 3 routes in the switch's Address Resolution Protocol (ARP) table:

```
user@switch> show arp
MAC Address              Address           Name                      Flags

00:00:0c:06:2c:0d        192.0.2.7         irb.100                   None
00:13:e2:50:62:e0        192.0.2.132       irb.200                   None
```

Meaning

The output of the **show interfaces** and **show route** commands show that the Layer 3 IRB logical units are working and the switch has used them to create direct routes that it will use to forward traffic between the VLAN subnets. The **show arp** command displays the mappings between the IP addresses and MAC addresses for devices on both `irb.100` (associated with VLAN `blue`) and `irb.200` (associated with VLAN `red`).These two devices can communicate.

Example: Configuring an IRB Interface on a Security Device

IN THIS SECTION

This example shows how to configure an IRB interface so it can act as a Layer 3 routing interface for a VLAN.

Requirements

Before you begin, configure a VLAN with a single VLAN identifier. See "Example: Configuring VLANs on Security Devices" on page 153.

Overview

In this example, you configure the IRB logical interface unit 0 with the family type inet and IP address 10.1.1.1/24, and then reference the IRB interface irb.10 in the vlan10 configuration. Then you enable Web authentication on the IRB interface and activate the webserver on the device.

NOTE: To complete the Web authentication configuration, you must perform the following tasks:

- Define the access profile and password for a Web authentication client.

- Define the security policy that enables Web authentication for the client.

Either a local database or an external authentication server can be used as the Web authentication server.

Configuration

IN THIS SECTION

CLI Quick Configuration

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

```
set interfaces ge-1/0/0 unit 0 family ethernet-switching interface-mode trunk
set interfaces ge-1/0/0 unit 0 family ethernet-switching vlan members 10
set interface irb unit 10 family inet address 10.1.1.1/24 web-authentication http
set vlans vlan10 vlan-id 10
set vlans vlan10 l3-interface irb.10
set system services web-management http
```

Procedure

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure an IRB interface:

1. Create a Layer 2 trunk interface.

```
[edit]
user@host# set interfaces ge-1/0/0 unit 0 family ethernet-switching interface-mode trunk
user@host# set interfaces ge-1/0/0 unit 0 family ethernet-switching vlan members 10
```

2. Create an IRB logical interface.

```
[edit]
user@host# set interface irb unit 10 family inet address 10.1.1.1/24 web-authentication http
```

3. Create a Layer 2 VLAN.

```
[edit]
user@host# set vlans vlan10 vlan-id 10
```

4. Associate the IRB interface with the VLAN.

```
[edit]
user@host# set vlans vlan10 l3-interface irb.10
```

5. Activate the webserver.

```
[edit]
user@host# set system services web-management http
```

6. If you are done configuring the device, commit the configuration.

```
[edit]
user@host# commit
```

Verification

To verify the configuration is working properly, enter the `show interface irb` , and `show vlans` commands.

SEE ALSO

Example: Configuring Layer 2 Security Zones | 1051

Example: Configuring VLAN with Members Across Two Nodes on a Security Device

IN THIS SECTION

Requirements

This example uses the following hardware and software components:

- configure a switching fabric interface on both nodes to configure Ethernet switching-related features on the nodes. See *Example: Configuring Switch Fabric Interfaces to Enable Switching in Chassis Cluster Mode on a Security Device*
- SRX240 security device
- Junos OS 12.3X48-D90
- interface-mode is supported in 15.1X49 release.
- port-mode is supported in 12.1 and 12.3X48 releases.

Overview

This example shows the configuration of a VLAN with members across node 0 and node 1.

Configuration

IN THIS SECTION

- Procedure | 791

Procedure

CLI Quick Configuration

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

```
set interfaces ge-0/0/3 unit 0 family ethernet-switching port-mode access
set interfaces ge-0/0/3 unit 0 family ethernet-switching vlan members vlan100
set interfaces ge0/0/4 unit 0 family ethernrt-switching port-mode access
set interfaces ge-0/0/4 unit 0 family ethernet-switching vlan members vlan100
set interfaces ge-7/0/5 unit 0 family ethernet-switching port-mode trunk
set interfaces ge-7/0/5 unit 0 family ethernet-switching vlan members vlan100
set interfaces vlan unit 100 family inet address 11.1.1.1/24
set vlans vlan100 vlan-id 100
set vlans vlan100 l3-interface vlan.100
```

Step-by-Step Procedure

To configure VLAN:

1. Configure Ethernet switching on the node0 interface.

```
{primary:node0} [edit]
user@host# set interfaces ge-0/0/3 unit 0 family ethernet-switching port-mode access
user@host# set interfaces ge0/0/4 unit 0 family ethernet-switching port-mode access
```

2. Configure Ethernet switching on the node1 interface.

```
{primary:node0} [edit]
user@host# set interfaces ge-7/0/5 unit 0 family ethernet-switching port-mode trunk
```

3. Create VLAN vlan100 with vlan-id 100.

```
{primary:node0} [edit]
user@host# set vlans vlan100 vlan-id 100
```

4. Add interfaces from both nodes to the VLAN.

```
{primary:node0} [edit]
user@host# set interfaces ge-0/0/3 unit 0 family ethernet-switching vlan members vlan100
user@host# set interfaces ge-0/0/4 unit 0 family ethernet-switching vlan members vlan100
user@host# set interfaces ge-7/0/5 unit 0 family ethernet-switching vlan members vlan100
```

5. Create a VLAN interface.

```
user@host# set interfaces vlan unit 100 family inet address 11.1.1.1/24
```

6. Associate an VLAN interface with the VLAN.

```
user@host# set vlans vlan100 l3-interface vlan.100
```

7. If you are done configuring the device, commit the configuration.

```
[edit]
user@host# commit
```

Results

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

```
[edit]
user@host# show vlans
vlan100 {
     vlan-id 100;
    l3-interface vlan.100;
}
[edit]
user@host# show interfaces
ge-0/0/3 {
    unit 0 {
        family ethernet-switching {
            port-mode access;
            vlan {
                members vlan100;
            }
        }
    }
}
ge-0/0/4 {
    unit 0 {
        family ethernet-switching {
            port-mode access;
            vlan {
                members vlan100;
            }
        }
    }
}
ge-7/0/5 {
    unit 0 {
        family ethernet-switching {
            port-mode trunk;
            vlan {
                members vlan100;
            }
        }
```

```
    }
}
```

Verification

IN THIS SECTION

- Verifying VLAN | **794**

Verifying VLAN

Purpose

Verify that the configuration of VLAN is working properly.

Action

From operational mode, enter the `show interfaces terse ge-0/0/3` command to view the node 0 interface.

```
user@host> show interfaces terse ge-0/0/3
Interface               Admin Link Proto    Local                 Remote
ge-0/0/3                up    up
ge-0/0/3.0              up    up   eth-switch
```

From operational mode, enter the `show interfaces terse ge-0/0/4` command to view the node 0 interface.

```
user@host> show interfaces terse ge-0/0/4
Interface               Admin Link Proto    Local                 Remote
ge-0/0/4                up    up
ge-0/0/4.0              up    up   eth-switch
```

From operational mode, enter the `show interfaces terse ge-7/0/5` command to view the node1 interface.

```
user@host> show interfaces terse ge-7/0/5
Interface               Admin Link Proto    Local                 Remote
```

```
ge-7/0/5                up    up
ge-7/0/5.0              up    up   eth-switch
```

From operational mode, enter the `show vlans` command to view the VLAN interface.

```
user@host> show vlans
Routing instance      VLAN name     Tag      Interfaces
default-switch        default        1
default-switch        vlan100        100     ge-0/0/3.0*
                                             ge-0/0/4.0*
                                             ge-7/0/5.0*
```

From operational mode, enter the `show ethernet-switching interface` command to view the information about Ethernet switching interfaces.

```
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down,
                         MMAS - Mac-move action shutdown,  AS - Autostate-exclude enabled,
                         SCTL - shutdown by Storm-control )

Logical         Vlan                  TAG   MAC    STP         Logical          Tagging
interface       members                     limit  state       interface flags
ge-0/0/3.0                                  16383              DN               untagged
                vlan100               100   1024   Discarding                   untagged
ge-0/0/4.0                                  16383              DN               untagged
                vlan100               100   1024   Discarding                   untagged
ge-7/0/5.0                                  16383              DN               tagged
                vlan100               100   1024   Discarding                   tagged
```

Meaning

The output shows the VLANs are configured and working fine.

SEE ALSO

- Example: Configuring an IRB Interface

Example: Configuring IRB Interfaces on QFX5100 Switches over an MPLS Core Network

IN THIS SECTION

Starting with Junos OS Release 14.1X53-D40 and Junos OS Release 17.1R1, QFX5100 switches support integrated routing and bridging (IRB) interfaces over an MPLS core network. An IRB interface is a logical Layer 3 VLAN interface used to route traffic between VLANs.

By definition, VLANs divide a LAN's broadcast environment into isolated virtual broadcast domains, thereby limiting the amount of traffic flowing across the entire LAN and reducing the possible number of collisions and packet retransmissions within the LAN. To forward packets between different VLANs, you traditionally needed a router that connects the VLANs. However, using the Junos OS you can accomplish this inter-VLAN forwarding without using a router by simply configuring an IRB interface on the switch.

The IRB interface functions as a logical switch on which you can configure a Layer 3 logical interface for each VLAN. The switch relies on its Layer 3 capabilities to provide this basic routing between VLANs. With an IRB interface, you can configure label-switched paths (LSPs) to enable the switch to recognize which packets are being sent to local addresses, so that they are bridged (switched) whenever possible and are routed only when necessary. Whenever packets can be switched instead of routed, several layers of processing are eliminated.

This example shows how to configure an IRB interface over an MPLS core network using QFX5100 switches.

Requirements

This example uses the following hardware and software components:

- Three QFX5100 switches
- Junos OS Release 14.1X53-D40 or later

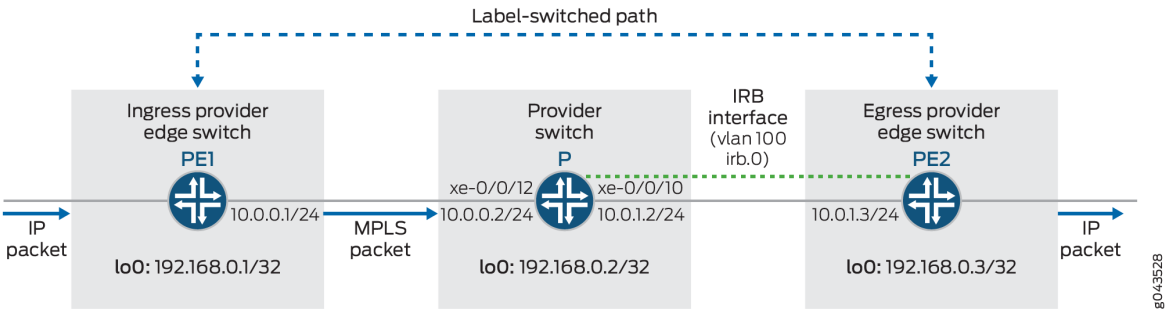
Before you begin, be sure you have:

- An understanding of IRB concepts. See "Understanding Integrated Routing and Bridging" on page 765 for an overview of IRB.
- The required ternary content addressable memory (TCAM) space available on the switch. TCAM rules must be observed while configuring and implementing IRBs. For detailed information, see *MPLS Limitations on QFX Series and EX4600 Switches*.

Overview and Topology

Figure 42 on page 797 illustrates a sample topology for configuring IRB over an MPLS core network. In this example, an LSP is established between the ingress provider edge switch (PE1) and the provider edge egress switch (PE2). An IRB Layer 3 interface (irb.0) is configured on switches P and PE2, and associated to VLAN 100. In this configuration, the P switch replaces (swaps) the label at the top of the label stack with a new label, adds the VLAN identifier 100 to the MPLS packet, and then sends the packet out the IRB interface. PE2 receives this vlan-tagged MPLS packet, removes (pops) the label from the top of the label stack, performs a regular IP route lookup, and then forwards the packet with its IP header to the next-hop address.

Figure 42: IRB Topology over an MPLS Core Network



Configuration

IN THIS SECTION

To configure the topology in this example, perform these tasks:

Configuring the Local Ingress PE Switch

CLI Quick Configuration

To quickly configure the local ingress PE switch (PE1), copy and paste the following commands into the switch terminal window of switch PE1:

```
set interfaces xe-0/0/12 unit 0 family inet address 10.0.0.1/24
set interfaces xe-0/0/12 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 65550
set policy-options policy-statement pplb then load-balance per-packet
set routing-options forwarding-table export pplb
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface em0.0 disable
set protocols mpls interface all
set protocols ldp interface xe-0/0/12.0
set protocols ldp interface lo0.0
```

Step-by-Step Procedure

To configure the ingress PE switch (PE1):

1. Configure the interfaces.

```
[edit interfaces]
user@switchPE1# set xe-0/0/12 unit 0 family inet address 10.0.0.1/24
user@switchPE1# set xe-0/0/12 unit 0 family mpls
user@switchPE1# set lo0 unit 0 family inet address 192.168.0.1/32
```

2. Configure the router ID and autonomous system (AS) number.

NOTE: We recommend that you explicitly configure the router identifier under the `[edit routing-options]` hierarchy level to prevent unpredictable behavior if the interface address on a loopback interface changes.

```
[edit routing-options]
user@switchPE1# set router-id 192.168.0.1/32
user@switchPE1# set autonomous-system 65550
```

3. Configure and apply an export routing policy to the forwarding table for per-packet load balancing.

```
[edit policy-options]
user@switchPE1# set policy-statement pplb then load-balance per-packet
[edit routing-options]
user@switchPE1# set forwarding-table export pplb
```

4. Create an OSPF area and set the loopback address to be passive.

```
[edit protocols ospf]
user@switchPE1# set area 0.0.0.0 interface all
user@switchPE1# set area 0.0.0.0 interface lo0.0 passive
user@switchPE1# set area 0.0.0.0 interface em0.0 disable
```

5. Enable MPLS on all interfaces.

```
[edit protocols mpls]
user@switchPE1# set interface all
```

6. Configure LDP on the provider-facing and loopback interfaces.

```
[edit protocols ldp]
user@switchPE1# set interface xe-0/0/12.0
user@switchPE1# set interface lo0.0
```

Results

Display the results of the PE1 switch configuration:

```
user@switchPE1# show
interfaces {
    xe-0/0/12 {
        unit 0 {
            family inet {
                address 10.0.0.1/24;
            }
            family mpls;
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 192.168.0.1/32;
                }
            }
        }
    }
}
routing-options {
    router-id 192.168.0.1;
    autonomous-system 65550;
    forwarding-table {
        export pplb;
    }
}
protocols {
    mpls {
        interface all;
    }
    ospf {
        area 0.0.0.0 {
            interface all;
            interface lo0.0 {
                passive;
            }
            interface em0.0 {
                disable;
```

```
            }
        }
    }
    ldp {
        interface xe-0/0/12.0
        interface lo0.0;
    }
}
policy-options {
    policy-statement pplb {
        then {
            load-balance per-packet;
        }
    }
}
```

Configuring the Provider Switch

CLI Quick Configuration

To quickly configure the provider switch (P), copy and paste the following commands into the switch terminal window of the P switch:

```
set interfaces xe-0/0/12 unit 0  family inet address 10.0.0.2/24
set interfaces xe-0/0/12 unit 0  family mpls
set interfaces xe-0/0/10 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/10 unit 0 family ethernet-switching vlan members v100
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set interfaces irb unit 0 family inet address 10.0.1.2/24
set interfaces irb unit 0 family mpls
set routing-options router-id 192.168.0.2
set routing-options autonomous-system 65550
set policy-options policy-statement pplb then load-balance per-packet
set routing-options forwarding-table export pplb
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface em0.0 disable
set protocols mpls interface all
set protocols ldp interface all
```

```
set vlans v100 vlan-id 100
set vlans v100 l3-interface irb.0
```

Step-by-Step Procedure

To configure the provider switch (P):

1. Configure the physical and loopback interfaces.

```
[edit interfaces]
user@switchP#  set xe-0/0/12 unit 0 family inet address 10.0.0.2/24
user@switchP#  set xe-0/0/12 unit 0 family mpls
user@switchP# set xe-0/0/10 unit 0 family ethernet-switching interface-mode trunk
user@switchP# set xe-0/0/10 unit 0 family ethernet-switching vlan members v100
user@switchP# set lo0 unit 0 family inet address 192.168.0.2/32
```

2. Configure an IRB interface.

```
[edit interfaces]
user@switchP# set irb unit 0 family inet address 10.0.1.2/24
user@switchP# set irb unit 0 family mpls
```

3. Configure the router ID and AS number.

NOTE: We recommend that you explicitly configure the router identifier under the `[edit routing-options]` hierarchy level to avoid unpredictable behavior if the interface address on a loopback interface changes.

```
[edit routing-options]
user@switchP# router-id 192.168.0.2
user@switchP# set autonomous-system 65550
```

4. Configure and apply an export routing policy to the forwarding table for per-packet load balancing.

```
[edit policy-options]
user@switchP# set policy-statement pplb then load-balance per-packet
```

```
[edit routing-options]
user@switchP# set forwarding-table export pplb
```

5. Enable OSPF and set the loopback address to passive.

```
[edit protocols ospf]
user@switchP# set area 0.0.0.0 interface all
user@switchP# set area 0.0.0.0 interface lo0.0 passive
user@switchP# set area 0.0.0.0 interface em0.0 disable
```

6. Enable MPLS on all interfaces.

```
[edit protocols mpls]
user@switchP# set interface all
```

7. Configure LDP to include all interfaces.

```
[edit protocols ldp]
user@switchP# set interface all
```

8. Create the VLAN and associate the IRB interface to it.

```
[edit vlans]
user@switchP# set v100 vlan-id 100
user@switchP# set v100 l3-interface irb.0
```

NOTE: Layer 3 interfaces on trunk ports allow the interface to transfer traffic between multiple VLANs. Within a VLAN, traffic is switched, while across VLANs, traffic is routed.

Results

Display the results of the provider switch configuration:

```
user@switchP# show
interfaces {
```

```
    xe-0/0/10 {
        unit 0 {
            family ethernet-switching {
                interface-mode trunk;
                vlan {
                    members v100;
                }
            }
        }
    }
    xe-0/0/12 {
        unit 0
        family inet {
            address 10.0.0.2/24;
        }
        family mpls;
    }
    irb {
        unit 0 {
            family inet {
                address 10.0.1.2/24;
            }
            family mpls;
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 192.168.0.2/32;
            }
        }
    }
}
```

```
routing-options {
    router-id 192.168.0.2;
    autonomous-system 65550;
    forwarding-table {
        export pplb;
```

```
    }
}
```

```
protocols {
    mpls {
        interface all;
    }
    ospf {
        area 0.0.0.0 {
            interface all;
            interface lo0.0 {
                passive;
            }
            interface em0.0 {
                disable;
            }
        }
    }
    ldp {
        interface all;
    }
}
```

```
policy-options {
    policy-statement pplb {
        then {
            load-balance per-packet;
        }
    }
}
```

```
vlans {
    v100 {
        vlan-id 100;
        l3-interface irb.0;
    }
}
```

Configuring the Remote Egress PE Switch

CLI Quick Configuration

To quickly configure the remote egress PE switch (PE2), copy and paste the following commands into the switch terminal window of PE2:

```
set interfaces xe-0/0/10 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/10 unit 0 family ethernet-switching vlan members v100
set interfaces irb unit 0 family inet address 10.0.1.3/24
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set interfaces irb unit 0 family mpls
set routing-options router-id 192.168.0.3
set routing-options autonomous-system 65550
set policy-options policy-statement pplb then load-balance per-packet
set routing-options forwarding-table export pplb
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface em0.0 disable
set protocols mpls interface all
set protocols ldp interface all
set vlans v100 vlan-id 100
set vlans v100 l3-interface irb.0
```

Step-by-Step Procedure

To configure the remote PE switch (PE2):

1. Configure the physical and loopback interfaces.

```
[edit interfaces]
user@switchPE2# set xe-0/0/10 unit 0 family ethernet-switching interface-mode trunk
user@switchPE2# set xe-0/0/10 unit 0 family ethernet-switching vlan members v100
user@switchPE2# set lo0 unit 0 family inet address 192.168.0.3/32
```

2. Configure an IRB interface.

```
[edit interfaces]
user@switchPE2# set irb unit 0 family inet address 10.0.1.3/24
user@switchPE2# set irb unit 0 family mpls
```

3. Configure the the router ID and AS number.

```
[edit routing-options]
user@switchPE2# set router-id 192.168.0.3/32
user@switchPE2# set autonomous-system 65550
```

4. Configure and apply an export routing policy to the forwarding table for per-packet load balancing.

```
[edit policy-options]
user@switchPE2# set policy-statement pplb then load-balance per-packet
[edit routing-options]
user@switchPE2# set forwarding-table export pplb
```

5. Enable OSPF.

```
[edit protocols ospf]
user@switchPE2# set area 0.0.0.0 interface all
user@switchPE2# set area 0.0.0.0 interface lo0.0 passive
user@switchPE2# set area 0.0.0.0 interface em0.0 disable
```

6. Enable MPLS on all interfaces.

```
[edit protocols mpls]
user@switchPE2# set interface all
```

7. Configure LDP to include all interfaces.

```
[edit protocols ldp]
user@switchPE2# set interface all
```

8. Create the VLAN and associate the IRB interface to it.

```
[edit vlans]
user@switchPE2# set v100 vlan-id 100
user@switchPE2# set v100 l3-interface irb.0
```

Results

Display the results of the PE2 switch configuration:

```
user@switchPE2# show
interfaces {
    xe-0/0/10 {
        unit 0 {
            family ethernet-switching {
                interface-mode trunk;
                vlan {
                    members v100;
                }
            }
        }
    irb {
        unit 0 {
            family inet {
                address 10.0.1.3/24;
            }
            family mpls;
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 192.168.0.3;
            }
        }
```

```
    }
}
```

```
routing-options {
    router-id 192.168.0.3;
    autonomous-system 65550;
    forwarding-table {
        export pplb;
    }
}
```

```
protocols {
    mpls {
        interface all;
    }
    ospf {
        area 0.0.0.0 {
            interface all;
            interface lo0.0 {
                passive;
            }
            interface em0.0 {
                disable;
            }
        }
    }
    ldp {
        interface all;
    }
}
```

```
policy-options {
    policy-statement pplb {
        then {
            load-balance per-packet;
        }
```

```
    }
}
```

```
vlans {
    v100 {
        vlan-id 100;
        l3-interface irb.0;
    }
}
```

Example: Configuring a Large Delay Buffer on a Security Device IRB Interface

IN THIS SECTION

This example shows how to configure a large delay buffer on an IRB interface to help slower interfaces avoid congestion and packet dropping when they receive large bursts of traffic.

Requirements

Before you begin, enable the large buffer feature on the IRB interface and then configure a buffer size for each queue in the CoS scheduler. See *Scheduler Buffer Size Overview*.

Overview

On devices, you can configure large delay buffers on an irb interfaces.

In this example, you configure scheduler map to associate schedulers to a defined forwarding class `be-class`, `ef-class` , `af-class`, and `nc-class` using scheduler map `large-buf-sched-map`. You apply scheduler maps to irb interface, and define per-unit scheduler for the IRB interface.

Configuration

IN THIS SECTION

- Procedure | 811

Procedure

CLI Quick Configuration

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

```
set class-of-service scheduler-maps large-buf-sched-map forwarding-class be-class scheduler be-scheduler
set class-of-service scheduler-maps large-buf-sched-map forwarding-class ef-class scheduler ef-scheduler
set class-of-service scheduler-maps large-buf-sched-map forwarding-class af-class scheduler af-scheduler
set class-of-service scheduler-maps large-buf-sched-map forwarding-class nc-class scheduler nc-scheduler
set class-of-service interfaces irb unit 0 scheduler-map large-buf-sched-map
set interfaces irb per-unit-scheduler
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the Junos OS CLI User Guide.

To configure a large delay buffer on a channelized T1 interface:

1. Configure the scheduler map to associate schedulers with defined forwarding classes.

```
[edit class-of-service]
set scheduler-maps large-buf-sched-map forwarding-class be-class scheduler be-scheduler
set scheduler-maps large-buf-sched-map forwarding-class ef-class scheduler ef-scheduler
```

```
set scheduler-maps large-buf-sched-map forwarding-class af-class scheduler af-scheduler
set scheduler-maps large-buf-sched-map forwarding-class nc-class scheduler nc-scheduler
```

2. Apply the scheduler map to the IRB interface.

```
[edit ]
user@host# set interfaces irb unit 0 scheduler-map large-buf-sched-map
```

3. Define the per-unit scheduler for the irb interface.

```
[edit ]
user@host# set interfaces irb per-unit-scheduler
```

Results

From configuration mode, confirm your configuration by entering the `show class-of-service` and `show chassis` commands. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
[edit]
    user@host# show class-of-service
    interfaces {
    irb {
        unit 0 {
            scheduler-map large-buf-sched-map;
        }
    }
}
scheduler-maps {
    large-buf-sched-map {
        forwarding-class be-class scheduler be-scheduler;
        forwarding-class ef-class scheduler ef-scheduler;
        forwarding-class af-class scheduler af-scheduler;
        forwarding-class nc-class scheduler nc-scheduler;
    }
}
```

If you are done configuring the device, enter `commit` from configuration mode.

Verification

IN THIS SECTION

Verifying Large Delay Buffers Configuration

Purpose

Verify that the large delay buffers are configured properly.

Action

From configuration mode, enter the `show class-of-service interface irb` command.

```
user@host> show class-of-service interface irb


Physical interface: irb, Index: 132
Maximum usable queues: 8, Queues in use: 4Code point type: dscp
Scheduler map: <default>, Index :2
Congestion-notification: Disabled
Logical interface: irb.10, Index: 73
Object                  Name                    Type                    Index
Classifier              ipprec-compatibility    ip                         13
```

Meaning

The large delay buffers are configured on IRB interface as expected.

SEE ALSO

Schedulers Overview

Default Scheduler Settings

Example: Configuring and Applying Scheduler Maps

Transmission Scheduling Overview

Configuring a Set of VLANs to Act as a Switch for a Layer 2 Trunk Port

You can configure a set of VLANs that are associated with a Layer 2 trunk port. The set of VLANs function as a switch. Packets received on a trunk interface are forwarded within a VLAN that has the same VLAN identifier. A trunk interface also provides support for IRB, which provides support for Layer 2 bridging and Layer 3 IP routing on the same interface.

To configure a Layer 2 trunk port and set of VLANs, include the following statements:

```
[edit interfaces]
interface-name {
    unit number {
        family ethernet-switching {
            interface-mode access;
            vlan-members (vlan-name | vlan-tag);
        }
    }
}
interface-name {
    native-vlan-id number;
    unit number {
        family ethernet-switching {
            interface-mode trunk;
            vlan-members (vlan-name | vlan-tag);
        }
    }
}
[edit vlans ]
vlan-name {
    vlan-id number;
    vlan-id-list [ vlan-id-numbers ];
    . . . .
}
```

You must configure a VLAN and VLAN identifier for each VLAN associated with the trunk interface. You can configure one or more trunk or access interfaces at the `[edit interfaces]` hierarchy level. An access interface enables you to accept packets with no VLAN identifier.

Excluding an IRB Interface from State Calculations on a QFX Series Switch

IRB interfaces are used to bind specific VLANs to Layer 3 interfaces, enabling a switch to forward packets between those VLANs— without having to configure another device, such as a router, to connect VLANs. Because an IRB interface often has multiple ports in a single VLAN, the state calculation for a VLAN member might include a port that is down, possibly resulting in traffic loss.

Starting with Junos OS Release 14.1X53-D40 and Junos OS Release 17.3R1 on QFX5100 switches, this feature enables you to exclude a trunk or access interface from the state calculation, which means that as soon as the port assigned to a member VLAN goes down, the IRB interface for the VLAN is also marked as down. In a typical scenario, one port on the interface is assigned to a single VLAN, while a second port on that interface is assigned to a trunk interface that carries traffic between multiple VLANs. A third port is often also assigned to an access interface to connect the VLAN to network devices.

Before you begin:

- Configure VLANs
- Configure IRB interfaces for the VLANs.

  For more information about configuring IRB interfaces, see "Example: Configuring Routing Between VLANs on One Switch Using an IRB Interface" on page 778.

To exclude an access or 802.1Q trunk interface from the state calculations for an IRB interface:

1. Configure a trunk or access interface.

   ```
   [edit interfaces interface-name]
   user@switch# set unit logical-unit-number family ethernet-switching port-mode (access | trunk)
   ```

   For example, configure interface xe-0/1/0.0 as a trunk interface:

   ```
   [edit interfaces xe-0/1/0]
   user@swtich# set unit 0 family ethernet-switching port-mode trunk
   ```

2. Assign VLAN members to the access or trunk interface.

   ```
   [edit interfaces interface-name unit logical-unit-number ethernet-switching]
   user@switch# set vlan members [ (all | names | vlan-ids) ]
   ```

For example, assign all VLAN members configured on the device to the trunk interface xe-0/1/0:

```
[edit interfaces xe-0/1/0 unit 0 ethernet-switchg]
user@switch# set vlan members all
```

3. Exclude an access or trunk interface from state calculations for the IRB interfaces for member VLANs.

```
[edit interfaces interface-name ether-options]
user@switch# set autostate-exclude
```

For example, exclude the trunk interface xe-0/1/0 from state calculations for the IRB interfaces for member VLANs:

```
[edit interfaces xe-0/1/0]
user@switch# set autostate-exclude
```

4. To confirm your configuration, from configuration mode, enter the `show interfaces xe-0/1/0` command. If your output does not display the intended configuration, repeat steps 1 through 4 to correct the configuration.

```
user@switch# show interfaces xe-0/1/0
ether-options {
    autostate-exclude;
}
unit 0 {
    family ethernet-switching {
        port-mode trunk;
        vlan {
            members all;
        }
    }
}
```

5. After you commit the configuration, issue the `show ethernet-switching interface xe-0/1/0.0` to verify that the logical interface is enabled with `autostate-exclude`.

```
user@switch> show ethernet-switching interface xe-0/1/0.0
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
```

```
                        LH - MAC limit hit, DN - interface down,
                        SCTL - shutdown by Storm-control,
                        MMAS - Mac-move action shutdown, AS - Autostate-exclude enabled)

Logical          Vlan          TAG     MAC         STP         Logical           Tagging
interface        members               limit       state       interface flags
xe-0/1/0.0                             294912                   AS                untagged
                 vlan_100      100     294912      Forwarding                    untagged
```

The `AS` in the `Logical interface flags` field indicates that `autostate-exclude` is enabled and that this interface will be excluded from the state calculations for the IRB interfaces for the member VLANs.

Verifying Integrated Routing and Bridging Interface Status and Statistics on EX Series Switches

IN THIS SECTION

Purpose

Determine status information and traffic statistics for integrated routing and bridging (IRB) interfaces.

Action

Display IRB interfaces and their current states:

```
user@switch> show interfaces irb  terse
Interface               Admin Link Proto    Local                 Remote
irb                     up    up
irb.111                 up    up   inet     10.111.111.1/24
...
```

Display Layer 2 VLANs, including any tags assigned to the VLANs and the interfaces associated with the VLANs:

```
user@switch> show vlans
Routing instance        VLAN name             Tag         Interfaces
default-switch          irb                     101

default-switch          support                 111

                                                            ge-0/0/18.0

...
```

Display Ethernet switching table entries for the VLAN that is attached to the IRB interface:

```
user@switch> show ethernet-switching table
MAC flags (S -static MAC, D -dynamic MAC, L -locally learned
           SE -Statistics enabled, NM -Non configured MAC, R -Remote PE MAC)

Routing instance : default-switch
    Vlan                MAC                 MAC         Age    Logical
    Name                address             flags              interface
    support             00:01:02:03:04:05   S             -   ge-0/0/18.0
...
```

Display the ingress-counting statistics of an IRB interface with either the **show interfaces irb detail** command or the **show interfaces irb extensive** command. Ingress counting is displayed as **Input bytes** and **Input packets** and egress counting is displayed as `Output bytes` and `Output packets` under **Transit Statistics**.

```
user@switch> show interfaces irb .111 detail

Logical interface irb.111 (Index 65) (SNMP ifIndex 503) (HW Token 100) (Generation 131)
Flags: SNMP-Traps 0x4000  Encapsulation: ENET2
Bandwidth: 1000mbps
Routing Instance: default-switch Bridging Domain: irb+111
Traffic statistics:
    Input bytes:       17516756
    Output bytes:      411764
    Input packets:     271745
    Output packets:    8256
```

```
Local statistics:
    Input bytes:       3240
    Output bytes:      411764
    Input packets:     54
    Output packets:    8256
Transit statistics:
    Input bytes:       17513516      0 bps
    Output bytes:      0                0 bps
    Input packets:     271745        0 pps
    Output packets:    0                0 pps
Protocol inet, MTU: 1514, Generation: 148, Route table: 0
    Flags: None
    Addresses, Flags: iS-Preferred  Is-Primary
        Destination: 10.1.1/24, Local: 10.1.1.1, Broadcast: 10.1.1.255, Generation: 136
```

Meaning

- `show interfaces irb terse` displays a list of interfaces, including IRB interfaces, and their current states (up, down).
- `show vlans` displays a list of VLANs, including any tags assigned to the VLANs and the interfaces associated with the VLANs.
- `show ethernet-switching table` displays the Ethernet switching table entries, including VLANs attached to the IRB interface.
- `show interfaces irb detail` displays IRB interface ingress counting as `Input Bytes` and `Input Packets` under `Transit Statistics`.

Change History Table

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

Release	Description
14.1X53-D40	Starting with Junos OS Release 14.1X53-D40 and Junos OS Release 17.1R1, QFX5100 switches support integrated routing and bridging (IRB) interfaces over an MPLS core network.
14.1X53-D40	Starting with Junos OS Release 14.1X53-D40 and Junos OS Release 17.3R1 on QFX5100 switches, this feature enables you to exclude a trunk or access interface from the state calculation, which means that as soon as the port assigned to a member VLAN goes down, the IRB interface for the VLAN is also marked as down.

26

CHAPTER

Configuring VLANS and VPLS Routing Instances

IN THIS CHAPTER

VLANs and VPLS Routing Instances

IN THIS SECTION

Guidelines for Configuring VLAN Identifiers for VLANs and VPLS Routing Instances

For a VLAN that is performing Layer 2 switching only, you do not have to specify a VLAN identifier.

For a VLAN that is performing Layer 3 IP routing, you must specify either a VLAN identifier or dual VLAN identifier tags.

For a VPLS routing instance, you must specify either a VLAN identifier or dual VLAN identifier tags.

SEE ALSO

Layer 2 Learning and Forwarding for VLANs Overview | 102

Configuring VLAN Identifiers for VLANs and VPLS Routing Instances

You can configure VLAN identifiers for a VLAN or a VPLS routing instance in the following ways:

- By using either the `vlan-id` statement or the `vlan-tags` statement to configure a normalizing VLAN identifier. This topic describes how normalizing VLAN identifiers are processed and translated in a VLAN or a VPLS routing instance.
- By using the **input-vlan-map** and the `output-vlan-map` statements at the `[edit interfaces` *`interface-name`* `unit` *`logic-unit-number`*`]` or `[edit logical-systems` *`logical-system-name`* `interfaces` *`interface-name`* `unit` *`logic-unit-number`*`]` hierarchy level to configure VLAN mapping.

The **vlan-id** and `vlan-tags` statements are used to specify the normalizing VLAN identifier under the VLAN or VPLS routing instance. The normalizing VLAN identifier is used to perform the following functions:

- Translate, or normalize, the VLAN tags of packets received into a learn VLAN identifier.
- Create multiple learning domains that each contain a learn VLAN identifier. A learning domain is a MAC address database to which MAC addresses are added based on the learn VLAN identifier.

NOTE: You cannot configure VLAN mapping using the **input-vlan-map** and `output-vlan-map` statements if you configure a normalizing VLAN identifier for a VLAN or VPLS routing instance using the **vlan-id** or `vlan-tags` statements.

To configure a VLAN identifier for a VLAN, include either the **vlan-id** or the `vlan-tags` statement at the `[edit interfaces` *`interface-name`* `unit` *`logic-unit-number`*`]` or `[edit logical-systems` *`logical-system-name`* `interfaces` *`interface-name`* `unit` *`logic-unit-number`*`]` hierarchy level, and then include that logical interface in the VLAN configuration.

For a VPLS routing instance, include either the **vlan-id** or `vlan-tags` statement at the `[edit interfaces` *`interface-name`* `unit` *`logic-unit-number`*`]` or `[edit logical-systems` *`logical-system-name`* `interfaces` *`interface-name`* `unit` *`logic-unit-number`*`]` hierarchy level, and then include that logical interface in the VPLS routing instance configuration.

NOTE: ACX Series routers do not support the `[edit logical-systems]` hierarchy.

NOTE: For a single VLAN or VPLS routing instance, you can include either the **vlan-id** or the `vlan-tags` statement, but not both. If you do not configure a **vlan-id**, **vlan-tags**, or **vlan-id-list [** ***vlan-id-numbers*** **]** for the VLAN or the VPLS routing instance, the Layer 2 packets received are forwarded to the outbound Layer 2 interface without having the VLAN tag modified unless an **output-vlan-map** is configured on the Layer 2 interface. This results in a frame being forwarded to a Layer 2 interface with a VLAN tag that is different from what is configured for the Layer 2 interface. Note that a frame received from the Layer 2 interface is still required to match the VLAN tag(s) specified in the interface configuration. The invalid configuration may cause a Layer 2 loop to occur.

The VLAN tags associated with the inbound logical interface are compared with the normalizing VLAN identifier. If the tags are different, they are rewritten as described in Table 114 on page 825. The source MAC address of a received packet is learned based on the normalizing VLAN identifier.

NOTE: You do not have to specify a VLAN identifier for a VLAN that is performing Layer 2 switching only. To support Layer 3 IP routing, you must specify either a VLAN identifier or a pair of VLAN tags. However, you cannot specify the same VLAN identifier for more than one VLAN within a routing instance. Each VLAN must have a unique VLAN identifier.

If the VLAN tags associated with the outbound logical interface and the normalizing VLAN identifier are different, the normalizing VLAN identifier is rewritten to match the VLAN tags of the outbound logical interface, as described in Table 115 on page 827.

For the packets sent over the VPLS routing instance to be tagged by the normalizing VLAN identifier, include one of the following configuration statements:

- **vlan-id** ***number*** to tag all packets that are sent over the VPLS virtual tunnel (VT) interfaces with the VLAN identifier.
- **vlan-tags outer** ***number*** **inner** ***number*** to tag all packets sent over the VPLS VT interfaces with dual outer and inner VLAN tags.

Use the `vlan-id none` statement to have the VLAN tags removed from packets associated with an inbound logical interface when those packets are sent over VPLS VT interfaces. Note that those packets might still be sent with other customer VLAN tags.

The `vlan-id all` statement enables you to configure bridging for several VLANs with a minimum amount of configuration. Configuring this statement creates a learning domain for:

- Each inner VLAN, or learn VLAN, identifier of a logical interface configured with two VLAN tags
- Each VLAN, or learn VLAN, identifier of a logical interface configured with one VLAN tag

We recommend that you do not use customer VLAN IDs in a VPLS routing instance because customer VLAN IDs are used for learning only.

You should use the service VLAN ID in a VPLS routing instance, as in the following configuration:

```
[edit]
interface ge-1/1/1 {
    vlan-tagging;
    unit 1 {
        vlan-id s1; /* Service vlan */
        encapsulation vlan-vpls;
        input-vlan-map pop; /* Pop the service vlan on input */
        output-vlan-map push; /* Push the service vlan on output */
    }
```

```
}
interface ge-1/1/2 {
    encapsulation ethernet-vpls;
    unit 0;
}
routing-instances {
    V1 {
        instance-type vpls;
        vlan-id all;
        interface ge-1/1/1.1;
        interface ge-1/1/2.0;
    }
}
```

NOTE: If you configure the `vlan-id all` statement in a VPLS routing instance, we recommend using the **input-vlan-map pop** and `output-vlan-map push` statements on the logical interface to pop the service VLAN ID on input and push the service VLAN ID on output and in this way, limit the impact of double-tagged frames on scaling. You cannot use the native `vlan- id` statement when the `vlan-id all` statement is included in the configuration.

The `vlan-id-list [ vlan-id-numbers ]` statement enables you to configure bridging for multiple VLANs on a trunk interface. Configuring this statement creates a learning domain for:

- Each VLAN listed: `vlan-id-list [ 100 200 300 ]`
- Each VLAN in a range: `vlan-id-list [ 100-200 ]`
- Each VLAN in a list and range combination: `vlan-id-list [ 50, 100-200, 300 ]`

The following steps outline the process for bridging a packet received over a Layer 2 logical interface when you specify a normalizing VLAN identifier using either the **vlan-id *number*** or `vlan-tags` statement for a VLAN or a VPLS routing instance:

1. When a packet is received on a physical port, it is accepted only if the VLAN identifier of the packet matches the VLAN identifier of one of the logical interfaces configured on that port.
2. The VLAN tags of the received packet are then compared with the normalizing VLAN identifier. If the VLAN tags of the packet are different from the normalizing VLAN identifier, the VLAN tags are rewritten as described in Table 114 on page 825.
3. If the source MAC address of the received packet is not present in the source MAC table, it is learned based on the normalizing VLAN identifier.

4. The packet is then forwarded toward one or more outbound Layer 2 logical interfaces based on the destination MAC address. A packet with a known unicast destination MAC address is forwarded only to one outbound logical interface. For each outbound Layer 2 logical interface, the normalizing VLAN identifier configured for the VLAN or VPLS routing instance is compared with the VLAN tags configured on that logical interface. If the VLAN tags associated with an outbound logical interface do not match the normalizing VLAN identifier configured for the VLAN or VPLS routing instance, the VLAN tags are rewritten as described in Table 115 on page 827.

The tables below show how VLAN tags are applied for traffic sent to and from the VLAN, depending on how the **vlan-id** and `vlan-tags` statements are configured for the VLAN and on how identifiers are configured for the logical interfaces in a VLAN or VPLS routing instance. Depending on your configuration, the following rewrite operations are performed on VLAN tags:

- **pop**—Remove a VLAN tag from the top of the VLAN tag stack.
- **pop-pop**—Remove both the outer and inner VLAN tags of the frame.
- **pop-swap**—Remove the outer VLAN tag of the frame and replace the inner VLAN tag of the frame.
- **swap**—Replace the VLAN tag of the frame.
- **push**—Add a new VLAN tag to the top of the VLAN stack.
- **push-push**—Push two VLAN tags in front of the frame.
- **swap-push**—Replace the VLAN tag of the frame and add a new VLAN tag to the top of the VLAN stack.
- **swap-swap**—Replace both the outer and inner VLAN tags of the frame.

Table 114 on page 825 shows specific examples of how the VLAN tags for packets sent to the VLAN are processed and translated, depending on your configuration. "–" means that the statement is not supported for the specified logical interface VLAN identifier. "No operation" means that the VLAN tags of the received packet are not translated for the specified input logical interface.

Table 114: Statement Usage and Input Rewrite Operations for VLAN Identifiers for a VLAN

VLAN Identifier of Logical Interface	VLAN Configurations for a VLAN			
	vlan-id none	vlan-id 200	vlan-id all	vlan tags outer 100 inner 300
none	No operation	push 200	–	push 100, push 300

Table 114: Statement Usage and Input Rewrite Operations for VLAN Identifiers for a VLAN *(Continued)*

VLAN Identifier of Logical Interface	VLAN Configurations for a VLAN			
	vlan-id none	vlan-id 200	vlan-id all	vlan tags outer 100 inner 300
200	pop 200	No operation	No operation	swap 200 to 300, push 100
1000	pop 1000	swap 1000 to 200	No operation	swap 1000 to 300, push 100
vlan-tags outer 2000 inner 300	pop 2000, pop 300	pop 2000, swap 300 to 200	pop 2000	swap 2000 to 100
vlan-tags outer 100 inner 400	pop 100, pop 400	pop 100, swap 400 to 200	pop 100	swap 400 to 300
vlan-id-range 10-100	–	–	No operation	–
vlan-tags outer 200 inner-range 10-100	–	–	pop 200	–

Table 115 on page 827 shows specific examples of how the VLAN tags for packets sent from the VLAN are processed and translated, depending on your configuration. "–" means that the statement is not supported for the specified logical interface VLAN identifier. "No operation" means that the VLAN tags of the outbound packet are not translated for the specified output logical interface.

Table 115: Statement Usage and Output Rewrite Operations for VLAN Identifiers for a VLAN

VLAN Identifier of Logical Interface	VLAN Configurations for a VLAN			
	vlan-id none	vlan-id 200	vlan-id all	vlan tags outer 100 inner 300
none	no operation	pop 200	–	pop 100, pop 300
200	push 200	No operation	No operation	pop 100, swap 300 to 200
1000	push 1000	swap 200 to 1000	No operation	pop 100, swap 300 to 1000
vlan-tags outer 2000 inner 300	push 2000, push 300	swap 200 to 300, push 2000	push 2000	swap 100 to 2000
vlan-tags outer 100 inner 400	push 100, push 400	swap 200 to 400, push 100	push 100	swap 300 to 400
vlan-id-range 10-100	–	–	No operation	–
vlan-tags outer 200 inner-range 10-100	–	–	push 200	–

27

CHAPTER

Configuring Multiple VLAN Registration Protocol (MVRP)

IN THIS CHAPTER

Multiple VLAN Registration Protocol

IN THIS SECTION

Understanding Multiple VLAN Registration Protocol (MVRP)

IN THIS SECTION

Multiple VLAN Registration Protocol (MVRP) is a Layer 2 messaging protocol that automates the creation and management of virtual LANs, thereby reducing the time you have to spend on these tasks. Use MVRP on Juniper Networks switches to dynamically register and unregister active VLANs on trunk interfaces. Using MVRP means that you do not have to manually register VLANs on all connections—that is, you do not need to explicitly bind a VLAN to each trunk interface. With MVRP, you configure a VLAN on one switch interface and the VLAN configuration is distributed through all active switches in the domain.

NOTE: MVRP is an application protocol of the Multiple Registration Protocol (MRP) and is defined in the IEEE 802.1ak standard. MRP and MVRP replace Generic Attribute Registration Protocol (GARP) and GARP VLAN Registration Protocol (GVRP) and overcome GARP and GVRP limitations.

NOTE: MVRP on QFabric systems does not support private VLANs.

If your QFabric system connects to servers that host many virtual machines that require their own VLANs, using MVRP can save you the time and effort that would be required to manually create and administer the VLANs on the ports that connect to the servers. For example, if a virtual machine moves between servers—and therefore connects to a different redundant server Node group interface—MVRP can configure the appropriate VLAN membership on the new server Node group interface.

When using MVRP on a QFabric system, you must manually create on the QFabric the VLANs that exist on the attached servers because the QFabric implementation of MVRP does not allow VLANs to be created dynamically. However, you do not need to manually assign VLAN membership to the QFabric ports that connect to the servers. MVRP automatically assigns VLAN membership to server-facing QFabric ports when it learns about a VLAN from an attached server.

MVRP Operations

MVRP stays synchronized by using MVRP protocol data units (PDUs). These PDUs specify which QFabric systems and switches are members of which VLANs, and which switch interfaces are in each VLAN. The MVRP PDUs are sent to other switches in the QFabric system when an MVRP state change occurs, and the receiving switches update their MVRP states accordingly. MVRP timers dictate when PDUs can be sent and when switches receiving MVRP PDUs can update their MVRP information.

In addition to this behavior, QFX switches include a mode—called passive mode—in which an MVRP-configured interface does not announce its membership in a VLAN or send any VLAN declarations (updates) unless it receives registration for that VLAN from a peer (server) on that interface. By default MVRP-configured interfaces behave in the standard manner and automatically send PDU updates to announce any VLAN changes. (This is called active mode.)

To enable passive mode on an interface, enter and commit this statement:

set protocols mvrp interface *interface-name* passive

To keep VLAN membership information current, MVRP removes switches and interfaces when they become unavailable. Pruning VLAN information has these benefits:

- Limits the network VLAN configuration to active participants, thereby reducing network overhead.
- Limits broadcast, unknown unicast, and multicast (BUM) traffic to interested devices.

MVRP is disabled by default and is valid only for trunk interfaces.

How MVRP Updates, Creates, and Deletes VLANs on Switches

When any MVRP-member VLAN is changed, that VLAN sends a protocol data unit (PDU) to all other MVRP-member active VLANs. The PDU informs the other VLANs which switches and interfaces currently belong to the sending VLAN. This way, all MVRP-member VLANs are always updated with the current VLAN state of all other MVRP-member VLANs. Timers dictate when PDUs can be sent and when switches receiving MVRP PDUs can update their MVRP VLAN information.

In addition to sending PDU updates, MVRP dynamically creates VLANs on member interfaces when a new VLAN is added to any one interface. This way, VLANs created on one member switch are propagated to other member switches as part of the MVRP message exchange process.

To keep VLAN membership information current, MVRP removes switches and interfaces when they become unavailable. Pruning VLAN information has these benefits:

- Limits the network VLAN configuration to active participants, thereby reducing network overhead.
- Limits broadcast, unknown unicast, and multicast (BUM) traffic to interested devices.

MVRP Is Disabled by Default on Switches

MVRP is disabled by default on the switches and, when enabled, affects only trunk interfaces. Once you enable MVRP, all VLAN interfaces on the switch belong to MVRP (the default **normal** registration mode) and those interfaces accept PDU messages and send their own PDU messages. To prevent one or more interfaces from participating in MVRP, you can specifically configure an interface to **forbidden** registration mode instead of the default **normal** mode.

VLAN updating, dynamic VLAN configuration through MVRP, and VLAN pruning are all active on trunk interfaces when MVRP is enabled.

MRP Timers Control MVRP Updates

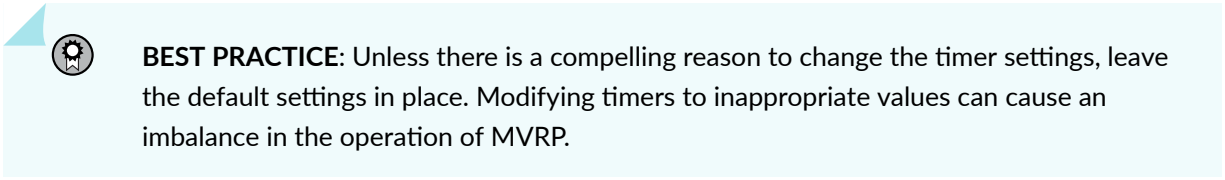
MVRP registration and updates are controlled by timers that are part of the MRP. The timers define when MVRP PDUs can be sent and when MVRP information can be updated on a switch.

The timers are set on a per-interface basis, and on EX Series switches that use Juniper Networks Junos operating system (Junos OS) with support for the Enhanced Layer 2 Software (ELS) configuration style, the timers are also set on a per-switch basis.

On an EX Series switch that uses Junos OS with support for ELS, if the timer value set on an interface level is different from the value set on a switch level, the value on the interface level takes precedence.

The following MRP timers are used to control the operation of MVRP:

- Join timer—Controls the interval for the next MVRP PDU transmit opportunity.
- Leave timer—Controls the period of time that an interface on the switch waits in the leave state before changing to the unregistered state.
- LeaveAll timer—Controls the frequency with which the interface generates LeaveAll messages.



BEST PRACTICE: Unless there is a compelling reason to change the timer settings, leave the default settings in place. Modifying timers to inappropriate values can cause an imbalance in the operation of MVRP.

MVRP Uses MRP Messages to Transmit Switch and VLAN States

MVRP uses MRP messages to register and declare MVRP states for a switch or VLAN and to inform the switching network that a switch or VLAN is leaving MVRP. These messages are communicated as part of the PDU sent by any switch interface to the other switches in the network.

The following MRP messages are communicated for MVRP:

- Empty—MVRP information is not declared and no VLAN is registered.
- In—MVRP information is not declared but a VLAN is registered.
- JoinEmpty—MVRP information is declared but no VLAN is registered.
- JoinIn—MVRP information is declared and a VLAN is registered.
- Leave—MVRP information that was previously declared is withdrawn.
- LeaveAll—Unregister all VLANs on the switch. VLANs must re-register to participate in MVRP.
- New—The MVRP information is new and a VLAN might not be registered yet.

Compatibility Issues with Junos OS Releases of MVRP

Except in Junos OS Releases 11.2 and earlier, MVRP has conformed with IEEE standard 802.1ak and IEEE Draft 802.1Q regarding the inclusion of an extra byte in the protocol data units (PDUs) sent and

received by MVRP. Table 116 on page 833 outlines the MVRP versions and whether or not each version includes the extra byte in the PDU. Table 116 on page 833 also labels each MVRP version with a scenario number, which is used throughout the remainder of this discussion for brevity.

Table 116: Junos OS MVRP Versions and Inclusion of Extra Byte in PDU

MVRP in Junos OS Releases 11.2 and Earlier For EX Series Switches That Do Not Support Enhanced Layer 2 Software (ELS) Configuration Style Scenario 1	MVRP in Junos OS Releases 11.3 and Later For EX Series Switches That Do Not Support ELS Scenario 2	MVRP in Junos OS Releases 13.2 and Later For EX Series Switches With Support For ELS Scenario 3
Includes extra byte in the PDU	By default, does not include extra byte in the PDU	By default, includes extra byte in the PDU

As a result of the non-conformance of Releases 11.2 and earlier and changes in the standards with regard to the extra byte, a compatibility issue exists between some of the Junos OS versions of MVRP. This issue can result in some versions of MVRP not recognizing PDUs without the extra byte.

To address this compatibility issue, the MVRP versions described in scenarios 2 and 3 include the ability to control whether or not the PDU includes the extra byte. Before using these controls, however, you must understand each scenario that applies to your environment and plan carefully so that you do not inadvertently create an additional compatibility issue between the MVRP versions in scenarios 2 and 3.

Table 117 on page 833 provides a summary of environments that include the various MVRP scenarios and whether or not a particular environment requires you to take action.

Table 117: MVRP Environments and Description of Required Actions

Environment	Action Required?	Action Description
Includes MVRP versions in scenario 1 only	No	—
Includes MVRP versions in scenario 2 only	No	—
Includes MVRP versions in scenario 3 only	No	—

Table 117: MVRP Environments and Description of Required Actions *(Continued)*

Environment	Action Required?	Action Description
Includes MVRP versions in scenarios 1 and 2. MVRP version in scenario 2 is in its default state.	Yes	On switches running MVRP version in scenario 2, use the `add-attribute-length-in-pdu` statement. For more information, see "Configuring Multiple VLAN Registration Protocol (MVRP) on Switches " on page 839.
Includes MVRP versions in scenarios 1 and 3. MVRP version in scenario 3 is in its default state.	No	—
Includes MVRP versions in scenarios 2 and 3, and both versions are in their respective default states	Yes	Do one of the following: • On switches running MVRP version in scenario 2, use the `add-attribute-length-in-pdu` statement. For more information, see "Configuring Multiple VLAN Registration Protocol (MVRP) on Switches " on page 839. • On switches running MVRP version in scenario 3, use the `no-attribute-length-in-pdu` statement. For more information, see "Configuring Multiple VLAN Registration Protocol (MVRP) on Switches " on page 839.

QFabric Requirements

When configuring MVRP on a QFabric system, you can enable it globally or enable it only on the trunk ports that need to carry VLAN traffic from the attached servers. You also must manually create the expected VLANs, but you do not have to assign VLAN membership to the server-facing redundant server Node ports (as mentioned previously). However, you *do* have to manually assign VLAN membership to the uplink ports on the redundant server Node group and network Node group devices that will carry the VLAN traffic. Table 118 on page 835 summarizes the VLAN requirements for redundant server Node groups and network Node groups:

Table 118: MVRP VLAN Requirements for Node Devices

Node Group Type	Interface	Assign VLAN Membership to Trunk Ports?
Redundant server Node group	Server-facing trunk	No
Redundant server Node group	Uplink trunk (to interconnect device)	Yes
Network Node groups	Uplink trunk (to interconnect device)	Yes

Determining Whether MVRP is Working

You can determine whether the switches in your network are running incompatible versions of MVRP by issuing the **show mvrp statistics** command. For more information on diagnosing and correcting this MVRP compatibility situation, see "Configuring Multiple VLAN Registration Protocol (MVRP) on Switches " on page 839.

SEE ALSO

Understanding Bridging and VLANs on Switches | 140

Understanding Multiple VLAN Registration Protocol (MVRP) for Dynamic VLAN Registration

IN THIS SECTION

Multiple VLAN Registration Protocol (MVRP) is a Layer 2 messaging protocol that manages the addition, deletion, and renaming of active virtual LANs, thereby reducing network administrators' time spent on these tasks. Use MVRP on Juniper Networks MX Series routers, EX Series switches and SRX Series Firewalls to dynamically register and unregister active VLANs on trunk interfaces. Using MVRP means that you do not have to manually register VLANs on all connections—that is, you do not need to explicitly bind a VLAN to each trunk interface. With MVRP, you configure a VLAN on one interface and the VLAN configuration is distributed through all active interfaces in the domain.

The primary purpose of MVRP is to manage dynamic VLAN registration in Layer 2 networks. In managing dynamic VLAN registration, MVRP also prunes VLAN information.

MVRP is an Layer 2 application protocol of the Multiple Registration Protocol (MRP) and is defined in the IEEE 802.1ak standard. MRP and MVRP were designed by IEEE to perform the same functions as Generic Attribute Registration Protocol (GARP) and GARP VLAN Registration Protocol (GVRP) while overcoming some GARP and GVRP limitations, in particular, limitations involving bandwidth usage and convergence time in large networks with large numbers of VLANs.

MVRP was created by IEEE as a replacement application for GVRP. MVRP and GVRP cannot be run concurrently to share VLAN information in a Layer 2 network.

This topic describes:

How MVRP Works

When any MVRP-member VLAN is changed, that VLAN sends a protocol data unit (PDU) to all other MVRP-member active VLANs. The PDU informs the other VLANs which devices and interfaces currently belong to the sending VLAN. This way, all MVRP-member VLANs are always updated with the current VLAN state of all other MVRP-member VLANs. Timers dictate when PDUs can be sent and when devices receiving MVRP PDUs can update their MVRP VLAN information.

The VLAN registration information sent by MVRP protocol data units (PDUs) includes the current VLANs membership—that is, which routers are members of which VLANs—and which router interfaces are in which VLAN. MVRP shares all information in the PDU with all routers participating in MVRP in the Layer 2 network.

MVRP stays synchronized using these PDUs. The routers in the network participating in MVRP receive these PDUs during state changes and update their MVRP states accordingly. MVRP timers dictate when PDUs can be sent and when routers receiving MVRP PDUs can update their MVRP information.

In addition to sending PDU updates, MVRP dynamically creates VLANs on member interfaces when a new VLAN is added to any one interface. This way, VLANs created on one member device are propagated to other member devices as part of the MVRP message exchange process.

VLAN information is distributed as part of the MVRP message exchange process and can be used to dynamically create VLANs, which are VLANs created on one switch and propagated to other routers as

part of the MVRP message exchange process. Dynamic VLAN creation using MVRP is enabled by default, but can be disabled.

As part of ensuring that VLAN membership information is current, MVRP removes routers and interfaces from the VLAN information when they become unavailable. Pruning VLAN information has these benefits:

- Limits the network VLAN configuration to active participants only, reducing network overhead.
- Targets the scope of broadcast, unknown unicast, and multicast (BUM) traffic to interested devices only.

Using MVRP

MVRP is disabled by default on the devices and, when enabled, affects only trunk interfaces. Once you enable MVRP, all VLAN interfaces on the device belong to MVRP (the default **normal** registration mode) and those interfaces accept PDU messages and send their own PDU messages. To prevent one or more interfaces from participating in MVRP, you can specifically configure an interface to **forbidden** registration mode instead of the default **normal** mode.

VLAN updating, dynamic VLAN configuration through MVRP, and VLAN pruning are all active on trunk interfaces when MVRP is enabled.

MVRP Registration Modes

The MVRP registration mode defines whether an interface does or does not participate in MVRP.

The following MVRP registration modes are configurable:

- forbidden—The interface does not register or declare VLANS (except statically configured VLANs).
- normal—The interface accepts MVRP messages and participates in MVRP. This is the default registration mode setting.
- restricted—The interface ignores all MVRP JOIN messages received for VLANs that are not statically configured on the interface.

MRP Timers Control MVRP Updates

MVRP registration and updates are controlled by timers that are part of the MRP protocol. These timers are set on a per-interface basis and define when MVRP PDUs can be sent and when MVRP information can be updated on a switch.

The following timers are used to control the operation of MVRP:

- Join timer—Controls the interval for the next MVRP PDU transmit opportunity.

- Leave timer—Controls the period of time that an interface on the switch waits in the Leave state before changing to the unregistered state.
- LeaveAll timer—Controls the frequency with which the interface generates LeaveAll messages.

BEST PRACTICE: Maintain default timer settings unless there is a compelling reason to change the settings. Modifying timers to inappropriate values might cause an imbalance in the operation of MVRP.

MVRP Uses MRP Messages to Transmit Device and VLAN States

MVRP uses MRP messages to register and declare MVRP states for a switch and to inform the Layer 2 network that a switch is leaving MVRP. These messages are communicated as part of the PDU to communicate the state of a particular switch interface on the Layer 2 network to the other switches in the network.

The following messages are communicated for MVRP:

- Empty—VLAN information is not being declared and is not registered.
- In—VLAN information is not being declared but is registered.
- JoinEmpty—VLAN information is being declared but not registered.
- JoinIn—VLAN information is being declared and is registered.
- Leave—VLAN information that was previously registered is being withdrawn.
- LeaveAll—All registrations will be de-registered. Participants that want to participate in MVRP will need to re-register.
- New—VLAN information is new and possibly not previously registered.

MVRP Limitations

The following limitations apply when configuring MVRP:

- MVRP works with Rapid Spanning Tree Protocol (RSTP) and Multiple Spanning Tree Protocol (MSTP), but not with VLAN Spanning Tree Protocol (VSTP).
- MVRP is allowed only on single tagged trunk ports.
- MVRP is not allowed if a physical interface has more than one *logical interface*.
- MVRP is only allowed if a logical has one trunk interface (unit 0).

Configuring Multiple VLAN Registration Protocol (MVRP) on Switches

IN THIS SECTION

Multiple VLAN Registration Protocol (MVRP) is used to manage dynamic VLAN registration in a LAN. You can use MVRP on QFX switches and on EX Series switches that support or do not support ELS.

MVRP is disabled by default.

To enable MVRP or set MVRP options, follow these instructions:

Enabling MVRP on Switches With ELS Support

This example uses Junos OS for EX Series switches with support for the Enhanced Layer 2 Software (ELS) configuration style. MVRP can only be enabled on trunk interfaces.

NOTE: For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15.

MVRP can only be enabled on trunk interfaces. To enable MVRP on a trunk interface:

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

Enabling MVRP on Switches Without ELS Support

This example uses Junos OS for EX Series switches that do not support the Enhanced Layer 2 Software (ELS) configuration style. For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15.

Multiple VLAN Registration Protocol (MVRP) is used to manage dynamic VLAN registration in a LAN. You can use MVRP on EX Series switches.

MVRP is disabled by default on EX Series switches.

MVRP can only be enabled on trunk interfaces. To enable MVRP on a trunk interface:

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

To enable MVRP on a specific trunk interface:

```
[edit protocols mvrp]
user@switch# set interface xe-0/0/1.0
```

Enabling MVRP on Switches With QFX Support

Multiple VLAN Registration Protocol (MVRP) automates the creation and management of VLANs. When using MVRP on a QFabric system, you must manually create on the QFabric the VLANs that exist on the attached servers because the QFabric implementation of MVRP does not allow VLANs to be created dynamically. However, you do not need to manually assign VLAN membership to the QFabric ports that connect to the servers. MVRP automatically assigns VLAN membership to server-facing QFabric ports when it learns about a VLAN from an attached server. .

MVRP is disabled by default. To enable MVRP or set MVRP options, follow these instructions:

This example uses Junos OS for EX Series switches with support for the Enhanced Layer 2 Software (ELS) configuration style. MVRP can only be enabled on trunk interfaces.

NOTE: For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15.

MVRP can only be enabled on trunk interfaces. To enable MVRP on a trunk interface:

```
[edit protocols mvrp]
user@qfabric# set interface interface-name
```

NOTE: On QFX Series switches, you must configure specific interfaces—you cannot specify `interface all`. You can enable MVRP on an interface range.

Disabling MVRP

MVRP is disabled by default. Perform this procedure only if you have previously enabled MVRP.

You can disable MVRP globally only. To disable MVRP on all trunk interfaces on a switch with ELS support, use one of the following commands:

```
user@switch# deactivate protocols mvrp
user@switch# delete protocols mvrp
```

To disable MVRP on all trunk interfaces of a QFX switch, an EX switch without ELS Support or an entire QFabric system:

```
[edit protocols mvrp]
user@switch# set disable
```

To disable MVRP on a specific trunk QFX switch or an EX switch without interface support:

```
[edit protocols mvrp]
user@qfabric# set disable interface interface-name
```

```
[edit protocols mvrp]
user@switch# set disable interface xe-0/0/1.0
```

SEE ALSO

disable

add-attribute-length-in-pdu

no-attribute-length-in-pdu

Disabling Dynamic VLANs on EX Series Switches

By default, dynamic VLANs can be created on interfaces participating in MVRP. Dynamic VLANs are VLANs created on one switch that are propagated to other switches dynamically, in this case, using MVRP.

Dynamic VLAN creation through MVRP cannot be disabled per switch interface. To disable dynamic VLAN creation for interfaces participating in MVRP, you must disable it for all interfaces on the switch.

To disable dynamic VLAN creation:

```
[edit protocols mvrp]
user@switch# set no-dynamic-vlan
```

SEE ALSO

- *no-dynamic-vlan*
- *add-attribute-length-in-pdu*
- *no-attribute-length-in-pdu*

Configuring Timer Values

The timers in MVRP define the amount of time all interfaces on a switch or a specific interface wait to join or leave MVRP, or to send or process the MVRP information for the switch after receiving an MVRP PDU. The join timer controls the amount of time the switch waits to accept a registration request, the leave timer controls the period of time that the switch waits in the Leave state before changing to the unregistered state, and the leaveall timer controls the frequency with which the LeaveAll messages are communicated.

The default MVRP timer values are 200 ms for the join timer, 1000 ms for the leave timer, and 10 seconds for the leaveall timer.

BEST PRACTICE: Maintain default timer settings unless there is a compelling reason to change the settings. Modifying timers to inappropriate values might cause an imbalance in the operation of MVRP.

On an EX Series switch that uses Junos OS with support for ELS, if the timer value set on an interface level is different from the value set on a switch level, then the value on the interface level takes precedence.

To set the join timer for all interfaces on the switch:

```
[edit protocols mvrp]
user@switch# set join-timer milliseconds
```

```
[edit protocols mvrp]
user@switch# set interface all join-timer 300
```

To set the join timer for a specific interface:

```
[edit protocols mvrp]
user@switch# set interface interface-name join-timer milliseconds
```

```
[edit protocols mvrp]
user@qfabric# set interface interface-name 300
```

```
[edit protocols mvrp]
user@switch# set interface xe-0/0/1.0 300
```

To set the leave timer for all interfaces on the switch:

```
[edit protocols mvrp]
user@switch# set leave-timer milliseconds
```

```
[edit protocols mvrp]
user@switch# set interface all leave-timer 1200
```

To set the leave timer for a specific interface:

```
[edit protocols mvrp]
user@switch# set interface interface-name leave-timer milliseconds
```

```
[edit protocols mvrp]
user@qfabric# set interface interface-name leave-timer 1200
```

To set the leaveall timer for all interfaces on the switch:

```
[edit protocols mvrp]
user@switch# set leaveall-timer seconds
```

```
[edit protocols mvrp]
user@qfabric# set interface interface-name leaveall-timer 12000
```

```
[edit protocols mvrp]
user@switch# set interface all leaveall-timer 12000
```

To set the leaveall timer for a specific interface:

```
[edit protocols mvrp]
user@switch# set interface interface-name leaveall-timer seconds
```

```
[edit protocols mvrp]
user@switch# set interface xe-0/0/1.0 leaveall-timer 12000
```

SEE ALSO

- *join-timer (MVRP)*
- *leave-timer (MVRP)*
- *leaveall-timer (MVRP)*

Configuring Passive Mode on QFX Switches

QFX switches include a mode—called passive mode—in which an MVRP-configured interface does not announce its membership in a VLAN or send any VLAN declarations (updates) unless it receives registration for that VLAN from a peer (server).

To configure an interface to operate in passive mode:

```
[edit protocols mvrp]
user@qfabric# set interface interface-name passive
```

Configuring MVRP Registration Mode on EX Switches

NOTE: Not supported in QFabric.

The default MVRP registration mode for any interface participating in MVRP is normal. An interface in normal registration mode participates in MVRP when MVRP is enabled on the switch.

You can change the registration mode of a specific interface to `forbidden`. An interface in forbidden registration mode does not participate in MVRP even if MVRP is enabled on the switch.

To set an interface to forbidden registration mode:

```
[edit protocols mvrp]
user@switch# set interface xe-0/0/1.0 registration forbidden
```

```
[edit protocols mvrp]
user@switch# set interface all registration forbidden
```

To set an interface to normal registration mode:

```
[edit protocols mvrp]
user@switch# set interface xe-0/0/1.0 registration normal
```

```
[edit protocols mvrp]
user@switch# set interface xe-0/0/1.0 registration normal
```

To set all interfaces to normal registration mode:

```
[edit protocols mvrp]
user@switch# set interface all registration normal
```

SEE ALSO

> *registration*

Using MVRP in a Mixed-Release EX Series Switching Network

Except in Junos OS Releases 11.2 and earlier, MVRP has conformed with IEEE standard 802.1ak and IEEE Draft 802.1Q regarding the inclusion of an extra byte in the protocol data units (PDUs) sent and received by MVRP.

As a result of the non-comformance of releases 11.2 and earlier and changes in the standards regarding the extra byte, the following mixed environments can arise in EX Series switches without ELS support:

- Mixed environment A: MVRP in Junos OS Releases 11.2 and earlier includes the extra byte, while MVRP in Junos OS Releases 11.3 and later for EX Series switches that do not support the Enhanced Layer 2 Software (ELS) configuration style does not include the extra byte.
- Mixed environment B: MVRP in Junos OS Releases 13.2 and later for EX Series switches with support for ELS includes the extra byte, while MVRP in Junos OS Releases 11.3 and later for EX Series switches that do not support ELS does not include the extra byte.

As a result of changes in the standards with regard to the extra byte, MVRP in Junos OS Releases 13.2 and later for EX Series switches with support for the Enhanced Layer 2 Software (ELS) includes the extra byte, while MVRP in Junos OS Releases 11.3 and later for EX Series switches that do not support ELS does not include the extra byte. A compatibility issue arises, wherein the ELS version of MVRP does not recognize PDUs without the extra byte sent by the non-ELS version of MVRP.

A compatibility issue arises in mixed environments A and B, wherein the versions of MVRP that include the extra byte do not recognize PDUs that do not include the extra byte.

If your network has a mix of MVRP versions, you can alter MVRP on the switches running Release 11.3 and later on switches that do not support ELS so they include the extra byte in the PDU and are therefore, compatible with the other MVRP versions.

A compatibility issue arises in mixed environments A and B, wherein the versions of MVRP that include the extra byte do not recognize PDUs that do not include the extra byte.

For more information about these issues, see "Understanding Multiple VLAN Registration Protocol (MVRP)" on page 829.

To make MVRP on on switches that do not support ELS (Release 11.3 or later) compatible with MVRP in the other releases:

```
[edit protocols mvrp]
user@switch# set add-attribute-length-in-pdu
```

If your network includes a mix of EX Series switches running ELS and non-ELS versions of MVRP, you can eliminate the compatibility issue by entering the following command on the switches running the ELS version of MVRP:

```
[edit protocols mvrp]
user@switch# set no-attribute-length-in-pdu
```

The `no-attribute-length-in-pdu` statement prevents the ELS version of MVRP from sending PDUs with the extra byte, thereby eliminating the compatibility issue with the non-ELS version of MVRP.

You can recognize an MVRP version compatibility issue by observing the switch running the ELS version of MVRP. Because a switch running the ELS version of MVRP cannot interpret an unmodified PDU from a switch running the non-ELS version of MVRP, the switch will not add VLANs from the non-ELS version of MVRP. When you use the `show mvrp statistics` command in the ELS version of MVRP, the values for `Received Join Empty` and `Received Join In` will incorrectly display zero, even though the value for the `Received MVRP PDUs without error` has been increased. Another indication that MVRP is having a version compatibility issue is that unexpected VLAN activity, such as multiple VLAN creation, takes place on the switch running the ELS version of MVRP.

Configuring Multiple VLAN Registration Protocol (MVRP) to Manage Dynamic VLAN Registration on Security Devices

IN THIS SECTION

Starting in Junos OS Release 15.1X49-D80, Multiple VLAN Registration Protocol (MVRP) to manage dynamic VLAN registration is supported on SRX1500 devices. Multiple VLAN Registration Protocol (MVRP) is used to manage dynamic VLAN registration in a Layer 2 network. You can configure MVRP on SRX Series Firewalls.

MVRP is disabled by default on SRX Series Firewalls.

To enable MVRP and to set MVRP options, follow these instructions:

Enabling MVRP

MVRP can be enabled only on trunk interfaces.

To enable MVRP on a specific trunk interface (here, interface ge-0/0/1):

```
[edit protocols mvrp]
user@host# set interface ge-0/0/1
```

Changing the Registration Mode to Disable Dynamic VLANs

When the registration mode for an interface is set to `normal` (the default), dynamic VLANs are created on interfaces participating in MVRP. The dynamic VLANs created on one SRX Series Firewall are then propagated by means of MVRP to other SRX Series Firewalls in the topology.

However, dynamic VLAN creation through MVRP can be disabled for all trunk interfaces or for individual trunk interfaces.

Configuring Timer Values

The timers in MVRP define the amount of time an interface waits to join or leave MVRP or to send or process the MVRP information for the router or switch after receiving an MVRP PDU:

- The join timer controls the amount of time the router or switch waits to accept a registration request.
- The leave timer controls the period of time that the router or switch waits in the Leave state before changing to the unregistered state.
- The leaveall timer controls the frequency with which the LeaveAll messages are communicated.

The default MVRP timer values are 200 ms for the join timer, 1000 ms for the leave timer, and 60 seconds for the leaveall timer.

BEST PRACTICE: Maintain default timer settings unless there is a compelling reason to change the settings. Modifying timers to inappropriate values might cause an imbalance in the operation of MVRP.

To set the join timer at 300 ms for a specific interface (here, interface ge-0/0/1):

```
[edit protocols mvrp]
user@host# set interface ge-0/0/1 join-timer (MVRP) 300
```

To set the leave timer at 400 ms for a specific interface (here, interface ge-0/0/1):

```
[edit protocols mvrp]
user@host# set interface ge-0/0/1 leave-timer 400
```

To set the leaveall timer at 20 seconds for a specific interface (here, interface ge-0/0/1):

```
[edit protocols mvrp]
user@host# set interface ge-0/0/1 leaveall-timer 20
```

SEE ALSO

join-timer (MVRP)

leave-timer (MVRP)

leaveall-timer (MVRP)

Configuring the Multicast MAC Address for MVRP

MVRP uses the customer MVRP multicast MAC address when MVRP is enabled. However, you can configure MVRP to use the provider MVRP multicast MAC address instead.

To configure MVRP to use the provider MVRP multicast MAC address:

```
[edit protocols mvrp]
user@host# set bpdu-destination-mac-address provider-bridge-group;
```

SEE ALSO

bpdu-destination-mac-address

Configuring an MVRP Interface as a Point-to-Point Interface

Specify that a configured interface is connected point-to-point. If specified, a point-to-point subset of the MRP state machine provides a simpler and more efficient method to accelerate convergence on the network.

To specify that an MVRP interface is point-to-point (here, interface ge-0/0/1):

```
[edit protocols mvrp]
user@host# set interface ge-0/0/1 point-to-point (MVRP) ;
```

SEE ALSO

point-to-point (MVRP)

Configuring MVRP Tracing Options

Set MVRP protocol-level tracing options.

To specify MVRP protocol tracing (here, the file is `/var/log/mvrp-log`, size is `2m`, number of files is `28`, the option `world-readable` indicates the log can be read by user, and MVRP is flagging `events`):

```
[edit protocols mvrp]
user@host# edit traceoptions file /var/log/mvrp-log size 2m files 28 world-readable flag events
```

Disabling MVRP

MVRP is disabled by default. You need to perform this procedure only if MVRP is previously enabled.

To disable MVRP on all trunk interfaces, use one of the following commands:

```
[edit]
user@host# deactivate protocols mvrp
user@host# delete protocols mvrp
```

SEE ALSO

Understanding VLANs

Example: Configuring Automatic VLAN Administration on QFX Switches Using MVRP

IN THIS SECTION

As the numbers of servers and VLANs attached to a QFabric systems increase, VLAN administration becomes complex and the task of efficiently configuring VLANs on multiple redundant server Node group devices becomes increasingly difficult. To partially automate VLAN administration, you can enable Multiple VLAN Registration Protocol (MVRP) on your QFabric system. If your QFabric system connects to servers that host many virtual machines that require their own VLANs, using MVRP can save you the time and effort that would be required to manually configure and administer the VLANs on the interfaces that connect to the servers. For example, if a virtual machine moves between servers—and therefore connects to a different redundant server Node group interface—MVRP can configure the appropriate VLAN membership on the new server Node group interface.

NOTE: Only trunk interfaces can be enabled for MVRP.

This example describes how to configure MVRP on a QFabric system.

Requirements

This example uses the following hardware and software components:

- One QFabric system
- Junos OS Release 13.1 for the QFX Series

Overview and Topology

IN THIS SECTION

MVRP ensures that the VLAN membership information on the trunk interface is updated as the switch's access interfaces become active or inactive in the configured VLANs in a static or dynamic VLAN creation setup.

You do not need to explicitly bind a VLAN to the trunk interface. When MVRP is enabled, the trunk interface advertises all the VLANs that are active (bound to access interfaces) on that switch. An MVRP-enabled trunk interface does not advertise VLANs that have been configured on the switch but that are not currently bound to an access interface. Thus, MVRP provides the benefit of reducing network overhead—by limiting the scope of broadcast, unknown unicast, and multicast (BUM) traffic to interested devices only.

When VLAN access interfaces become active or inactive, MVRP ensures that the updated information is advertised on the trunk interface. Thus, in this example, distribution Switch C does not forward traffic to inactive VLANs.

A redundant server Node group device is connected to a server that hosts virtual machines for three customers, each of which requires its own VLAN.

- `customer-1`: VLAN ID 100
- `customer-2`: VLAN ID 200
- `customer-3`: VLAN ID 300

Topology

Table 119 on page 852 explains the components of the example topology.

Table 119: Components of the Example Topology

Settings	Settings
Hardware	• Redundant server Node group device • Network Node group device

Table 119: Components of the Example Topology *(Continued)*

Settings	Settings
VLAN names and IDs	• customer-1, VLAN ID (tag)100 • customer-2, VLAN ID (tag)200 • customer-3, VLAN ID (tag)300
Interfaces	Redundant server Node group device interfaces: • RSNG:xe-0/1/1—Uplink to interconnect device • RSNG:xe-0/0/1—Server-facing interface Network Node group device interface: • NNG:xe-0/0/1—Uplink to interconnect device

Configuring VLANs and Network Node Group Interfaces

IN THIS SECTION

To configure VLANs, bind the VLANs to the server-facing trunk interface, and enable MVRP on the trunk interface of the network Node group device, perform these tasks:

Procedure

CLI Quick Configuration

To quickly configure VLANs on the QFabric system, assign VLAN membership to the uplink port on the network Node group device, and configure the uplink port to be trunk:

```
[edit]
set vlans customer-1 vlan-id 100
 set vlans customer-2 vlan-id 200
```

```
 set vlans customer-3 vlan-id 300
set interfaces NNG:xe-0/0/1 unit 0 family ethernet-switching port-mode trunk
set interfaces NNG:xe-0/0/1 unit 0 family ethernet-switching vlan members [customer-1 customer-2
customer-3]
```

NOTE: As recommended as a best practice, default MVRP timers are used in this example, so they are not configured. The default values associated with each MVRP timer are: 200 ms for the join timer, 1000 ms for the leave timer, and 10000 ms for the leaveall timer. Modifying timers to inappropriate values might cause an imbalance in the operation of MVRP.

Step-by-Step Procedure

To create the VLANs and configure the network Node group device for MVRP, follow these steps. Note that you are creating VLANs for the entire QFabric system, so you do not need to create them on specific QFabric devices.

1. Configure the VLAN for customer 1:

```
[edit]
user@qfabric# set vlans customer-1 vlan-id 100
```

2. Configure the VLAN for customer 2:

```
[edit]
user@qfabric# set vlans customer-2 vlan-id 200
```

3. Configure the VLAN for customer 3:

```
[edit]
user@qfabric# set vlans customer-3 vlan-id 300
```

4. Configure an uplink interface (one that connects to an interconnect device) to be a trunk:

```
[edit]
user@qfabric# set interfaces NNG:xe-0/0/1 unit 0 family ethernet-switching port-mode trunk
```

5. Configure the uplink interface to be a member of all three VLANs:

```
[edit]
user@qfabric# set interfaces NNG:xe-0/0/1 unit 1 family ethernet-switching vlan members
[customer-1 customer-2 customer-3]
```

NOTE: If you want the uplink interface to be a member of all the VLANs in the QFabric system, you can enter `all` instead of specifying the individual VLANs.

Results

Check the results of the configuration on the network Node group device:

```
[edit]

user@qfabric# show interfaces NNG:xe-0/0/1.0
family ethernet-switching {
    port-mode trunk;
    vlan {
        members customer-1 customer-2 customer-3;
    }
}
```

```
[edit]

user@qfabric# show vlans
customer-1 {
    vlan-id 100;
}
customer-2 {
    vlan-id 200;
}
customer-3 {
    vlan-id 300;
}
```

Configuring the Redundant Server Node Group

IN THIS SECTION

Procedure

CLI Quick Configuration

To quickly configure the redundant server Node group device for MVRP:

```
[edit]

set interfaces RSNG:xe-0/1/1 unit 0 family ethernet-switching port-mode trunk
set interfaces RSNG:xe-0/1/1 unit 0 family ethernet-switching vlan members [customer-1
customer-2 customer-3]
set interfaces RSNG:xe-0/0/1 unit 0 family ethernet-switching port-mode trunk
set protocols mvrp interface RSNG:xe-0/0/1.0 passive
```

Step-by-Step Procedure

To configure the redundant server Node group device, follow these steps. Note that you do not need to configure the VLANs on the server-facing interface (RSNG:xe-0/0/1), but you do need to configure the VLANs on the uplink interface. Also notice that in this example you configure the server-facing interface to be passive, which means that it will not announce its membership in a VLAN or send any VLAN declarations (updates) unless it receives registration for that VLAN from the server.

1. Configure an uplink interface (one that connects to the interconnect device) to be a trunk:

```
[edit]
user@qfabric# set interfaces RSNG:xe-0/1/1 unit 0 family ethernet-switching port-mode trunk
```

2. Configure the uplink interface to be a member of all three VLANs:

```
[edit]
user@qfabric# set interfaces NNG:xe-0/1/1 unit 0 family ethernet-switching vlan members
[customer-1 customer-2 customer-3]
```

3. Configure an interface that connects to the server that hosts multiple virtual machines to be a trunk:

```
[edit]
user@qfabric# set interfaces RSNG:xe-0/0/1 unit 0 family ethernet-switching port-mode trunk
```

4. Enable MVRP on the server-facing trunk interface and configure it to be passive:

```
[edit]
user@qfabric# set protocols mvrp interface RSNG:xe-0/0/1.0 passive
```

Results

Check the results of the configuration for the redundant server Node group:

```
[edit]

user@qfabric# show interfaces RSNG:xe-0/0/1.0
family ethernet-switching {
    port-mode trunk;
}
```

```
[edit]

user@qfabric# show interfaces RSNG:xe-0/1/1.0
family ethernet-switching {
    port-mode trunk;
    }
```

```
passive
}
```

```
[edit]

user@qfabric# show protocols mvrp
interface RSNG:xe-0/0/1.0;
```

Verification

IN THIS SECTION

- Verifying That MVRP Is Enabled On The QFabric System | **858**

To confirm that the configuration is updating VLAN membership, perform these tasks:

Verifying That MVRP Is Enabled On The QFabric System

Purpose

Verify that MVRP is enabled on the appropriate interfaces

Action

Show the MVRP configuration:

```
user@qfabric> show mvrp

MVRP configuration
MVRP status                 : Enabled

MVRP timers (ms):
Interface         Join   Leave       LeaveAll
--------------    -----  --------    -----------
NNG:xe-0/0/1.0    200     1000       10000
RSNG:xe-0/0/1.0   200     1000       10000
RSNG:xe-0/1/1.0   200     1000       10000
```

```
Interface           Status        Registration Mode
--------------  --------      -----------------
NNG:xe-0/0/1.0      Enabled       Normal
RSNG:xe-0/1/1.0     Enabled       Normal
RSNG:xe-0/0/1.0     Enabled       Passive
```

Meaning

The results show that MVRP is enabled on the appropriate network Node group and redundant server Node group interfaces and that the default timers are used.

Example: Configuring Automatic VLAN Administration Using MVRP on EX Series Switches with ELS Support

IN THIS SECTION

NOTE: This example uses Junos OS for EX Series switches with support for the Enhanced Layer 2 Software (ELS) configuration style. If your switch runs software that does not support ELS, see "Example: Configuring Automatic VLAN Administration Using MVRP on EX Series Switches" on page 879. For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15.

As a network expands and the number of clients and VLANs increases, VLAN administration becomes complex and the task of efficiently configuring VLANs on multiple EX Series switches becomes

increasingly difficult. However, you can automate VLAN administration by enabling Multiple VLAN Registration Protocol (MVRP) on the network.

MVRP also dynamically creates VLANs, further simplifying the network overhead required to statically configure VLANs.

NOTE: Only trunk interfaces can be enabled for MVRP.

This example describes how to use MVRP to automate administration of VLAN membership changes within your network and how to use MVRP to dynamically create VLANs:

Requirements

This example uses the following hardware and software components:

- Two EX Series access switches
- One EX Series distribution switch
- Junos OS Release 13.2X50-D10 or later for EX Series switches

Before you configure MVRP on an interface, you must enable one of the following spanning tree protocols on that interface:

- Rapid Spanning-Tree Protocol (RSTP). For more information about RSTP, see *Understanding RSTP*.
- Multiple Spanning-Tree Protocol (MSTP). For more information about MSTP, see *Understanding MSTP*.

Overview and Topology

IN THIS SECTION

MVRP is used to manage dynamic VLAN registration in a LAN. It can also be used to dynamically create VLANs.

This example uses MVRP to dynamically create VLANs on the switching network. Alternatively, you can disable dynamic VLAN creation and create VLANs statically. Enabling MVRP on the trunk interface of each switch in your switching network ensures that the active VLAN information for the switches in the

network is propagated to each switch through the trunk interfaces, assuming dynamic VLAN creation is enabled for MVRP.

MVRP ensures that the VLAN membership information on the trunk interface is updated as the switch's access interfaces become active or inactive in the configured VLANs in a static or dynamic VLAN creation setup.

You do not need to explicitly bind a VLAN to the trunk interface. When MVRP is enabled, the trunk interface advertises all the VLANs that are active (bound to access interfaces) on that switch. An MVRP-enabled trunk interface does not advertise VLANs that are configured on the switch but are not currently bound to an access interface. Thus, MVRP provides the benefit of reducing network overhead —by limiting the scope of broadcast, unknown unicast, and multicast (BUM) traffic to interested devices only.

When VLAN access interfaces become active or inactive, MVRP ensures that the updated information is advertised on the trunk interface. Thus, in this example, distribution Switch C does not forward traffic to inactive VLANs.

NOTE: This example shows a network with three VLANs: **finance**, **sales**, and **lab**. All three VLANs are running the same version of Junos OS. If switches in this network were running a mix of Junos OS releases that included Release 11.3, additional configuration would be necessary—see "Configuring Multiple VLAN Registration Protocol (MVRP) on Switches " on page 839 for details.

Topology

Figure 43 on page 862 shows MVRP configured on two access switches and one distribution switch.

Figure 43: MVRP Configured on Two Access Switches and One Distribution Switch for Automatic VLAN Administration

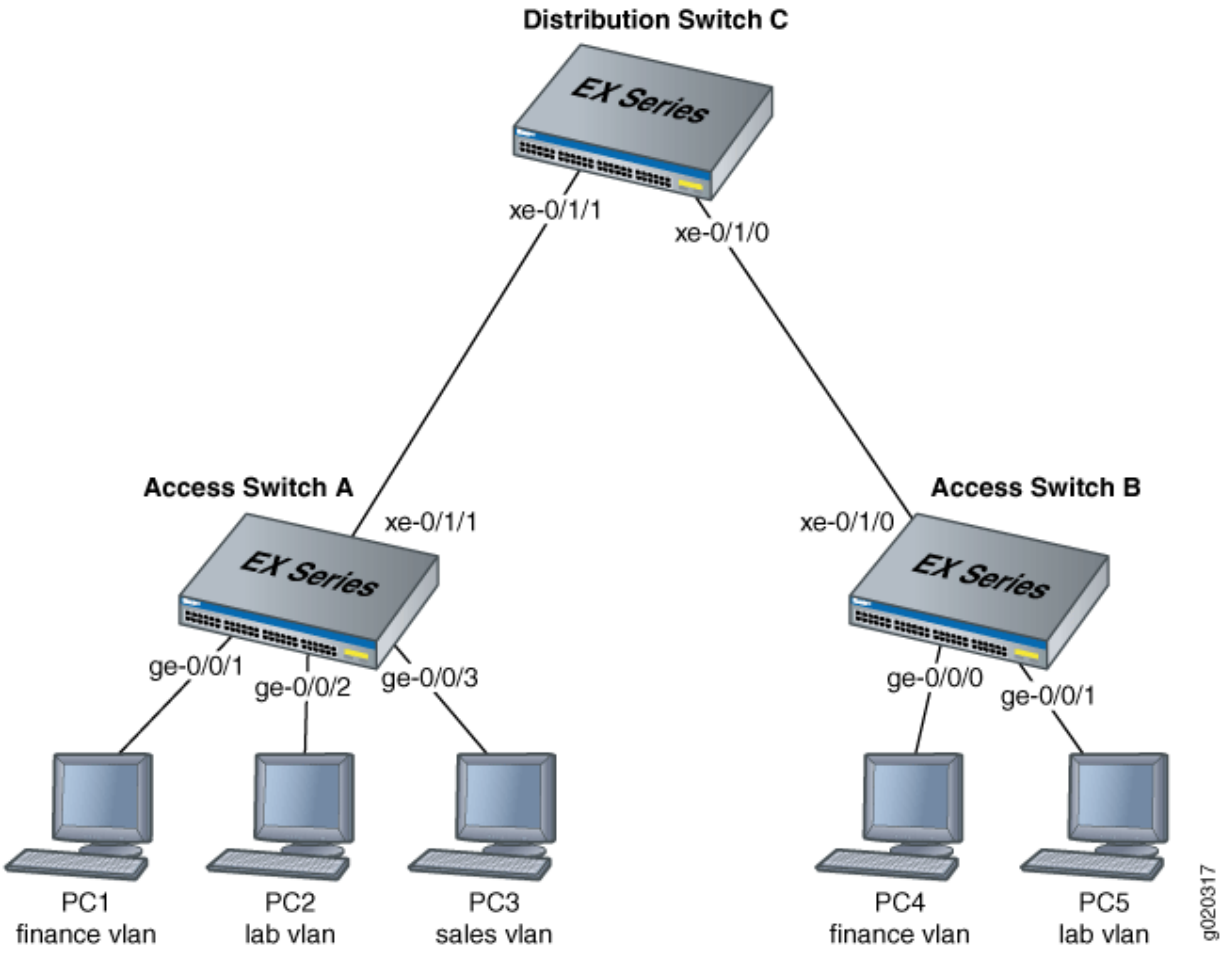


Table 120 on page 862 explains the components of the example topology.

Table 120: Components of the Network Topology

Settings	Settings
Switch hardware	• Access Switch A • Access Switch B • Distribution Switch C

Table 120: Components of the Network Topology *(Continued)*

Settings	Settings
VLAN names and tag IDs	**finance**, tag **100** **lab**, tag **200** **sales**, tag **300**
Interfaces	Access Switch A interfaces: • ge-0/0/1—Connects PC1 to access Switch A. • ge-0/0/2—Connects PC2 to access Switch A. • ge-0/0/3—Connects PC3 to access Switch A. • xe-0/1/1—Connects access Switch A to distribution Switch C (trunk). Access Switch B interfaces: • ge-0/0/0—Connects PC4 to access Switch B. • ge-0/0/1—Connects PC5 to access Switch B. • ge-0/0/2—Reserved for future use, • xe-0/1/0—Connects access Switch B to distribution Switch C. (trunk) Distribution Switch C interfaces: • xe-0/1/1—Connects distribution Switch C to access Switch A. (trunk) • xe-0/1/0—Connects distribution Switch C to access Switch B. (trunk)

Configuring VLANs and MVRP on Access Switch A

IN THIS SECTION

To configure VLANs on the switch, bind access interfaces to the VLANs, and enable MVRP on the trunk interface of access Switch A, perform these tasks:

Procedure

CLI Quick Configuration

To quickly configure access Switch A for MVRP, copy the following commands and paste them into the switch terminal window of Switch A:

```
[edit]
set vlans finance vlan-id 100
 set vlans lab vlan-id 200
 set vlans sales vlan-id 300
set interfaces ge-0/0/1 unit 0 family ethernet-switching vlan members finance
set interfaces ge-0/0/2 unit 0 family ethernet-switching vlan members lab
set interfaces ge-0/0/3 unit 0 family ethernet-switching vlan members sales
set interfaces xe-0/1/1 unit 0 family ethernet-switching interface-mode trunk
                        set protocols mvrp interface xe-0/1/1
```

NOTE: This example uses default MVRP timers. The default values associated with each MVRP timer are: 200 ms for the join timer, 1000 ms for the leave timer, and 10000 ms (10 seconds) for the leaveall timer. We recommend retaining the use of default timer values as modifying timers to inappropriate values might cause an imbalance in the operation of MVRP. However, if you choose to change the default settings, keep in mind that on an EX Series switch that uses Junos OS with support for ELS, if the timer value set on an interface level is different from the value set on a switch level, then the value on the interface level takes precedence.

Step-by-Step Procedure

To configure access Switch A for MVRP:

1. Configure the finance VLAN:

```
[edit]
user@Access-Switch-A# set vlans finance vlan-id 100
```

2. Configure the lab VLAN:

```
[edit]
user@Access-Switch-A# set vlans lab vlan-id 200
```

3. Configure the sales VLAN:

```
[edit]
user@Access-Switch-A# set vlans sales vlan-id 300
```

4. Configure an Ethernet interface as a member of the finance VLAN:

```
[edit]
user@Access-Switch-A# set interfaces ge-0/0/1 unit 0 family ethernet-switching vlan members
finance
```

5. Configure an Ethernet interface as a member of the lab VLAN:

```
[edit]
user@Access-Switch-A# set interfaces ge-0/0/2 unit 0 family ethernet-switching vlan members
lab
```

6. Configure an Ethernet interface as a member of the sales VLAN:

```
[edit]
user@Access-Switch-A# set interfaces ge-0/0/3 unit 0 family ethernet-switching vlan members
sales
```

7. Configure a trunk interface:

```
[edit]
user@Access-Switch-A# set interfaces xe-0/1/1 unit 0 family ethernet-switching interface-mode
trunk
```

8. Enable MVRP on the trunk interface:

```
[edit]
user@Access-Switch-A# set protocols mvrp interface xe-0/1/1
```

Results

Check the results of the configuration on Switch A:

```
[edit]

user@Access-Switch-A# show
interfaces {
    ge-0/0/1 {
        unit 0 {
            family ethernet-switching {
                vlan {
                    members finance;
                }
            }
        }
    }
    ge-0/0/2 {
        unit 0 {
            family ethernet-switching {
                vlan {
                    members lab;
                }
            }
        }
    }
    ge-0/0/3 {
        unit 0 {
            family ethernet-switching {
                vlan {
                    members sales;
                }
            }
        }
```

```
    }
    xe-0/1/1 {
        unit 0 {
            family ethernet-switching {
                interface-mode trunk;
            }
        }
    }
}
protocols {
    mvrp {
        interface xe-0/1/1;
    }
}
vlans {
    finance {
        vlan-id 100;
    }
    lab {
        vlan-id 200;
    }
    sales {
        vlan-id 300;
    }
}
```

Configuring VLANs and MVRP on Access Switch B

IN THIS SECTION

- Procedure | 868

To configure three VLANs on the switch, bind access interfaces for PC4 and PC5 to the VLANs, and enable MVRP on the trunk interface of access Switch B, perform these tasks:

Procedure

CLI Quick Configuration

To quickly configure Access Switch B for MVRP, copy the following commands and paste them into the switch terminal window of Switch B:

```
[edit]
set vlans finance vlan-id 100
 set vlans lab vlan-id 200
 set vlans sales vlan-id 300
set interfaces ge-0/0/0 unit 0 family ethernet-switching vlan members finance
set interfaces ge-0/0/1 unit 0 family ethernet-switching vlan members lab
set interfaces xe-0/1/0 unit 0 family ethernet-switching interface-mode trunk
                        set protocols mvrp interface xe-0/1/0
```

Step-by-Step Procedure

To configure access Switch B for MVRP:

1. Configure the finance VLAN:

```
[edit]
user@Access-Switch-B# set vlans finance vlan-id 100
```

2. Configure the lab VLAN:

```
[edit]
user@Access-Switch-B# set vlans lab vlan-id 200
```

3. Configure the sales VLAN:

```
[edit]
user@Access-Switch-B# set vlans sales vlan-id 300
```

4. Configure an Ethernet interface as a member of the finance VLAN:

```
[edit]
user@Access-Switch-B# set interfaces ge-0/0/0 unit 0 family ethernet-switching vlan members
finance
```

5. Configure an Ethernet interface as a member of the lab VLAN:

```
[edit]
user@Access-Switch-B# set interfaces ge-0/0/1 unit 0 family ethernet-switching vlan members
lab
```

6. Configure a trunk interface:

```
user@Access-Switch-B# set interfaces xe-0/1/0 unit 0 family ethernet-switching interface-mode
trunk
```

7. Enable MVRP on the trunk interface:

```
[edit]
user@Access-Switch-B# set protocols mvrp xe-0/1/0
```

NOTE: This example uses default MVRP timers. The default values associated with each MVRP timer are: 200 ms for the join timer, 1000 ms for the leave timer, and 10000 ms (10 seconds) for the leaveall timer. We recommend retaining the use of default timer values as modifying timers to inappropriate values might cause an imbalance in the operation of MVRP. However, if you choose to change the default values, keep in mind that on an EX Series switch that uses Junos OS with support for ELS, if the timer value set on an interface level is different from the value set on a switch level, then the value on the interface level takes precedence.

Results

Check the results of the configuration for Switch B:

```
[edit]

user@Access-Switch-B# show
interfaces {
    ge-0/0/0 {
        unit 0 {
            family ethernet-switching {
                vlan {
                    members finance;
                }
            }
        }
    }
    ge-0/0/1 {
        unit 0 {
            family ethernet-switching {
                vlan {
                    members lab;
                }
            }
        }
    }
    xe-0/1/0 {
        unit 0 {
            family ethernet-switching {
                interface-mode trunk;
            }
        }
    }
}
```

```
protocols {
    mvrp {
        interface xe-0/1/0;
    }
}
vlans {
```

```
    finance {
        vlan-id 100;
    }
    lab {
        vlan-id 200;
    }
    sales {
        vlan-id 300;
    }
}
```

Configuring VLANS and MVRP on Distribution Switch C

IN THIS SECTION

- Procedure | 871

Procedure

CLI Quick Configuration

To quickly configure distribution Switch C for MVRP, copy the following commands and paste them into the switch terminal window of distribution Switch C:

```
[edit]
set interfaces xe-0/1/1 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/1/0 unit 0 family ethernet-switching interface-mode trunk

 set protocols mvrp interface xe-0/1/1
     set protocols mvrp interface xe-0/1/0
```

Step-by-Step Procedure

To configure distribution Switch C for MVRP:

1. Configure the trunk interface to access Switch A:

```
[edit]
user@Distribution-Switch-C# set interfaces xe-0/1/1 unit 0 family ethernet-switching
interface-mode trunk
```

2. Configure the trunk interface to access Switch B:

```
[edit]
user@Distribution-Switch-C# set interfaces xe-0/1/0 unit 0 family ethernet-switching
interface-mode trunk
```

3. Enable MVRP on the trunk interface for **xe-0/1/1** :

```
[edit]
user@Distribution-Switch-C# set protocols mvrp interface xe-0/1/1
```

4. Enable MVRP on the trunk interface for **xe-0/1/0** :

```
[edit]
user@Distribution-Switch-C# set protocols mvrp interface xe-0/1/0
```

Results

Check the results of the configuration for Switch C:

```
[edit]

user@Distribution Switch-C# show
interfaces {
    xe-0/1/0 {
        unit 0 {
            family ethernet-switching {
                interface-mode trunk;
```

```
            }
        }
    }
    xe-0/1/1 {
        unit 0 {
            family ethernet-switching {
                interface-mode trunk;
            }
        }
    }
}
protocols {
    mvrp {
        interface xe-0/1/0;
        interface xe-0/1/1;
    }
```

Verification

IN THIS SECTION

To confirm that the configuration is updating VLAN membership, perform these tasks:

Verifying That MVRP Is Enabled on Access Switch A

Purpose

Verify that MVRP is enabled on the switch.

Action

Show the MVRP configuration:

```
user@Access-Switch-A> show mvrp
MVRP configuration for routing instance 'default-switch'
MVRP dynamic VLAN creation : Enabled
MVRP BPDU MAC address      : Customer bridge group (01-80-C2-00-00-21)
MVRP timers (ms)
  Interface       Join   Leave  LeaveAll
  xe-0/1/1         200    1000     10000
```

Meaning

The results show that MVRP is enabled on the trunk interface of Switch A and that the default timers are used.

Verifying That MVRP Is Updating VLAN Membership on Access Switch A

Purpose

Verify that MVRP is updating VLAN membership by displaying the Ethernet switching interfaces and associated VLANs that are active on Switch A.

Action

List Ethernet switching interfaces on the switch:

```
user@Access-Switch-A> show ethernet-switching interface
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
ge-0/0/1.0                    65535                                 tagged
             finance    100
                               65535    Forwarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
```

```
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
ge-0/0/2.0                    65535                                 tagged
             lab        200
                               65535    Forwarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
ge-0/0/3.0                    65535                                   tagged
             sales      300
                               65535    Forwarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/1/1.0                    65535                                   tagged
             finance    100
                               65535    Forwarding
             lab        200
                               65535    Forwarding
```

Meaning

MVRP has automatically added **finance** and **lab** as VLAN members on the trunk interface because they are being advertised by access Switch B.

Verifying That MVRP Is Enabled on Access Switch B

Purpose

Verify that MVRP is enabled on the switch.

Action

Show the MVRP configuration:

```
user@Access-Switch-B> show mvrp
MVRP configuration for routing instance 'default-switch'
```

```
MVRP dynamic VLAN creation : Enabled
MVRP BPDU MAC address      : Customer bridge group (01-80-C2-00-00-21)
MVRP timers (ms)
  Interface       Join   Leave  LeaveAll
  xe-0/1/0         200    1000     10000
```

Meaning

The results show that MVRP is enabled on the trunk interface of Switch B and that the default timers are used.

Verifying That MVRP Is Updating VLAN Membership on Access Switch B

Purpose

Verify that MVRP is updating VLAN membership by displaying the Ethernet switching interfaces and associated VLANs that are active on Switch B.

Action

List Ethernet switching interfaces on the switch:

```
user@Access-Switch-B> show ethernet-switching interface
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
ge-0/0/0.0                    65535                                 tagged
             finance    100
                               65535    Forwarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
ge-0/0/1.0                    65535                                 tagged
             lab        200
                               65535    Forwarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
```

```
                        LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/1/0.0                    65535                                  tagged
             finance    100
                              65535    Forwarding
             lab        200
                              65535    Forwarding
             sales      300
                              65535    Forwarding
```

Meaning

MVRP has automatically added **finance**, **lab**, and **sales** as VLAN members on the trunk interface because they are being advertised by access Switch A.

Verifying That MVRP Is Enabled on Distribution Switch C

Purpose

Verify that MVRP is enabled on the switch.

Action

Show the MVRP configuration:

```
user@Distribution-Switch-C> show mvrp
MVRP configuration for routing instance 'default-switch'
MVRP dynamic VLAN creation : Enabled
MVRP BPDU MAC address      : Customer bridge group (01-80-C2-00-00-21)
MVRP timers (ms)
  Interface       Join   Leave  LeaveAll
  xe-0/1/1         200    1000     10000
  xe-0/1/0         200    1000     10000
```

Meaning

The results show that MVRP is enabled on the trunk interfaces of Switch C and that the default timers are used.

Verifying That MVRP Is Updating VLAN Membership on Distribution Switch C

Purpose

Verify that MVRP is updating VLAN membership on distribution Switch C by displaying the Ethernet switching interfaces and associated VLANs on distribution Switch C.

Action

List the Ethernet switching interfaces on the switch:

```
user@Distribution-Switch-C> show ethernet-switching interface
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/1/1.0                    65535                                 tagged
             mvrp_100
                               65535    Forwarding
             mvrp_200
                               65535    Forwarding
             mvrp_300
                               65535    Forwarding
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down )
Logical      Vlan       TAG   MAC      STP          Logical         Tagging
interface    members          limit    state        interface flags
xe-0/1/0.0                    65535                                   tagged
             mvrp_100
                               65535    Forwarding
             mvrp_200
                               65535    Forwarding
```

List the VLANs that were created dynamically using MVRP on the switch:

```
user@Distribution-Switch-C> show mvrp dynamic-vlan-memberships


MVRP dynamic vlans for routing instance 'default-switch'
```

```
 (s) static vlan, (f) fixed registration

VLAN ID          Interfaces
100              xe-0/1/1.0
                 xe-0/1/0.0
200              xe-0/1/1.0
                 xe-0/1/0.0
300              xe-0/1/1.0
```

Note that this scenario does not have any fixed registration, which is typical when MVRP is enabled.

Meaning

Distribution Switch C has two trunk interfaces. Interface `xe-0/1/1.0` connects Distribution Switch C to Access Switch A and is, therefore, updated to show that it is a member of all the VLANs that are active on Switch A. Any traffic for those VLANs will be passed on from Switch C to Switch A, through interface `xe-0/1/1.0`. Interface `xe-0/1/0.0` connects Switch C to Switch B and is updated to show that it is a member of the two VLANs that are active on Switch B. Thus, Switch C sends traffic for **finance** and **lab** to both Switch A and Switch B. But Switch C sends traffic for **sales** only to Switch A.

Switch C also has three dynamic VLANs created using MVRP: `mvrp_100`, `mvrp_200`, and `mvrp_300`. The dynamically created VLANs `mvrp_100` and `mvrp_200` are active on interfaces `xe-0/1/1.0` and `xe-0/1/0.0`, and dynamically created VLAN `mvrp_300` is active on interface `xe-0/1/1.0`.

Example: Configuring Automatic VLAN Administration Using MVRP on EX Series Switches

IN THIS SECTION

NOTE: This example uses Junos OS for EX Series switches that does not support the Enhanced Layer 2 Software (ELS) configuration style. If your switch runs software that supports ELS, see "Example: Configuring Automatic VLAN Administration Using MVRP on EX Series Switches with ELS Support" on page 859. For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15.

As a network expands and the number of clients and VLANs increases, VLAN administration becomes complex and the task of efficiently configuring VLANs on multiple EX Series switches becomes increasingly difficult. To automate VLAN administration, you can enable Multiple VLAN Registration Protocol (MVRP) on the network.

MVRP also dynamically creates VLANs, further simplifying the network overhead required to statically configure VLANs.

NOTE: Only trunk interfaces can be enabled for MVRP.

This example describes how to use MVRP to automate administration of VLAN membership changes within your network and how to use MVRP to dynamically create VLANs:

Requirements

This example uses the following hardware and software components:

- Two EX Series access switches
- One EX Series distribution switch
- Junos OS Release 10.0 or later for EX Series switches

Overview and Topology

IN THIS SECTION

MVRP is used to manage dynamic VLAN registration in a LAN. It can also be used to dynamically create VLANs.

This example uses MVRP to dynamically create VLANs on the switching network. You can disable dynamic VLAN creation and create VLANs statically, if desired. Enabling MVRP on the trunk interface of

each switch in your switching network ensures that the active VLAN information for the switches in the network is propagated to each switch through the trunk interfaces, assuming dynamic VLAN creation is enabled for MVRP.

MVRP ensures that the VLAN membership information on the trunk interface is updated as the switch's access interfaces become active or inactive in the configured VLANs in a static or dynamic VLAN creation setup.

You do not need to explicitly bind a VLAN to the trunk interface. When MVRP is enabled, the trunk interface advertises all the VLANs that are active (bound to access interfaces) on that switch. An MVRP-enabled trunk interface does not advertise VLANs that have been configured on the switch but that are not currently bound to an access interface. Thus, MVRP provides the benefit of reducing network overhead—by limiting the scope of broadcast, unknown unicast, and multicast (BUM) traffic to interested devices only.

When VLAN access interfaces become active or inactive, MVRP ensures that the updated information is advertised on the trunk interface. Thus, in this example, distribution Switch C does not forward traffic to inactive VLANs.

NOTE: This example shows a network with three VLANs: **finance**, **sales**, and **lab**. All three VLANs are running the same version of Junos OS. If switches in this network were running a mix of Junos OS releases that included Release 11.3, additional configuration would be necessary—see "Configuring Multiple VLAN Registration Protocol (MVRP) on Switches " on page 839 for details.

Access Switch A has been configured to support all three VLANS and all three VLANS are active, bound to interfaces that are connected to personal computers:

- **ge-0/0/1**—Connects PC1 as a member of **finance**, VLAN ID 100
- **ge-0/0/2**—Connects PC2 as a member of **lab**, VLAN ID 200
- **ge-0/0/3**—Connects PC3 as a member of **sales**, VLAN ID 300

Access Switch B has also been configured to support three VLANS. However, currently only two VLANs are active, bound to interfaces that are connected to personal computers:

- **ge-0/0/0**—Connects PC4 as a member of **finance**, VLAN ID 100
- **ge-0/0/1**—Connects PC5 as a member of **lab**, VLAN ID 200

Distribution Switch C learns the VLANs dynamically using MVRP through the connection to the access switches. Distribution Switch C has two trunk interfaces:

- **xe-0/1/1**—Connects the switch to access Switch A.

- **xe-0/1/0**—Connects the switch to access Switch B.

Topology

Figure 44 on page 882 shows MVRP configured on two access switches and one distribution switch.

Figure 44: MVRP Configured on Two Access Switches and One Distribution Switch for Automatic VLAN Administration

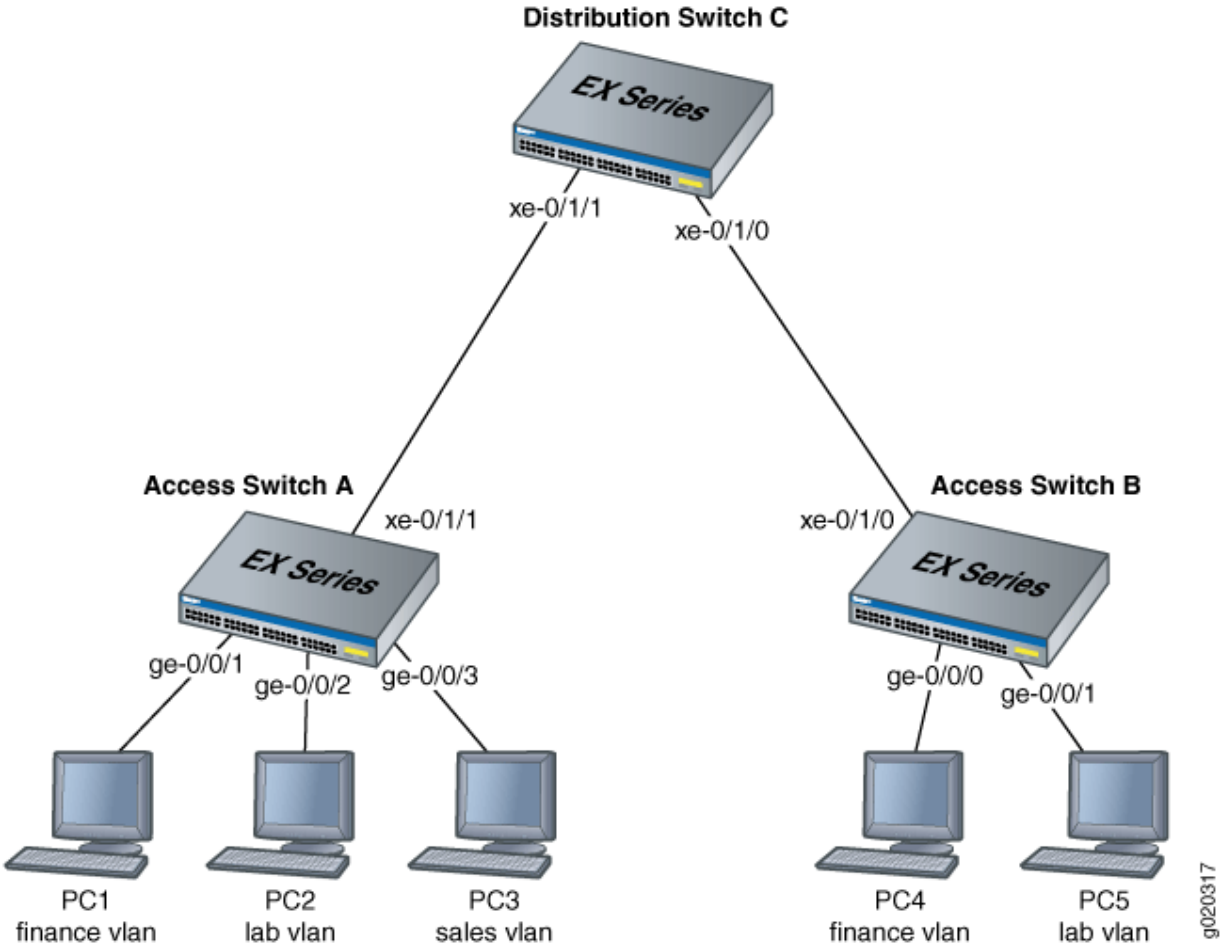


Table 121 on page 883 explains the components of the example topology.

Table 121: Components of the Network Topology

Settings	Settings
Switch hardware	• Access Switch A • Access Switch B • Distribution Switch C
VLAN names and tag IDs	**finance**, tag **100** **lab**, tag **200** **sales**, tag **300**
Interfaces	Access Switch A interfaces: • **ge-0/0/1**—Connects PC1 to access Switch A. • **ge-0/0/2**—Connects PC2 to access Switch A. • **ge-0/0/3**—Connects PC3 to access Switch A. • **xe-0/1/1**—Connects access Switch A to distribution Switch C (trunk). Access Switch B interfaces: • **ge-0/0/0**—Connects PC4 to access Switch B. • **ge-0/0/1**—Connects PC5 to access Switch B. • **xe-0/1/0**—Connects access Switch B to distribution Switch C. (trunk) Distribution Switch C interfaces: • **xe-0/1/1**—Connects distribution Switch C to access Switch A. (trunk) • **xe-0/1/0**—Connects distribution Switch C to access Switch B. (trunk)

Configuring VLANs and MVRP on Access Switch A

IN THIS SECTION

To configure VLANs on the switch, bind access interfaces to the VLANs, and enable MVRP on the trunk interface of access Switch A, perform these tasks:

Procedure

CLI Quick Configuration

To quickly configure access Switch A for MVRP, copy the following commands and paste them into the switch terminal window of Switch A:

```
[edit]
set vlans finance vlan-id 100
 set vlans lab vlan-id 200
 set vlans sales vlan-id 300
set interfaces ge-0/0/1 unit 0 family ethernet-switching vlan members finance
set interfaces ge-0/0/2 unit 0 family ethernet-switching vlan members lab
set interfaces ge-0/0/3 unit 0 family ethernet-switching vlan members sales
set interfaces xe-0/1/1 unit 0 family ethernet-switching port-mode trunk
                        set protocols mvrp interface xe-0/1/1.0
```

NOTE: As recommended as a best practice, default MVRP timers are used in this example. The default values associated with each MVRP timer are: 200 ms for the join timer, 1000 ms for the leave timer, and 10000 ms for the leaveall timer. Modifying timers to inappropriate values might cause an imbalance in the operation of MVRP.

Step-by-Step Procedure

To configure access Switch A for MVRP:

1. Configure the finance VLAN:

```
[edit]
user@Access-Switch-A# set vlans finance vlan-id 100
```

2. Configure the lab VLAN:

```
[edit]
user@Access-Switch-A# set vlans lab vlan-id 200
```

3. Configure the sales VLAN:

```
[edit]
user@Access-Switch-A# set vlans sales vlan-id 300
```

4. Configure an Ethernet interface as a member of the finance VLAN:

```
[edit]
user@Access-Switch-A# set interfaces ge-0/0/1 unit 0 family ethernet-switching vlan members
finance
```

5. Configure an Ethernet interface as a member of the lab VLAN:

```
[edit]
user@Access-Switch-A# set interfaces ge-0/0/2 unit 0 family ethernet-switching vlan members
lab
```

6. Configure an Ethernet interface as a member of the sales VLAN:

```
[edit]
user@Access-Switch-A# set interfaces ge-0/0/3 unit 0 family ethernet-switching vlan members
sales
```

7. Configure a trunk interface:

```
[edit]
user@Access-Switch-A# set interfaces xe-0/1/1 unit 0 family ethernet-switching port-mode
trunk
```

8. Enable MVRP on the trunk interface:

```
[edit]
user@Access-Switch-A# set protocols mvrp interface xe-0/1/1.0
```

Results

Check the results of the configuration on Switch A:

```
[edit]

user@Access-Switch-A# show
interfaces {
    ge-0/0/1 {
        unit 0 {
            family ethernet-switching {
                vlan {
                    members finance;
                }
            }
        }
    }
    ge-0/0/2 {
        unit 0 {
            family ethernet-switching {
                vlan {
                    members lab;
                }
            }
        }
    }
    ge-0/0/3 {
```

```
        unit 0 {
            family ethernet-switching {
                members sales;
                }
            }
        }
    }
    xe-0/1/1 {
        unit 0 {
            family ethernet-switching {
                port-mode trunk;
            }
        }
    }
}
protocols {
    mvrp {
        interface xe-0/1/1.0;
    }
}
vlans {
    finance {
        vlan-id 100;
    }
    lab {
        vlan-id 200;
    }
    sales {
        vlan-id 300;
    }
}
```

Configuring VLANs and MVRP on Access Switch B

IN THIS SECTION

- Procedure | 888

To configure three VLANs on the switch, bind access interfaces for PC4 and PC5 to the VLANs, and enable MVRP on the trunk interface of access Switch B, perform these tasks:

Procedure

CLI Quick Configuration

To quickly configure Access Switch B for MVRP, copy the following commands and paste them into the switch terminal window of Switch B:

```
[edit]
set vlans finance vlan-id 100
 set vlans lab vlan-id 200
 set vlans sales vlan-id 300
set interfaces ge-0/0/0 unit 0 family ethernet-switching vlan members finance
set interfaces ge-0/0/1 unit 0 family ethernet-switching vlan members lab
set interfaces xe-0/1/0 unit 0 family ethernet-switching port-mode trunk
                        set protocols mvrp interface xe-0/1/0.0
```

Step-by-Step Procedure

To configure access Switch B for MVRP:

1. Configure the finance VLAN:

```
[edit]
user@Access-Switch-B# set vlans finance vlan-id 100
```

2. Configure the lab VLAN:

```
[edit]
user@Access-Switch-B# set vlans lab vlan-id 200
```

3. Configure the sales VLAN:

```
[edit]
user@Access-Switch-B# set vlans sales vlan-id 300
```

4. Configure an Ethernet interface as a member of the finance VLAN:

```
[edit]
user@Access-Switch-B# set interfaces ge-0/0/0 unit 0 family ethernet-switching vlan members
finance
```

5. Configure an Ethernet interface as a member of the lab VLAN:

```
[edit]
user@Access-Switch-B# set interfaces ge-0/0/1 unit 0 family ethernet-switching vlan members
lab
```

6. Configure a trunk interface:

```
user@Access-Switch-B# set interfaces xe-0/1/0 unit 0 family ethernet-switching port-mode trunk
```

7. Enable MVRP on the trunk interface:

```
[edit]
user@Access-Switch-B# set protocols mvrp xe-0/1/0.0
```

NOTE: As we recommend as a best practice, default MVRP timers are used in this example. The default values associated with each MVRP timer are: 200 ms for the join timer, 1000 ms for the leave timer, and 10000 ms for the leaveall timer. Modifying timers to inappropriate values might cause an imbalance in the operation of MVRP.

Results

Check the results of the configuration for Switch B:

```
[edit]

user@Access-Switch-B# show
interfaces {
```

```
    ge-0/0/0 {
        unit 0 {
            family ethernet-switching {
                vlan {
                    members finance;
                }
            }
        }
    }
    ge-0/0/1 {
        unit 0 {
            family ethernet-switching {
                vlan {
                    members lab;
                }
            }
        }
    }
    xe-0/1/0 {
        unit 0 {
            family ethernet-switching {
                port-mode trunk;
            }
        }
    }
}
```

```
protocols {
    mvrp {
        interface xe-0/1/0.0;
    }
}
vlans {
    finance {
        vlan-id 100;
    }
    lab {
        vlan-id 200;
    }
    sales {
    vlan-id 300;
```

```
    }
}
```

Configuring VLANS and MVRP on Distribution Switch C

IN THIS SECTION

- Procedure | 891

Procedure

CLI Quick Configuration

To quickly configure distribution Switch C for MVRP, copy the following commands and paste them into the switch terminal window of distribution Switch C:

```
[edit]
set interfaces xe-0/1/1 unit 0 family ethernet-switching port-mode trunk
set interfaces xe-0/1/0 unit 0 family ethernet-switching port-mode trunk

 set protocols mvrp interface xe-0/1/1.0
     set protocols mvrp interface xe-0/1/0.0
```

Step-by-Step Procedure

To configure distribution Switch C for MVRP:

1. Configure the trunk interface to access Switch A:

```
[edit]
user@Distribution-Switch-C# set interfaces xe-0/1/1 unit 0 family ethernet-switching port-
mode trunk
```

2. Configure the trunk interface to access Switch B:

```
[edit]
user@Distribution-Switch-C# set interfaces xe-0/1/0 unit 0 family ethernet-switching port-
mode trunk
```

3. Enable MVRP on the trunk interface for **xe-0/1/1** :

```
[edit]
user@Distribution-Switch-C# set protocols mvrp interface xe-0/1/1.0
```

4. Enable MVRP on the trunk interface for **xe-0/1/0** :

```
[edit]
user@Distribution-Switch-C# set protocols mvrp interface xe-0/1/0.0
```

Results

Check the results of the configuration for Switch C:

```
[edit]

user@Distribution Switch-C# show
interfaces {
    xe-0/1/0 {
        unit 0 {
            family ethernet-switching {
                port-mode trunk;
            }
        }
    }
    xe-0/1/1 {
        unit 0 {
            family ethernet-switching {
                port-mode trunk;
```

```
            }
        }
    }
}
protocols {
    mvrp {
        interface xe-0/1/0.0;
        interface xe-0/1/1.0;
    }
```

Verification

IN THIS SECTION

To confirm that the configuration is updating VLAN membership, perform these tasks:

Verifying That MVRP Is Enabled on Access Switch A

Purpose

Verify that MVRP is enabled on the switch.

Action

Show the MVRP configuration:

```
user@Access-Switch-A> show mvrp
MVRP configuration
MVRP status                 : Enabled
MVRP dynamic VLAN creation  : Enabled
```

```
MVRP timers (ms):
Interface            Join   Leave       LeaveAll
--------------       -----  --------    -----------
all                   200   1000        10000
xe-0/1/1.0            200   1000        10000

Interface       Status        Registration Mode
--------------  --------      -----------------
all             Disabled      Normal
xe-0/1/1.0      Enabled       Normal
```

Meaning

The results show that MVRP is enabled on the trunk interface of Switch A and that the default timers are used.

Verifying That MVRP Is Updating VLAN Membership on Access Switch A

Purpose

Verify that MVRP is updating VLAN membership by displaying the Ethernet switching interfaces and associated VLANs that are active on Switch A.

Action

List Ethernet switching interfaces on the switch:

```
user@Access-Switch-A> show ethernet-switching interfaces

Interface   State    VLAN members   Tag  Tagging       Blocking
ge-0/0/1.0  up       finance        100  untagged      unblocked
ge-0/0/2.0  up       lab            200  untagged      unblocked
ge-0/0/3.0  up       sales          300  untagged      unblocked
xe-0/1/1.0  up       finance        100  untagged      unblocked
                     lab            200  untagged      unblocked
```

Meaning

MVRP has automatically added **finance** and **lab** as VLAN members on the trunk interface because they are being advertised by access Switch B.

Verifying That MVRP Is Enabled on Access Switch B

Purpose

Verify that MVRP is enabled on the switch.

Action

Show the MVRP configuration:

```
user@Access-Switch-B> show mvrp

MVRP configuration
MVRP status                 : Enabled
MVRP dynamic VLAN creation  : Enabled

MVRP timers (ms):
Interface         Join   Leave       LeaveAll
--------------    -----  --------    -----------
all                200   1000       10000
xe-0/1/0.0         200   1000       10000

Interface       Status        Registration Mode
--------------  --------      -----------------
all             Disabled      Normal
xe-0/1/0.0      Enabled       Normal
```

Meaning

The results show that MVRP is enabled on the trunk interface of Switch B and that the default timers are used.

Verifying That MVRP Is Updating VLAN Membership on Access Switch B

Purpose

Verify that MVRP is updating VLAN membership by displaying the Ethernet switching interfaces and associated VLANs that are active on Switch B.

Action

List Ethernet switching interfaces on the switch:

```
user@Access-Switch-B> show ethernet-switching interfaces

Interface   State    VLAN members   Tag  Tagging       Blocking
ge-0/0/0.0  up       finance        100  untagged      unblocked
ge-0/0/1.0  up       lab            200  untagged      unblocked
xe-0/1/1.0  up       finance        100  untagged      unblocked
                     lab            200  untagged      unblocked
                     sales          300  untagged      unblocked
```

Meaning

MVRP has automatically added **finance**, **lab**, and **sales** as VLAN members on the trunk interface because they are being advertised by access Switch A.

Verifying That MVRP Is Enabled on Distribution Switch C

Purpose

Verify that MVRP is enabled on the switch.

Action

Show the MVRP configuration:

```
user@Distribution-Switch-C> show mvrp

MVRP configuration
MVRP status                 : Enabled
MVRP dynamic VLAN creation  : Enabled
```

```
MVRP timers (ms):
Interface            Join   Leave       LeaveAll
--------------       -----  --------    -----------
all                   200   1000        10000
xe-0/0/1.0            200   1000        10000
xe-0/1/1.0            200   1000        10000

Interface       Status        Registration Mode
--------------  --------      -----------------
all             Disabled      Normal
xe-0/0/1.0      Enabled       Normal
xe-0/1/1.0      Enabled       Normal
```

Verifying That MVRP Is Updating VLAN Membership on Distribution Switch C

Purpose

Verify that MVRP is updating VLAN membership on distribution Switch C by displaying the Ethernet switching interfaces and associated VLANs on distribution Switch C.

Action

List the Ethernet switching interfaces on the switch:

```
user@Distribution-Switch-C> show ethernet-switching interfaces

Interface   State    VLAN members   Tag  Tagging       Blocking
xe-0/1/1.0  up       __mvrp_100__                          unblocked
                     __mvrp_200__                          unblocked
                     __mvrp_300__                          unblocked
xe-0/1/0.0  up       __mvrp_100__                          unblocked
                     __mvrp_200__                          unblocked
```

List the VLANs that were created dynamically using MVRP on the switch:

```
user@Distribution-Switch-C> show mvrp dynamic-vlan-memberships


MVRP dynamic vlans for routing instance 'default-switch'
```

```
 (s) static vlan, (f) fixed registration

VLAN ID          Interfaces
100              xe-0/1/1.0
                 xe-0/1/0.0
200              xe-0/1/1.0
                 xe-0/1/0.0
300              xe-0/1/1.0
```

Note that this scenario does not have any fixed registration, which is typical when MVRP is enabled.

Meaning

Distribution Switch C has two trunk interfaces. Interface **xe-0/1/1.0** connects distribution Switch C to Access Switch A and is therefore updated to show that it is a member of all the VLANs that are active on Switch A. Any traffic for those VLANs will be passed on from distribution Switch C to Switch A, through interface **xe-0/1/1.0**. Interface **xe-0/1/0.0** connects distribution Switch C to Switch B and is updated to show that it is a member of the two VLANs that are active on Switch B. Thus, distribution Switch C sends traffic for **finance** and **lab** to both Switch A and Switch B. But distribution Switch C sends traffic for **sales** only to Switch A.

Distribution Switch C also has three dynamic VLANs created using MVRP: **mvrp_100**, **mvrp_200**, and **mvrp_300**. The dynamically created VLANs **mvrp_100** and **mvrp_200** are active on interfaces **xe-0/1/1.0** and **xe-0/1/1.0**, and dynamically created VLAN **mvrp_300** is active on interface **xe-0/1/1.0**.

Verifying That MVRP Is Working Correctly on Switches

IN THIS SECTION

Purpose

After configuring your switch to participate in MVRP, verify that the configuration is properly set and that MVRP messages are being sent and received on your switch.

Action

1. Confirm that MVRP is enabled on your switch.

```
user@switch> show mvrp

Global MVRP configuration
  MVRP status               : Enabled
  MVRP dynamic vlan creation: Enabled
  MVRP Timers (ms):
   Interface        Join   Leave   LeaveAll
   --------------   ----   -----   --------
   all              200    600     10000
   xe-0/1/1.0       200    600     10000

  Interface based configuration:
   Interface        Status     Registration   Dynamic VLAN Creation
   --------------   --------   ------------   --------------------
   all                Disabled   Fixed            Enabled
   xe-0/1/1.0       Enabled    Normal         Enabled
```

2. Confirm that MVRP messages are being sent and received on your switch.

```
user@switch> show mvrp statistics interface xe-0/1/1.0
MVRP statistics
  MRPDU received               : 3342
  Invalid PDU received         : 0
  New received                 : 2
  Join Empty received          : 1116
  Join In received             : 2219
  Empty received               : 2
  In received                  : 2
  Leave received               : 1
  LeaveAll received            : 1117
  MRPDU transmitted            : 3280
  MRPDU transmit failures      : 0
  New transmitted              : 0
  Join Empty transmitted       : 1114
  Join In transmitted          : 2163
  Empty transmitted            : 1
  In transmitted               : 1
```

```
Leave transmitted             : 1
LeaveAll transmitted          : 1111
```

Meaning

The output of `show mvrp` shows that interface `xe-0/1/1.0` is enabled for MVRP participation as shown in the status in the `Interface based configuration` field.

The output for `show mvrp statistics interface xe-0/1/1.0` confirms that MVRP messages are being transmitted and received on the interface.

NOTE: You can identify an MVRP compatibility issue on EX Series switches by looking at the output from this command. If *Join Empty received* and *Join In received* incorrectly display zero, even though the value for *MRPDU received* has been increased, you are probably running different versions of Junos OS, including Release 11.3, on the switches in this network. Another indication that MVRP is having a version problem is that unexpected VLAN activity, such as multiple VLAN creation, takes place on the switch running the earlier release version. To remedy these problems, see "Configuring Multiple VLAN Registration Protocol (MVRP) on Switches " on page 839.

Verifying That MVRP Is Working Correctly on EX Series Switches with ELS Support

IN THIS SECTION

Purpose

NOTE: This task uses Junos OS for EX Series switches with support for the Enhanced Layer 2 Software (ELS) configuration style. If your switch runs software that does not support ELS, see "Verifying That MVRP Is Working Correctly on Switches" on page 898. For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15.

After configuring your EX Series switch to participate in MVRP, verify that the configuration is properly set and that MVRP messages are being sent and received on your switch.

Action

1. Confirm that MVRP is enabled on your switch.

```
user@switch> show mvrp
MVRP configuration for routing instance 'default-switch'
MVRP dynamic VLAN creation : Enabled
MVRP BPDU MAC address      : Customer bridge group (01-80-C2-00-00-21)
MVRP timers (ms)
  Interface       Join   Leave  LeaveAll
  xe-0/1/1         200    1000     10000
```

2. Confirm that MVRP messages are being sent and received on your switch.

```
user@switch> show mvrp statistics
MVRP statistics for routing instance 'default-switch'

Interface name                 : xe-0/1/1
VLAN IDs registered            : 117
Sent MVRP PDUs                 : 118824
Received MVRP PDUs without error: 118848
Received MVRP PDUs with error  : 0
Transmitted Join Empty         : 5229
Transmitted Leave All          : 2
Recieved Join In               : 11884924
Transmitted Join In            : 1835
Transmitted Empty              : 93606408
Transmitted Leave              : 888
Transmitted In                 : 13780024
Transmitted New                : 2692
```

```
Received Leave All              : 118761
Received Leave                  : 97
Received In                     : 3869
Received Empty                  : 828
Received Join Empty             : 2020152
Received New                    : 224
...
```

Meaning

The output of `show mvrp` shows that interface xe-0/1/1 is enabled for MVRP participation.

The output for `show mvrp statistics` confirms that MVRP messages are being transmitted and received on interface xe-0/1/1.

NOTE: You can identify an MVRP compatibility issue by observing the output from this command. If `Received Join Empty` and `Received Join In` incorrectly display zero, even though the value for `Received MVRP PDUs without error` has been increased, you are probably running different versions of Junos OS on the switches in this network. Another indication that MVRP is having a version problem is that unexpected VLAN activity, such as multiple VLAN creation, takes place on the switch running the earlier release version. To remedy these problems, see "Configuring Multiple VLAN Registration Protocol (MVRP) on Switches " on page 839.

Verifying That MVRP Is Working Correctly

IN THIS SECTION

Purpose

After configuring your MX Series router or EX Series switch to participate in Multiple VLAN Registration Protocol (MVRP), verify that the configuration is properly set and that MVRP messages are being sent and received on your switch.

Action

1. Confirm that the router is declaring VLANs.

   Show that MVRP is enabled:

```
user@host> show mvrp
MVRP configuration for routing instance 'default-switch'
MVRP dynamic VLAN creation : Enabled
MVRP BPDU MAC address      : Customer bridge group (01-80-C2-00-00-21)
MVRP timers (ms)
  Interface       Join   Leave  LeaveAll
  ge-11/3/0        200     800     10000
```

   Show the MVRP applicant state:

```
user@host> show mvrp applicant-state
MVRP applicant state for routing instance 'default-switch'
 (VO) Very anxious observer, (VP) Very anxious passive, (VA) Very anxious new,
 (AN) Anxious new, (AA) Anxious active, (QA) Quiet active, (LA) Leaving active,
 (AO) Anxious observer, (QO) Quiet observer, (LO) Leaving observer,
 (AP) Anxious passive, (QP) Quiet passive

VLAN Id     Interface          State
    100     ge-11/3/0          Declaring (QA)
    200     ge-11/3/0          Declaring (QA)
    300     ge-11/3/0          Declaring (QA)
```

2. Confirm that VLANs are registered on interfaces.

   List VLANs in the registered state:

```
user@host> show mvrp registration-state
MVRP registration state for routing instance 'default-switch'
```

```
VLAN Id   Interface   Registrar   Forced      Managed    STP
                      State       State       State      State
    100   ge-11/3/0   Registered  Registered  Normal     Forwarding
    200   ge-11/3/0   Registered  Registered  Normal     Forwarding
    300   ge-11/3/0   Empty       Empty       Normal     Forwarding
```

3. Display a list of VLANs created dynamically.

   List dynamic VLAN membership:

```
user@host> show mvrp dynamic-vlan-memberships
MVRP dynamic vlans for routing instance 'default-switch'
 (s) static vlan, (f) fixed registration

VLAN Id            Interfaces
    100            ge-3/3/0
                   ge-3/0/5
    200            ge-3/3/0
                   ge-3/0/5
```

Meaning

The output of `show mvrp applicant-state` shows that trunk interface **ge-11/3/0** is declaring (sending out) interest in the VLAN IDs **100**, **200**, and **300**, and MVRP is operating properly.

The output of `show mvrp registrant-state` shows the registrar state for VLANs **100** and **200** as **Registered**, indicating that these VLANs are receiving traffic from a customer site. VLAN **300** is in an **Empty** state and is not receiving traffic from a customer site.

The output of the `show mvrp dynamic-vlan-membership` shows that VLANs **100** and **200** are created dynamically (here, on an MX Series router operating as an aggregation switch between MX Series routers operating as edge switches). VLANs created statically are marked with an **(s)** (which is not indicated in this output).

Change History Table

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

Release	Description
15.1X49-D80	Starting in Junos OS Release 15.1X49-D80, Multiple VLAN Registration Protocol (MVRP) to manage dynamic VLAN registration is supported on SRX1500 devices.

28

CHAPTER

Configuring Ethernet Ring Protection Switching

IN THIS CHAPTER

Example: Configuring Ethernet Ring Protection Switching on EX Series Switches

IN THIS SECTION

You can configure Ethernet ring protection switching (ERPS) on connected EX Series or QFX Series switches to prevent fatal loops from disrupting a network. (Platform support depends on the Junos OS release in your installation.) ERPS is similar to spanning-tree protocols, but ERPS is more efficient because it is customized for ring topologies. You must configure at least three switches to form a ring.

This example shows how to configure Ethernet ring protection switching on four switches that are connected to one another on a dedicated link in a ring topology.

NOTE: This task uses Junos OS for EX Series switches without support for the Enhanced Layer 2 Software (ELS) configuration style. However, an ERPS ring can include different types of switches, with or without ELS support. If you are configuring an ERPS ring that also includes QFX Series or EX Series switches running software that supports ELS, see "Example: Configuring Ethernet Ring Protection Switching on QFX Series and EX Series Switches Supporting ELS " on page 927 for equivalent example configuration steps on those switches. For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15.

Requirements

This example uses the following hardware and software components:

- Four connected EX Series switches that will function as nodes in the ring topology.

NOTE: Because Junos uses an ERPV2 state machine for ERPV1 support , operation of ERPS on those two switches deviates from the ERPV1 ITU standard in the following ways:

- Wait to Restore (WTR) configuration values must be 5-12 minutes.
- The Wait To Block Timer (WTB) is always disabled because it is not supported in ERPSv1. Any configuration you make to the WTB setting has no effect. The output from the CLI command 'show protection-group ethernet-ring node-state detail' lists a WTB setting but that setting has no effect.
- During initial state machine initialization, both ERPV1 ring ports move to a discarding state on the non-RPL node.
- During ERPV1 initial state machine initialization, the Automatic Protection Switching (APS) state moves to an idle state on the non-RPL switch

Before you begin, be sure you have:

- Configured two trunk interfaces on each of the four switches. See Table 122 on page 909 for a list of the interface names used in this example.
- Configured the same VLAN (`erp-control-vlan-1`) with ID 100 on all four switches and associated two network interfaces from each of the four switches with the VLAN. See *Configuring VLANs for EX Series Switches*. See Table 122 on page 909 for a list of the interface names used in this example.
- Configured two VLANs (`erp-data-1` and `erp-data-2`) with IDs 101 and 102, respectively, on all four switches and associated both the east and west interfaces on each switch with `erp-data-1` and `erp-data-2`. See Table 122 on page 909 for a list of the interface names used in this example.

NOTE: When devices have a VLAN-ID configured with a name under an interface hierarchy, a commit error occurs. Avoid this by configuring VLAN-IDs using numbers when they are under an interface hierarchy with ERPS configured in the switch.

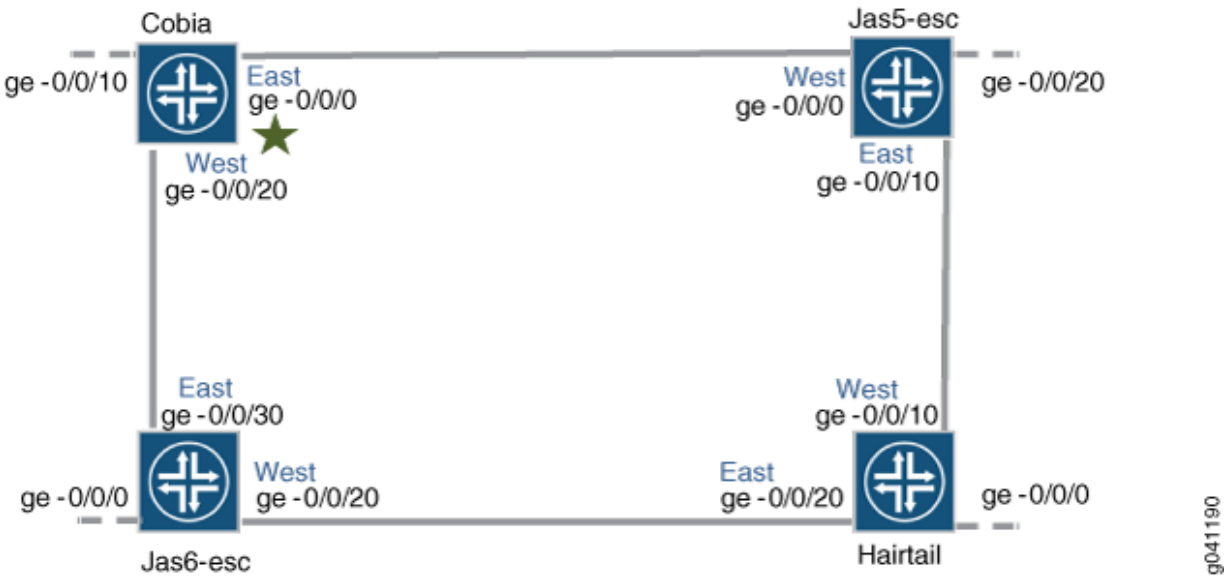
Overview and Topology

ERPS uses a dedicated physical link, including a control VLAN for trunk ports, between all of the switches to protect the active links. ERPS VLANs are all located on this link and are also blocked by default. When traffic between the switches is flowing with no problems, the active links take care of all traffic. Only if an error occurs on one of the data links would the ERPS control channel take over and start forwarding traffic.

NOTE: Trunk ports on switches use a VLAN to create individual control channels for ERPS. When multiple ERPS instances are configured for a ring, there are multiple sets of ring protection links (RPLs) and RPL owners on the ERPS link, and a different channel is blocked for each instance. Nontrunk ports use the physical link as the control channel and protocol data units (PDUs) are untagged, with no VLAN information in the packet.

This example creates one protection ring (called a node ring) named erp1 on four switches connected in a ring by trunk ports as shown in Figure 45 on page 908. Because the links are trunk ports, the VLAN named erp-control-vlan-1 is used for erp1 traffic. The east interface of each switch is connected with the west interface of an adjacent switch. Cobia is the RPL owner, with interface ge-0/0/0 configured as an RPL end interface. The interface ge-0/0/0 of Jas5-esc is configured as the RPL neighbor interface. In the idle state, the RPL end blocks the control VLAN and data channel VLAN for this particular ERP instance—the blocked port on Cobia is marked with a star in Figure 45 on page 908.

Figure 45: Ethernet Ring Protection Switching Example



In this example, we configure the four switches with the interfaces indicated in both Figure 45 on page 908 and Table 122 on page 909.

Table 122: Components to Configure for This Example

Interfaces	Cobia	Jas5-esc	Jas6-esc	Hairtail
East	ge-0/0/0	ge-0/0/10	ge-0/0/30	ge-0/0/20
West	ge-0/0/20	ge-0/0/0	ge-0/0/20	ge-0/0/10
Third	ge-0/0/10	ge-0/0/20	ge-0/0/0	ge-0/0/0

Configuration

IN THIS SECTION

Configuring ERPS on Cobia, the RPL Owner Node

CLI Quick Configuration

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

NOTE: Spanning-tree protocols and ERPS cannot both be configured on a ring port. Because RSTP is the spanning-tree protocol enabled in the default switch configuration,

this example shows disabling RSTP on each ring port before configuring ERPS. If another spanning-tree protocol is enabled, you must disable that first instead.

```
set protocols rstp interface ge-0/0/0 disable
set protocols rstp interface ge-0/0/20 disable
set protocols protection-group ethernet-ring erp1
set protocols protection-group ethernet-ring erp1 ring-protection-link-owner
set protocols protection-group ethernet-ring erp1 data-channel erp-data-1
set protocols protection-group ethernet-ring erp1 data-channel erp-data-2
set protocols protection-group ethernet-ring erp1 control-vlan erp-control-vlan-1
set protocols protection-group ethernet-ring erp1 east-interface control-channel ge-0/0/0.0
set protocols protection-group ethernet-ring erp1 east-interface ring-protection-link-end
set protocols protection-group ethernet-ring erp1 west-interface control-channel ge-0/0/20.0
```

Step-by-Step Procedure

To configure ERPS on Cobia:

1. Disable any spanning- tree protocols configured on the ERPS interfaces. STP, RSTP, VSTP, and MSTP are all available spanning tree protocols. RSTP is enabled in the default configuration, so this example shows disabling RSTP:

   ```
   [edit protocols]
   user@switch# set rstp interface ge-0/0/0 disable
   user@switch# set  rstp interface ge-0/0/20 disable
   ```

2. Create a node ring named erp1:

   ```
   [edit protocols]
   user@switch# set protection-group ethernet-ring erp1
   ```

3. Designate Cobia as the RPL owner node:

   ```
   [edit protocols protection-group ethernet-ring erp1]
   user@switch# set ring-protection-link-owner
   ```

4. Configure the VLANs erp-data-1 and erp-data-2 as data channels:

```
[edit protocols protection-group ethernet-ring erp1]
user@switch# set data-channel erp-data-1
user@switch# set data-channel erp-data-2
```

5. Configure the control VLAN erp-control-vlan-1 for this ERP instance on the trunk interface:

```
[edit protocols protection-group ethernet-ring erp1]
user@switch# set control-vlan erp-control-vlan-1
```

6. Configure the east interface of the node ring erp1 with the control channel ge-0/0/0.0 and indicate that this particular ring protection link ends here:

```
[edit protocols protection-group ethernet-ring erp1]
user@switch#  set east-interface control-channel ge-0/0/0.0
user@switch#  set east-interface ring-protection-link-end
```

7. Configure the west interface of the node ring erp1 with the control channel ge-0/0/20.0:

```
[edit protocols protection-group ethernet-ring erp1]
user@switch#  set west-interface control-channel ge-0/0/20.0
```

Results

In configuration mode, check your ERPS configuration by entering the `show protocols` command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
[edit]
user@switch# show protocols
rstp {
    interface ge-0/0/20.0 {
        disable;
    }
    interface ge-0/0/0.0 {
        disable;
    }
```

```
}
protection-group {
    ethernet-ring erp1 {
        ring-protection-link-owner;
        east-interface {
            control-channel {
                ge-0/0/0.0;
            }
            ring-protection-link-end;
        }
        west-interface {
            control-channel {
                ge-0/0/20.0;
            }
        }
        control-vlan erp-control-vlan-1;
        data-channel {
            vlan [ 101-102 ];
        }
    }
}
```

In configuration mode, check your VLAN configuration by entering the `show vlans` command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
[edit]
user@switch# show vlans
erp-control-vlan-1 {
    vlan-id 100;
    interface {
        ge-0/0/0.0;
        ge-0/0/20.0;
    }
}
erp-data-1 {
    vlan-id 101;
    interface {
        ge-0/0/10.0;
        ge-0/0/0.0;
        ge-0/0/20.0;
    }
```

```
}
erp-data-2 {
    vlan-id 102;
    interface {
        ge-0/0/10.0;
        ge-0/0/0.0;
        ge-0/0/20.0;
    }
}
```

In configuration mode, check your interface configurations by entering the `show interfaces` command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
[edit]
user@switch# show interfaces
ge-0/0/0 {
    unit 0 {
        family ethernet-switching {
            port-mode trunk;
        }
    }
}
ge-0/0/10 {
    unit 0 {
        family ethernet-switching {
            port-mode trunk;
        }
    }
}
ge-0/0/20 {
    unit 0 {
        family ethernet-switching {
             port-mode trunk;
        }
    }
}
```

If you are finished configuring the device, enter `commit` in configuration mode.

Configuring ERPS on Jas5-esc

CLI Quick Configuration

To quickly configure Jas5-esc, 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 rstp interface ge-0/0/10 disable
set protocols rstp interface ge-0/0/0 disable
set protocols protection-group ethernet-ring erp1
set protocols protection-group ethernet-ring erp1 data-channel erp-data-1
set protocols protection-group ethernet-ring erp1 data-channel erp-data-2
set protocols protection-group ethernet-ring erp1 control-vlan erp-control-vlan-1
set protocols protection-group ethernet-ring erp1 east-interface control-channel ge-0/0/10.0
set protocols protection-group ethernet-ring erp1 west-interface control-channel ge-0/0/0.0
```

Step-by-Step Procedure

To configure ERPS on Jas5-esc:

1. Disable any spanning- tree protocols configured on the ERPS interfaces. RSTP is enabled in the default configuration, so this example shows disabling RSTP:

   ```
   [edit protocols]
   user@switch# set rstp interface ge-0/0/10 disable
   user@switch# set rstp interface ge-0/0/0 disable
   ```

2. Create a node ring named erp1:

   ```
   [edit protocols]
   user@switch# set protection-group ethernet-ring erp1
   ```

3. Configure a control VLAN named erp-control-vlan-1 for the node ring erp1:

   ```
   [edit protocols protection-group ethernet-ring erp1]
   user@switch# set control-vlan erp-control-vlan-1
   ```

4. Configure two data channels named erp-data-1 and erp-data-2 to define a set of VLAN IDs that belong to a ring instance.

```
[edit protocols protection-group ethernet-ring erp1]
user@switch# set data-channel erp-data-1
user@switch# set data-channel erp-data-2
```

5. Configure the east interface of the node ring erp1 with the control channel ge-0/0/10.0:

```
[edit protocols protection-group ethernet-ring erp1]
user@switch#  set east-interface control-channel ge-0/0/10.0
```

6. Configure the west interface of the node ring erp1 with the control channel ge-0/0/0.0:

```
[edit protocols protection-group ethernet-ring erp1]
user@switch#  set west-interface control-channel ge-0/0/0.0
```

Results

In configuration mode, check your ERPS configuration by entering the `show protocols` command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
[edit]
user@switch# show protocols
rstp {
    interface ge-0/0/10.0 {
        disable;
    }
    interface ge-0/0/0.0 {
        disable;
    }
}
protection-group {
    ethernet-ring erp1 {
        east-interface {
            control-channel {
                ge-0/0/10.0;
            }
```

```
        }
        west-interface {
            control-channel {
                ge-0/0/0.0;
            }
        }
        control-vlan erp-control-vlan-1;
        data-channel {
            vlan [ 101-102 ];
        }
    }
}
```

In configuration mode, check your VLAN configuration by entering the `show vlans` command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
[edit]
user@switch# show vlans
erp-control-vlan-1 {
    vlan-id 100;
    interface {
        ge-0/0/10.0;
        ge-0/0/0.0;
    }
}
erp-data-1 {
    vlan-id 101;
    interface {
        ge-0/0/20.0;
        ge-0/0/10.0;
        ge-0/0/0.0;
    }
}
erp-data-2 {
    vlan-id 102;
    interface {
        ge-0/0/20.0;
        ge-0/0/10.0;
        ge-0/0/0.0;
```

```
    }
}
```

In configuration mode, check your interface configurations by entering the `show interfaces` command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
[edit]
user@switch# show interfaces
ge-0/0/0 {
    unit 0 {
        family ethernet-switching {
            port-mode trunk;
        }
    }
}
ge-0/0/10 {
    unit 0 {
        family ethernet-switching {
            port-mode trunk;
        }
    }
}
ge-0/0/20 {
    unit 0 {
        family ethernet-switching {
            port-mode trunk;
        }
    }
}
```

If you are finished configuring the device, enter `commit` in configuration mode.

Configuring ERPS on Hairtail

CLI Quick Configuration

To quickly configure Hairtail, 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 rstp interface ge-0/0/10 disable
set protocols rstp interface ge-0/0/20 disable
set protocols protection-group ethernet-ring erp1
set protocols protection-group ethernet-ring erp1 data-channel erp-data-1
set protocols protection-group ethernet-ring erp1 data-channel erp-data-2
set protocols protection-group ethernet-ring erp1 control-vlan erp-control-vlan-1
set protocols protection-group ethernet-ring erp1 east-interface control-channel ge-0/0/20.0
set protocols protection-group ethernet-ring erp1 west-interface control-channel ge-0/0/10.0
```

Step-by-Step Procedure

To configure ERPS on Hairtail:

1. Disable any spanning- tree protocols configured on the ERPS interfaces. RSTP is enabled in the default configuration, so this example shows disabling RSTP:

   ```
   [edit protocols]
   user@switch# set rstp interface ge-0/0/10 disable
   user@switch# set rstp interface ge-0/0/20 disable
   ```

2. Create a node ring named erp1:

   ```
   [edit protocols]
   user@switch# set protection-group ethernet-ring erp1
   ```

3. Configure the control VLAN erp-control-vlan-1 for the node ring erp1:

   ```
   [edit protocols protection-group ethernet-ring erp1]
   user@switch# set control-vlan erp-control-vlan-1
   ```

4. Configure two data channels named erp-data-1 and erp-data-2 to define a set of VLAN IDs that belong to a ring instance:

```
[edit protocols protection-group ethernet-ring erp1]
user@switch# set data-channel erp-data-1
user@switch# set data-channel erp-data-2
```

5. Configure the east interface of the node ring erp1 with the control channel ge-0/0/20.0 and indicate that it connects to a ring protection link:

```
[edit protocols protection-group ethernet-ring erp1]
user@switch#  set east-interface control-channel ge-0/0/20.0
```

6. Configure the west interface of the node ring erp1 with the control channel ge-0/0/10.0 and indicate that it connects to a ring protection link:

```
[edit protocols protection-group ethernet-ring erp1]
user@switch#  set west-interface control-channel ge-0/0/10.0
```

Results

In configuration mode, check your ERPS configuration by entering the `show protocols` command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
[edit]
user@switch# show protocols
rstp {
    interface ge-0/0/10.0 {
        disable;
    }
    interface ge-0/0/20.0 {
        disable;
    }
}
protection-group {
    ethernet-ring erp1 {
        east-interface {
            control-channel {
```

```
                ge-0/0/20.0;
            }
        }
        west-interface {
            control-channel {
            ge-0/0/10.0;
            }
        }
        control-vlan erp-control-vlan-1;
        data-channel {
            vlan [ 101-102 ];
        }
    }
}
```

In configuration mode, check your VLAN configuration by entering the `show vlans` command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
[edit]
user@switch# show vlans
erp-control-vlan-1 {
    vlan-id 100;
    interface {
        ge-0/0/20.0;
        ge-0/0/10.0;
    }
}
erp-data-1 {
    vlan-id 101;
    interface {
        ge-0/0/0.0;
        ge-0/0/20.0;
        ge-0/0/10.0;
    }
}
erp-data-2 {
    vlan-id 102;
    interface {
        ge-0/0/0.0;
        ge-0/0/20.0;
        ge-0/0/10.0;
```

```
    }
}
```

In configuration mode, check your interface configurations by entering the `show interfaces` command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
[edit]
user@switch# show interfaces
ge-0/0/0 {
    unit 0 {
        family ethernet-switching {
            port-mode trunk;
        }
    }
}
ge-0/0/10 {
    unit 0 {
        family ethernet-switching {
            port-mode trunk;
        }
ge-0/0/20 {
    unit 0 {
        family ethernet-switching {
            port-mode trunk;
        }
    }
```

If you are finished configuring the device, enter `commit` in configuration mode.

Configuring ERPS on Jas6-esc

CLI Quick Configuration

To quickly configure Jas6-esc, 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 rstp interface ge-0/0/30 disable
set protocols rstp interface ge-0/0/20 disable
set protocols protection-group ethernet-ring erp1
```

```
set protocols protection-group ethernet-ring erp1 data-channel erp-data-1
set protocols protection-group ethernet-ring erp1 data-channel erp-data-2
set protocols protection-group ethernet-ring erp1 control-vlan erp-control-vlan-1
set protocols protection-group ethernet-ring erp1 east-interface control-channel ge-0/0/30.0
set protocols protection-group ethernet-ring erp1 west-interface control-channel ge-0/0/20.0
```

Step-by-Step Procedure

To configure ERPS on Jas6-esc:

1. Disable any spanning- tree protocols configured on the ERPS interfaces. RSTP is enabled in the default configuration, so this example shows disabling RSTP:

```
[edit protocols]
user@switch# set rstp interface ge-0/0/30 disable
user@switch# set rstp interface ge-0/0/20 disable
```

2. Create a node ring named erp1:

```
[edit protocols]
user@switch# set protection-group ethernet-ring erp1
```

3. Configure the control VLAN erp-control-vlan-1 for the node ring erp1:

```
[edit protocols protection-group ethernet-ring erp1]
user@switch# set control-vlan erp-control-vlan-1
```

4. Configure two data channels named erp-data-1 and erp-data-2 to define a set of VLAN IDs that belong to a ring instance.

```
[edit protocols protection-group ethernet-ring erp1]
user@switch# set data-channel erp-data-1
user@switch# set data-channel erp-data-2
```

5. Configure the east interface of the node ring erp1 with the control channel ge-0/0/30.0 :

```
[edit protocols protection-group ethernet-ring erp1]
user@switch#  set east-interface control-channel ge-0/0/30.0
```

6. Configure the west interface of the node ring erp1 with the control channel ge-0/0/20.0:

```
[edit protocols protection-group ethernet-ring erp1]
user@switch#  set west-interface control-channel ge-0/0/20.0
```

Results

In configuration mode, check your ERPS configuration by entering the `show protocols` command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
[edit]
user@switch# show protocols
rstp {
    interface ge-0/0/20.0 {
        disable;
    }
    interface ge-0/0/30.0 {
        disable;
    }
}
protection-group {
    ethernet-ring erp1 {
        east-interface {
            control-channel {
                ge-0/0/30.0;
            }
        }
        west-interface {
            control-channel {
                ge-0/0/20.0;
            }
        }
        control-vlan erp-control-vlan-1;
        data-channel {
```

```
            vlan [ 101-102 ];
        }
    }
}
```

In configuration mode, check your VLAN configuration by entering the `show vlans` command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
[edit]
user@switch# show vlans
erp-control-vlan-1 {
    vlan-id 100;
    interface {
        ge-0/0/30.0;
        ge-0/0/20.0;
    }
}
erp-data-1 {
    vlan-id 101;
    interface {
        ge-0/0/0.0;
        ge-0/0/30.0;
        ge-0/0/20.0;
    }
}
erp-data-2 {
    vlan-id 102;
    interface {
        ge-0/0/0.0;
        ge-0/0/30.0;
        ge-0/0/20.0;
    }
}
```

In configuration mode, check your interfaces configuration by entering the `show interfaces` command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
[edit]
user@switch# show interfaces
```

```
ge-0/0/0 {
    unit 0 {
        family ethernet-switching {
            port-mode trunk;
        }
    }
}
ge-0/0/20 {
    unit 0 {
        family ethernet-switching {
            port-mode trunk;
        }
    }
}
ge-0/0/30 {
    unit 0 {
        family ethernet-switching {
            port-mode trunk;
       }
    }
}
```

Verification

IN THIS SECTION

Verify that ERPS is working correctly.

Verifying That ERPS Is Working Correctly

Purpose

Verify that ERPS is working on the four EX switches that function as nodes in the ring topology.

Action

Check the state of the ring links in the output of the `show protection-group ethernet-ring interface` command. When the ring is configured but not being used (no error exists on the data links), one ERP interface is forwarding traffic and one is discarding traffic. Discarding blocks the ring.

```
user@switch> show protection-group ethernet-ring interface
Ethernet ring port parameters for protection group erp1
Interface    Forward State  RPL End  Signal Failure    Admin State
ge-0/0/2.0   discarding     yes      clear             ready
ge-0/0/0.0   forwarding     no       clear             ready
```

To find out what has occurred since the last restart, check the RPS statistics for ring-blocked events. `NR` is a No Request ring block, which means that the switch is not blocking either of the two ERP interfaces. `NR-RB` is a No Request Ring Blocked event, which means that the switch is blocking one of its ERP interfaces and sending a packet out to notify the other switches.

```
user@switch> show protection-group ethernet-ring statistics
Ring Name Local SF Remote SF NR Event NR-RB Event
erp1       2          1            2            3
```

Meaning

The **show protection-group ethernet-ring interface** command output from the RPL owner node indicates that one interface is forwarding traffic and one is discarding traffic, meaning that the ERP is ready but not active. If at least one interface in the ring is not forwarding, the ring is blocked and therefore ERP is working.

The **show protection-group ethernet-ring statistics** command output indicates that, since the last reboot, both local and remote signal failures have occurred (`Local SF` and `Remote SF`).

The `NR Event` count is 2, indicating that the NR state was entered into twice. `NR` stands for No Request. This means that the switch either originated NR PDUs or received an NR PDU from another switch and stopped blocking the interface to allow ERP to function.

The three `NR-RB` events indicate that on three occasions, this switch either sent out NR-RB PDUs or received NR-RB PDUs from another switch. This occurs when a network problem is resolved and the switch once again blocks the ERP link at one end.

RELATED DOCUMENTATION

- Configuring Ethernet Ring Protection Switching on Switches (CLI Procedure)
- *Ethernet Ring Protection Switching Overview*
- *Understanding Ethernet Ring Protection Switching Functionality*

Example: Configuring Ethernet Ring Protection Switching on QFX Series and EX Series Switches Supporting ELS

IN THIS SECTION

You can configure Ethernet ring protection switching (ERPS) on connected EX Series or QFX Series switches to prevent fatal loops from disrupting a network. ERPS is similar to the Spanning Tree Protocol, but ERPS is more efficient because it is customized for ring topologies. You must connect and configure at least three switches to form a ring.

This example shows how to configure Ethernet ring protection switching on four switches with ELS support, connected to one another on a dedicated link in a ring topology. You can include different types of switches in an ERPS ring, including those with and without ELS support. If any of your EX Series switches runs software that does not support ELS, use these configuration directions: "Example: Configuring Ethernet Ring Protection Switching on EX Series Switches" on page 906. For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15.

Requirements

This example uses the following hardware and software components:

- Four connected EX Series switches or QFX Series switches that support the Enhanced Layer 2 Software (ELS) to function as nodes in the ring topology. You could use any of these QFX Series switches: QFX5100, QFX5200, and QFX10000. This configuration also applies to EX Series switches that support the Enhanced Layer 2 Software (ELS) configuration style that runs on EX4300, EX4600, EX2300, and EX3400 switches.
- Junos OS Release 13.2X50-D10 or later for EX Series switches.
- Junos OS Release 14.1X53-D10 or later for QFX5100 switches.
- Junos OS Release 15.1X53-D30 or later for QFX5200 and QFX10000 switches.

Before you begin, be sure you have:

- Configured two trunk interfaces on each of the four switches. See Table 123 on page 929 for a list of the interface names used in this example.
- Configured a VLAN (with name `erp-control-vlan-1` and ID `100`) on all four switches and associated two network interfaces from each of the four switches with the VLAN. See Configuring VLANs for the QFX Series OR Configuring VLANs for EX Series Switches with ELS Support (CLI Procedure). See Table 123 on page 929 for a list of the interface names used in this example.
- Configured two more VLANs (one with name `erp-data-1` and vlan ID `101` and a second vlan with the name `erp-data-2` and vlan ID `102`) on all four switches and associated both the east and west interfaces on each switch.

Overview and Topology

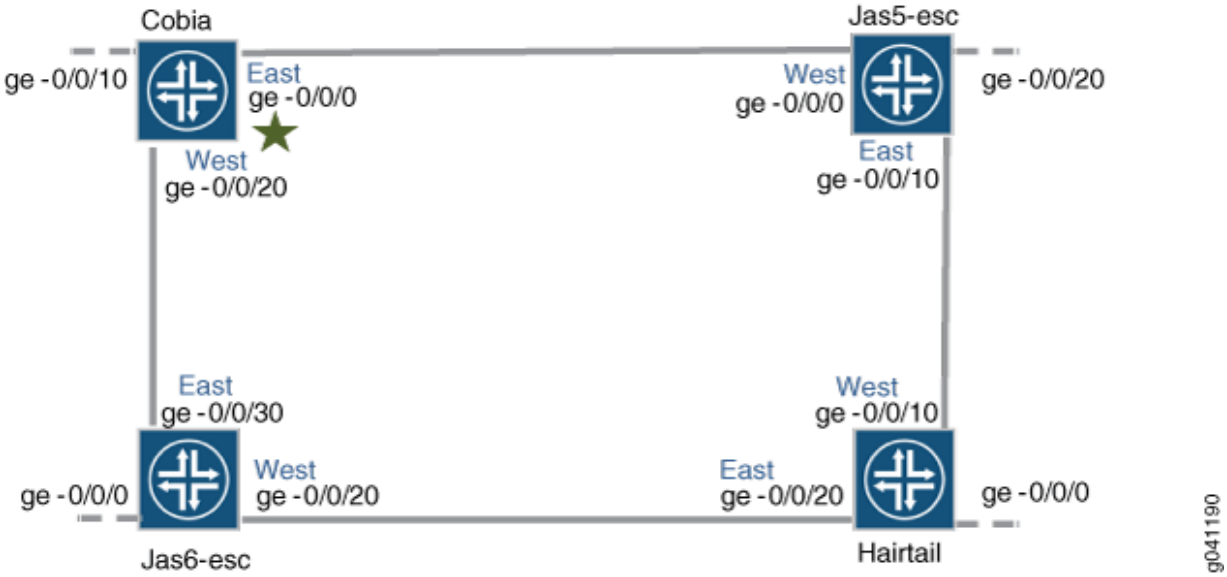
ERPS uses a dedicated physical link, including a control VLAN for trunk ports, between all of the switches to protect the active links. ERPS VLANs are all located on this link and are also blocked by default. When traffic between the switches is flowing with no problems, the active links take care of all traffic. Only if an error occurs on one of the data links would the ERPS control channel take over and start forwarding traffic.

NOTE: Trunk ports on switches use a VLAN to create individual control channels for ERPS. When multiple ERPS instances are configured for a ring, there are multiple sets of ring protection links (RPLs) and RPL owners on the ERPS link, and a different channel is blocked for each instance. Nontrunk ports use the physical link as the control channel and protocol data units (PDUs) are untagged, with no VLAN information in the packet.

This example creates one protection ring (called a node ring) named erp1 on four switches connected in a ring by trunk ports as shown in Figure 46 on page 929. Because the links are trunk ports, VLAN 100

is used for erp1 traffic. The east interface of each switch is connected with the west interface of an adjacent switch. Cobia is the RPL owner, with interface ge-0/0/0 configured as an RPL end interface. The interface ge-0/0/0 of Jas5-esc is configured as the RPL neighbor interface. In the idle state, the RPL end blocks the control VLAN and data channel VLAN for this particular ERP instance—the blocked port on Cobia is marked with a star in Figure 46 on page 929.

Figure 46: Ethernet Ring Protection Switching Example



In this example, we configure the four switches with the interfaces indicated in both Figure 46 on page 929 and Table 123 on page 929.

Table 123: Components to Configure for This Example

Interfaces	Cobia	Jas5-esc	Jas6-esc	Hairtail
East	ge-0/0/0	ge-0/0/10	ge-0/0/30	ge-0/0/20
West	ge-0/0/20	ge-0/0/0	ge-0/0/20	ge-0/0/10
Third	ge-0/0/10	ge-0/0/20	ge-0/0/0	ge-0/0/0

Configuration

IN THIS SECTION

Configuring ERPS on Cobia, the RPL Owner Node

CLI Quick Configuration

To quickly configure Cobia, 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.

ERPS cannot be configured on an interface if any spanning-tree protocol is configured. (RSTP is configured by default.) Therefore, in this example, RSTP is disabled on each ring port before configuring ERPS. Spanning-tree protocols are disabled two different ways, depending on which version of the Junos OS you are running. Therefore, the first two statements in this example vary: Junos release 15.1 or later uses one command to turn off RSTP and Junos releases prior to 15.1 uses another command.

```
Junos OS release 15.1 or later: set protocols rstp interface ge-0/0/0 disable
Junos OS release 15.1 or later: set protocols rstp interface ge-0/0/20 disable
Junos OS release prior to 15.1: delete rstp interface ge-0/0/0
Junos OS release prior to 15.1: delete rstp interface ge-0/0/20
set protocols protection-group ethernet-ring erp1
set protocols protection-group ethernet-ring erp1 ring-protection-link-owner
set protocols protection-group ethernet-ring erp1 data-channel 101
set protocols protection-group ethernet-ring erp1 data-channel 102
set protocols protection-group ethernet-ring erp1 control-vlan 100
set protocols protection-group ethernet-ring erp1 east-interface control-channel ge-0/0/0.0
set protocols protection-group ethernet-ring erp1 east-interface ring-protection-link-end
set protocols protection-group ethernet-ring erp1 west-interface control-channel ge-0/0/20.0
set protocols protection-group ethernet-ring erp1 west-interface control-channel vlan 100
set protocols protection-group ethernet-ring erp1 east-interface control-channel vlan 100
```

Step-by-Step Procedure

To configure ERPS on Cobia:

1. Disable any spanning-tree protocol currently configured on the ERPS interfaces. RSTP, VSTP, and MSTP are all available spanning-tree protocols. RSTP is enabled in the default configuration, so this example shows disabling RSTP. Spanning-tree protocols are disabled two different ways, depending on which version of the Junos OS you are running.

   If you are running Junos release 15.1 or later, disable any spanning-tree protocol with these commands. To disable RSTP:

   ```
   [edit protocols]
   user@switch# set rstp interface ge-0/0/0 disable
   user@switch# set  rstp interface ge-0/0/20 disable
   ```

   If you are running a Junos release prior to 15.1, disable any spanning-tree protocol with these commands. To disable RSTP:

   ```
   [edit protocols]
   user@switch# delete rstp interface ge-0/0/0
   user@switch# delete rstp interface ge-0/0/20
   ```

2. Create a node ring named erp1:

   ```
   [edit protocols]
   user@switch# set protection-group ethernet-ring erp1
   ```

3. Designate Cobia as the RPL owner node:

   ```
   [edit protocols protection-group ethernet-ring erp1]
   user@switch# set ring-protection-link-owner
   ```

4. Configure the VLANs 101 and 102 as data channels:

   ```
   [edit protocols protection-group ethernet-ring erp1]
   user@switch# set data-channel 101
   user@switch# set data-channel 102
   ```

5. Configure the control vlan 100 for this ERPS instance on the trunk interface:

```
[edit protocols protection-group ethernet-ring erp1]
user@switch# set control-vlan 100
```

6. Configure the east interface of the node ring erp1 with control channel ge-0/0/0.0 and indicate that this particular ring protection link ends here:

```
[edit protocols protection-group ethernet-ring erp1]
user@switch# set east-interface control-channel ge-0/0/0.0
user@switch# set east-interface ring-protection-link-end
```

7. Configure the west interface of the node ring erp1 with control channel ge-0/0/20.0:

```
[edit protocols protection-group ethernet-ring erp1]
user@switch# set west-interface control-channel ge-0/0/20.0
```

8. Every ring instance on a trunk port has one control VLAN in which ERP packets traverse. The control VLAN also controls data VLANs, if any are configured. Assign 100 as the control VLAN on both interfaces:

```
[edit protocols protection-group ethernet-ring erp1]
user@switch# set west-interface control-channel vlan 100
user@switch# set east-interface control-channel vlan 100
```

Configuring ERPS on Jas5-esc

CLI Quick Configuration

To quickly configure Jas5-esc, 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.

ERPS cannot be configured on an interface if any spanning tree protocol is configured. (RSTP is configured by default.) Therefore, in this example, RSTP is disabled on each ring port before configuring ERPS. Spanning tree is disabled two different ways, depending on which version of the Junos OS you are

running. Therefore, the first two statements will vary: Junos release 15.1 or later uses one command to turn off RSTP and Junos releases prior to 15.1 uses another command.

```
Junos OS release 15.1 or later: set protocols rstp interface ge-0/0/10 disable
Junos OS release 15.1 or later: set protocols rstp interface ge-0/0/0 disable
Junos OS release prior to 15.1: delete rstp interface ge-0/0/10
Junos OS release prior to 15.1: delete rstp interface ge-0/0/0
set protocols protection-group ethernet-ring erp1
set protocols protection-group ethernet-ring erp1 data-channel 101
set protocols protection-group ethernet-ring erp1 data-channel 102
set protocols protection-group ethernet-ring erp1 control-vlan 100
set protocols protection-group ethernet-ring erp1 east-interface control-channel ge-0/0/10.0
set protocols protection-group ethernet-ring erp1 east-interface control-channel ge-0/0/0.0
set protocols protection-group ethernet-ring erp1 west-interface control-channel ge-0/0/20.0
set protocols protection-group ethernet-ring erp1 west-interface ring-protection-link-end
set protocols protection-group ethernet-ring erp1 west-interface control-channel vlan 100
set protocols protection-group ethernet-ring erp1 east-interface control-channel vlan 100
```

Step-by-Step Procedure

To configure ERPS on Jas5-esc:

1. Disable any spanning-tree protocol currently configured on the ERPS interfaces. RSTP, VSTP, and MSTP are all available spanning-tree protocols. RSTP is enabled in the default configuration, so this example shows disabling RSTP. Spanning-tree protocols are disabled two different ways, depending on which version of the Junos OS you are running.

   If you are running Junos release 15.1 or later, disable any spanning-tree protocol with these commands. To disable RSTP:

   ```
   [edit protocols]
   user@switch# set rstp interface ge-0/0/10 disable
   user@switch# set rstp interface ge-0/0/0 disable
   ```

   If you are running a Junos release prior to 15.1, disable any version of spanning-tree protocol with these commands. To disable RSTP:

   ```
   [edit protocols]
   user@switch# delete rstp interface ge-0/0/10
   user@switch# delete rstp interface ge-0/0/0
   ```

2. Create a node ring named erp1:

```
[edit protocols]
user@switch# set protection-group ethernet-ring erp1
```

3. Configure two data channels named erp-data-1 and erp-data-2 to define a set of VLAN IDs that belong to a ring instance.

```
[edit protocols protection-group ethernet-ring erp1]
user@switch# set data-channel vlan 101
user@switch# set data-channel vlan 102
```

4. Configure a control VLAN with ID 100 for the node ring erp1:

```
[edit protocols protection-group ethernet-ring erp1]
user@switch# set control-vlan 100
```

5. Configure the east interface of the node ring erp1 with the control channel ge-0/0/10.0:

```
[edit protocols protection-group ethernet-ring erp1]
user@switch# set east-interface control-channel ge-0/0/10.0
```

6. Configure the west interface of the node ring erp1 with the control channel ge-0/0/0.0 vlan 100:

```
[edit protocols protection-group ethernet-ring erp1]
user@switch# set west-interface control-channel ge-0/0/0.0
set west-interface ring-protection-link-end
```

7. Every ring instance on a trunk port has one control VLAN in which ERP packets traverse. The control VLAN also controls data VLANs, if any are configured. Assign vlan # 100 as the control VLAN:

```
[edit protocols protection-group ethernet-ring erp1]
user@switch# set west-interface control-channel vlan 100
user@switch# set east-interface control-channel vlan 100
```

Configuring ERPS on Hairtail

CLI Quick Configuration

To quickly configure Hairtail, 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.

ERPS cannot be configured on an interface if any spanning tree protocol is configured. (RSTP is configured by default.) Therefore, in this example, RSTP is disabled on each ring port before configuring ERPS. Spanning tree is disabled two different ways, depending on which version of the Junos OS you are running. Therefore, the first two statements will vary: Junos release 15.1 or later uses one command to turn off RSTP and Junos releases prior to 15.1 uses another command.

```
Junos OS release 15.1 or later: set protocols rstp interface ge-0/0/10 disable
Junos OS release 15.1 or later: set protocols rstp interface ge-0/0/20 disable
Junos OS release prior to 15.1: delete rstp interface ge-0/0/10
Junos OS release prior to 15.1: delete rstp interface ge-0/0/20
set protocols protection-group ethernet-ring erp1
set protocols protection-group ethernet-ring erp1 data-channel 101
set protocols protection-group ethernet-ring erp1 data-channel 102
set protocols protection-group ethernet-ring erp1 control-vlan 100
Set protocols protection-group ethernet-ring erp1 east-interface control-channel ge-0/0/0.0
set protocols protection-group ethernet-ring erp1 east-interface control-channel ge-0/0/20.0
set protocols protection-group ethernet-ring erp1 west-interface control-channel vlan 100
set protocols protection-group ethernet-ring erp1 east-interface control-channel vlan 100
```

Step-by-Step Procedure

To configure ERPS on Hairtail:

1. Disable any spanning-tree protocol currently configured on the ERPS interfaces. RSTP, VSTP, and MSTP are all available spanning-tree protocols. RSTP is enabled in the default configuration, so this example shows disabling RSTP. Spanning-tree protocols are disabled two different ways, depending on which version of the Junos OS you are running.

If you are running Junos release 15.1 or later, disable any spanning-tree protocol with these commands. To disable RSTP:

```
[edit protocols]
user@switch# set rstp interface ge-0/0/10 disable
user@switch# set rstp interface ge-0/0/20 disable
```

If you are running a Junos release prior to 15.1, disable any spanning-tree protocol with these commands. To disable RSTP:

```
[edit protocols]
user@switch# delete rstp interface ge-0/0/10
user@switch# delete rstp interface ge-0/0/20
```

2. Create a node ring named erp1:

```
[edit protocols]
user@switch# set protection-group ethernet-ring erp1
```

3. Configure the control vlan 100 for the node ring erp1:

```
[edit protocols protection-group ethernet-ring erp1]
user@switch# set control-vlan 100
```

4. Configure two data channels numbered 101 and 102 to define a set of VLAN IDs that belong to a ring instance:

```
[edit protocols protection-group ethernet-ring erp1]
user@switch# set data-channel vlan 101
user@switch# set data-channel vlan 102
```

5. Configure the east interface of the node ring erp1 with the control channel ge-0/0/20.0:

```
[edit protocols protection-group ethernet-ring erp1]
user@switch# set east-interface control-channel ge-0/0/20.0
```

6. Configure the west interface of the node ring erp1 with the control channel ge-0/0/10.0:

```
[edit protocols protection-group ethernet-ring erp1]
user@switch# set west-interface control-channel ge-0/0/10.0
```

7. Every ring instance on a trunk port has one control VLAN in which ERP packets traverse. The control VLAN also controls data VLANs, if any are configured. Assign 100 as the control VLAN:

```
[edit protocols protection-group ethernet-ring erp1]
user@switch# set west-interface control-channel vlan 100
user@switch# set east-interface control-channel vlan 100
```

Configuring ERPS on Jas6-esc

CLI Quick Configuration

To quickly configure Jas6-esc, 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.

ERPS cannot be configured on an interface if any spanning tree protocol is configured. (RSTP is configured by default.) Therefore, in this example, RSTP is disabled on each ring port before configuring ERPS. Spanning tree is disabled two different ways, depending on which version of the Junos OS you are running. Therefore, the first two statements will vary: Junos release 15.1 or later uses one command to turn off RSTP and Junos releases prior to 15.1 uses another command.

```
Junos OS release 15.1 or later: set protocols rstp interface ge-0/0/30 disable
Junos OS release 15.1 or later: set protocols rstp interface ge-0/0/20 disable
Junos OS release prior to 15.1: delete rstp interface ge-0/0/30
Junos OS release prior to 15.1: delete rstp interface ge-0/0/20
set protocols protection-group ethernet-ring erp1
set protocols protection-group ethernet-ring erp1 data-channel 101
set protocols protection-group ethernet-ring erp1 data-channel 102
set protocols protection-group ethernet-ring erp1 control-vlan 100
set protocols protection-group ethernet-ring erp1 east-interface control-channel ge-0/0/30.0
set protocols protection-group ethernet-ring erp1 west-interface control-channel ge-0/0/20.0
set protocols protection-group ethernet-ring erp1 west-interface control-channel vlan 100
set protocols protection-group ethernet-ring erp1 east-interface control-channel vlan 100
```

Step-by-Step Procedure

To configure ERPS on Jas6-esc:

1. Disable any spanning-tree protocol currently configured on the ERPS interfaces. RSTP, VSTP, and MSTP are all available spanning-tree protocols. RSTP is enabled in the default configuration, so this example shows disabling RSTP. Spanning-tree protocols are disabled two different ways, depending on which version of the Junos OS you are running.

   If you are running Junos release 15.1 or later, disable any spanning-tree protocol with these commands. To disable RSTP:

   ```
   [edit protocols]
   user@switch# set rstp interface ge-0/0/30 disable
   user@switch# set rstp interface ge-0/0/20 disable
   ```

   If you are running a Junos release prior to 15.1, disable any spanning-tree protocol with these commands. To disable RSTP:

   ```
   [edit protocols]
   user@switch# delete rstp interface ge-0/0/30
   user@switch# delete rstp interface ge-0/0/20
   ```

2. Create a node ring named erp1:

   ```
   [edit protocols]
   user@switch# set protection-group ethernet-ring erp1
   ```

3. Configure the control vlan 100 for the node ring erp1:

   ```
   [edit protocols protection-group ethernet-ring erp1]
   user@switch# set control-vlan 100
   ```

4. Configure two data channels numbered 101 and 102 to define VLAN IDs that belong to a ring instance.

```
[edit protocols protection-group ethernet-ring erp1]
user@switch# set data-channel 101
user@switch# set data-channel 102
```

5. Configure the east interface of the node ring erp1 with the control channel ge-0/0/30.0 :

```
[edit protocols protection-group ethernet-ring erp1]
user@switch# set east-interface control-channel ge-0/0/30.0
```

6. Configure the west interface of the node ring erp1 with the control channel ge-0/0/20.0:

```
[edit protocols protection-group ethernet-ring erp1]
user@switch# set west-interface control-channel ge-0/0/20.0
```

7. Every ring instance on a trunk port has one control VLAN in which ERP packets traverse. The control VLAN also controls data VLANs, if any are configured. Assign vlan number 100 as the control VLAN:

```
[edit protocols protection-group ethernet-ring erp1]
user@switch# set west-interface control-channel vlan 100
user@switch# set east-interface control-channel vlan 100
```

RELATED DOCUMENTATION

Configuring Ethernet Ring Protection Switching on Switches (CLI Procedure)

Ethernet Ring Protection Switching Overview

Understanding Ethernet Ring Protection Switching Functionality

29

CHAPTER

Configuring Q-in-Q Tunneling and VLAN Translation

IN THIS CHAPTER

Configuring Q-in-Q Tunneling and VLAN Q-in-Q Tunneling and VLAN Translation

IN THIS SECTION

Understanding Q-in-Q Tunneling and VLAN Translation

IN THIS SECTION

Q-in-Q tunneling and VLAN translation allow service providers to create a Layer 2 Ethernet connection between two customer sites. Providers can segregate different customers' VLAN traffic on a link (for example, if the customers use overlapping VLAN IDs) or bundle different customer VLANs into a single service VLAN. Data centers can use Q-in-Q tunneling and VLAN translation to isolate customer traffic within a single site or to enable customer traffic flows between cloud data centers in different geographic locations.

Using Q-in-Q tunneling, providers can segregate or bundle customer traffic into fewer VLANs or different VLANs by adding another layer of 802.1Q tags. Q-in-Q tunneling is useful when customers have overlapping VLAN IDs, because the customer's 802.1Q (dot1Q) VLAN tags are prepended by the service VLAN (S-VLAN) tag. The Juniper Networks Junos operating system (Junos OS) implementation of Q-in-Q tunneling supports the IEEE 802.1ad standard.

This topic describes:

How Q-in-Q Tunneling Works

In Q-in-Q tunneling, as a packet travels from a customer VLAN (C-VLAN) to a service provider's VLAN, a customer-specific 802.1Q tag is added to the packet. This additional tag is used to segregate traffic into service-provider-defined service VLANs (S-VLANs). The original customer 802.1Q tag of the packet remains and is transmitted transparently, passing through the service provider's network. As the packet leaves the S-VLAN in the downstream direction, the extra 802.1Q tag is removed.

NOTE: All of the VLANs in an implementation can be service VLANs. That is, if the total number of supported VLANs is 4090, all of them can be service VLANs.

When Q-in-Q tunneling is enabled, trunk interfaces are assumed to be part of the service provider network and access interfaces are assumed to be customer facing. An access interface can receive both tagged and untagged frames in this case.

NOTE: Starting with Junos OS 14.1X53-D30, you can configure the same interface to be an S-VLAN/NNI interface and a C-VLAN/UNI interface. This means that the same physical interface can transmit single-tagged and double-tagged frames simultaneously. This allows you maximum flexibility in your network topology and lets you maximize the use of your interfaces.

An interface can be a member of multiple S-VLANs. You can map one C-VLAN to one S-VLAN (1:1) or multiple C-VLANs to one S-VLAN (N:1). Packets are double-tagged for an additional layer of segregating or bundling of C-VLANs. C-VLAN and S-VLAN tags are unique; so you can have both a C-VLAN 101 and an S-VLAN 101, for example. You can limit the set of accepted customer tags to a range of tags or to discrete values. Class-of-service (CoS) values of C-VLANs are unchanged in the downstream direction. You may, optionally, copy ingress priority and CoS settings to the S-VLAN. On non-ELS switches, you can use private VLANs to isolate users to prevent the forwarding of traffic between user interfaces even if the interfaces are on the same VLAN.

When Q-in-Q tunneling is enabled, trunk interfaces are assumed to be part of the service provider or data center network. Access interfaces are assumed to be customer-facing and accept both tagged and untagged frames. When using many-to-one bundling or mapping a specific interface, you must use the `native` option to specify an S-VLAN for untagged and priority tagged packets if you want to accept these packets. (Priority tagged packets have their VLAN ID set to 0, and their priority code point bits might be configured with a CoS value.)

NOTE: Priority tagged packets are not supported with Q-in-Q tunneling on QFX5100 and EX4600 switches.

If you do not specify an S-VLAN for them, untagged packets are discarded. The `native` option is not available for all-in-one bundling because there is no need to specify untagged and priority tagged packets when all packets are mapped to an S-VLAN.

You can use the `native` option to specify an S-VLAN for untagged and priority tagged packets when using many-to-one bundling and mapping a specific interface approaches to map C-VLANs to S-VLANs. (This does not apply to switches supporting ELS.) Otherwise the packets are discarded. The `native` option is not available for all-in-one bundling because there is no need to specify untagged and priority tagged packets when all packets are mapped to the S-VLAN. See the Mapping C-VLANs to S-VLANs section of this document for information on the methods of mapping C-VLANs to S-VLANs.

On QFabric systems only, you can use the `native` option to apply a specified inner tag to packets that ingress as untagged on access interfaces. This functionality is useful if your QFabric system connects to servers that host customer virtual machines that send untagged traffic and each customer's traffic requires its own VLAN while being transported through the QFabric. Instead of using individual VLANs for each customer (which can quickly lead to VLAN exhaustion), you can apply a unique inner (C-VLAN) tag to each customer's traffic and then apply a single outer tag (S-VLAN) tag for transport through the QFabric. This allows you to segregate your customers's traffic while consuming only one QFabric VLAN. Use the `inner-tag` option of the *mapping* statement to accomplish this.

On non-ELS switches, firewall filters allow you to map an interface to a VLAN based on a policy. Using firewall filters to map an interface to a VLAN is useful when you want a subset of traffic from a port to be mapped to a selected VLAN instead of the designated VLAN. To configure a *firewall filter* to map an

interface to a VLAN, the `vlan` option has to be configured as part of the firewall filter and the `mapping policy` option must be specified in the interface configuration for each *logical interface* using the filter.

NOTE: On an EX4300 switch, you can configure multiple logical interfaces on the same Ethernet port, but each logical interface supports only single-tagged packets and that tag must include a different VLAN ID than those supported by the other logical interfaces. Given this situation, you cannot enable Q-in-Q tunneling on Ethernet ports with multiple logical subinterfaces.

Q-in-Q tunneling does not affect any class-of-service (CoS) values that are configured on a C-VLAN. These settings are retained in the C-VLAN tag and can be used after a packet leaves an S-VLAN. CoS values are not copied from C-VLAN tags to S-VLAN tags.

Depending on your interface configuration, you might need to adjust the MTU value on your trunk or access ports to accommodate the 4 bytes used for the tag added by Q-in-Q tunneling. For example, if you use the default MTU value of 1514 bytes on your access and trunk ports, you need to make one of the following adjustments:

- Reduce the MTU on the access links by at least 4 bytes so that the frames do not exceed the MTU of the trunk link when S-VLAN tags are added.
- Increase the MTU on the trunk link so that the link can handle the larger frame size.

NOTE: You can configure Q-in-Q tunneling only on access ports (not trunk ports).

How VLAN Translation Works

VLAN translation replaces an incoming C-VLAN tag with an S-VLAN tag instead of adding an additional tag. The C-VLAN tag is therefore lost, so a single-tagged packet is normally untagged when it leaves the S-VLAN (at the other end of the link). If an incoming packet has had Q-in-Q tunneling applied in advance, VLAN translation replaces the outer tag and the inner tag is retained when the packet leaves the S-VLAN at the other end of the link. Incoming packets whose tags do not match the C-VLAN tag are dropped, unless additional VLAN translation configuration for those tags exist.

To configure VLAN translation, use the *mapping* `swap` statement at the `[edit vlans interface]` hierarchy level. As long as the C-VLAN and S-VLAN tags are unique, you can configure more than one C-VLAN-to-S-VLAN translation on an access port. If you are translating only one VLAN on an interface, you do not need to include the `dot1q-tunneling` statement in the S-VLAN configuration. If you are translating more than one VLAN, you must use the `dot1q-tunneling` statement.

NOTE: You can configure VLAN translation on access ports only. You cannot configure it on trunk ports, and you cannot configure Q-in-Q tunneling on the same access port. You can configure only one VLAN translation for a given VLAN and interface. For example, you can create no more than one translation for VLAN 100 on interface xe-0/0/0.

Using Dual VLAN Tag Translation

Starting with Junos OS Release 14.1X53-D40, you can use the dual VLAN tag translation (also known as dual VLAN tag rewrite) feature to deploy switches in service-provider domains, allowing dual-tagged, single-tagged, and untagged VLAN packets to come into or exit from the switch. Table 124 on page 945 shows the operations that are added for dual VLAN tag translation.

Table 124: Operations Added with Dual VLAN Tag Rewrite

Operation	Function
swap-push	Swap a VLAN tag and push a new VLAN tag
pop-swap	Pop an outer VLAN tag and swap an inner VLAN tag
swap-swap	Swap both outer and inner VLAN tags

Dual VLAN tag translation supports:

- Configuration of S-VLANs (NNI) and C-VLANs (UNI) on the same physical interface
- Control protocols such as VSTP, OSPF, and LACP
- IGMP snooping
- Configuration of a private VLAN (PVLAN) and VLAN on a single-tagged interface
- Use of TPID 0x8100 on both inner and outer VLAN tags

See "Setting Up a Dual VLAN Tag Translation Configuration on QFX Switches" on page 987.

Sending and Receiving Untagged Packets

To enable an interface to send and receive untagged packets, you must specify a native VLAN for a physical interface. When the interface receives an untagged packet, it adds the VLAN ID of the native

VLAN to the packet in the C-VLAN field and adds the S-VLAN tag as well (so the packet is double-tagged), and sends the newly tagged packet to the mapped interface.

The preceding paragraph does *not* apply to:

- Non-ELS switches.
- EX4300 switches running under a Junos release prior to Junos OS Release 19.3R1.

When the switches in the short list above receive an untagged packet, they add the S-VLAN tag to the packet (so the packet is single-tagged) and send the newly tagged packet to the mapped interface.

NOTE: Ensure that all switches configured in your Q-in-Q setup operate with either the single-tag approach or the double-tag approach. The setup will not work if the switches do not have the same approach.

Starting in Junos OS Release 19.3R1, you can configure EX4300 switches to use the double-tag approach. Set the configuration statement *input-native-vlan-push* to `enable` and ensure that the *input-vlan-map* configuration statement is set to `push`, as shown in the following example:

```
[edit interfaces ge-1/0/45]
flexible-vlan-tagging;
native-vlan-id 20;
input-native-vlan-push enable;
encapsulation extended-vlan-bridge;
unit 10 {
        vlan-id-list 10-100;
        input-vlan-map push;
        output-vlan-map pop;
}
```

NOTE: On switches that support this feature, except for the EX4300 switch, the *input-native-vlan-push* statement is set to `enable` by default. (The `input-native-vlan-push` statement is set to `disable` by default on the EX4300 switch.) However, we recommend that you check the configuration to ensure that *input-vlan-map* is set to `push`—the feature does not work if that setting isn't in place.

To specify a native VLAN, use the `native-vlan-id` statement at the `[edit interfaces` *`interface-name`*`]` hierarchy level. The native VLAN ID must match the C-VLAN or S-VLAN ID or be included in the VLAN ID list specified on the logical interface.

For example, on a logical interface for a C-VLAN interface, you might specify a C-VLAN ID list of 100-200. Then, on the C-VLAN physical interface, you could specify a native VLAN ID of 150. This configuration would work because the native VLAN of 150 is included in the C-VLAN ID list of 100-200.

We recommend configuring a native VLAN when using any of the approaches to map C-VLANs to S-VLANs. See the Mapping C-VLANs to S-VLANs section in this topic for information about the methods of mapping C-VLANs to S-VLANs.

Disabling MAC Address Learning

In a Q-in-Q deployment, customer packets from downstream interfaces are transported without any changes to source and destination MAC addresses. You can disable MAC address learning at global, interface, and VLAN levels:

- To disable learning globally, disable MAC address learning for the switch.
- To disable learning for an interface, disable MAC address learning for all VLANs of which the specified interface is a member.
- To disable learning for a VLAN, disable MAC address learning for a specified VLAN.

Disabling MAC address learning on an interface disables learning for all the VLANs of which that interface is a member. When you disable MAC address learning on a VLAN, MAC addresses that have already been learned are flushed.

If you disable MAC address learning on an interface or a VLAN, you cannot include 802.1X authentication in that same VLAN configuration.

When a *routed VLAN interface* (*RVI*) is associated with either an interface or a VLAN on which MAC address learning is disabled, the Layer 3 routes resolved on that VLAN or that interface are not resolved with the Layer 2 component. This results in routed packets flooding all the interfaces associated with the VLAN.

Mapping C-VLANs to S-VLANs

There are multiple ways to map C-VLANs to an S-VLAN:

NOTE: If you configure multiple mapping methods, the switch gives priority to mapping a specific interface, then to many-to-many bundling, and last to all-in-one bundling. However, for a particular mapping method, setting up overlapping rules for the same C-VLAN is not supported.

- All-in-one bundling—Use the `edit vlans` *`s-vlan-name`* `dot1q-tunneling` statement without specifying customer VLANs. All packets received on all access interfaces (including untagged packets) are mapped to the S-VLAN.

- Many-to-one bundling—Use the `edit vlans` *`s-vlan-name`* `dot1q-tunneling customer-vlans` statement to specify which C-VLANs are mapped to the S-VLAN. Use this method when you want a subset of the C-VLANs to be part of the S-VLAN. If you want untagged or priority tagged packets to be mapped to the S-VLAN, use the `native` option with the `customer-vlans` statement. (Priority tagged packets have their VLAN ID set to 0, and their priority code point bits might be configured with a CoS value.)

- Many-to-many bundling—Use many-to-many bundling when you want a subset of the C-VLANs on the access switch to be part of multiple S-VLANs.

- Mapping a specific interface—Use the `edit vlans` *`s-vlan-name`* `interface` *`interface-name`* `mapping` statement to specify a C-VLAN for a given S-VLAN. This configuration applies to only one interface—not all access interfaces as with all-in-one and many-to-one bundling. If you want untagged or priority tagged packets to be mapped to the S-VLAN, use the `native` option with the `customer-vlans` statement.

  This method has two options: swap and push. With the push option, a packet retains its tag and an additional VLAN tag is added. With the swap option, the incoming tag is replaced with an S-VLAN tag. (This is VLAN translation.)

  - You can configure multiple push rules for a given S-VLAN and interface. That is, you can configure an interface so that the same S-VLAN tag is added to packets arriving from multiple C-VLANs.

  - You can configure only one swap rule for a given S-VLAN and interface.

  This functionality is typically used to keep traffic from different customers separate or to provide individualized treatment for traffic on a certain interface.

If you configure multiple methods, the switch gives priority to mapping a specific interface, then to many-to-one bundling, and last to all-in-one bundling. However, you cannot have overlapping rules for the same C-VLAN under a given approach. For example, you cannot use many-to one bundling to map C-VLAN 100 to two different S-VLANs.

All-in-One Bundling

All-in-one bundling maps all packets from all C-VLAN interfaces to an S-VLAN.

The C-VLAN interface accepts untagged and single-tagged packets. An S-VLAN 802.1Q tag is then added to these packets, and the packets are sent to the S-VLAN interface, which accepts untagged, single-tagged, and double-tagged packets.

NOTE: The C-VLAN and S-VLAN interfaces accept untagged packets provided that the `native-vlan-id` statement is configured on these interfaces.

Many-to-One Bundling

Many-to-one bundling is used to specify which C-VLANs are mapped to an S-VLAN. Many-to-one bundling is configured using the `customer-vlans` option.

Many-to-one bundling is used when you want a subset of the C-VLANs on the access switch to be part of the S-VLAN. When using many-to-one bundling, untagged and priority tagged packets can be mapped to the S-VLAN when the `native` option is specified along with the `customer-vlans` option.

Many-to-Many Bundling

Many-to-many bundling is used to specify which C-VLANs are mapped to which S-VLANs.

Use many-to-many bundling when you want a subset of the C-VLANs on the access switch to be part of multiple S-VLANs. With many-to-many bundling, the C-VLAN interfaces accept untagged and single-tagged packets. An S-VLAN 802.1Q tag is then added to these packets, and the packets are sent to the S-VLAN interfaces, which accept untagged, single-tagged, and double-tagged packets.

NOTE: The C-VLAN and S-VLAN interfaces accept untagged packets provided that the `native-vlan-id` statement is configured on these interfaces.

Mapping a Specific Interface

Use specific interface mapping when you want to assign an S-VLAN to a specific C-VLAN on an interface. The configuration applies only to the specific interface, not to all access interfaces.

Specific interface mapping has two suboptions: `push` and `swap`. When traffic that is mapped to a specific interface is pushed, the packet retains its original tag as it moves from the C-VLAN to the S-VLAN and an additional S-VLAN tag is added to the packet. When traffic that is mapped to a specific interface is swapped, the incoming tag is replaced with a new VLAN tag. This is sometimes known VLAN rewriting or VLAN translation.

Typically, this method is used to keep data from different customers separate or to provide individualized treatment of the packets on a certain interface. You might also use this method map VLAN traffic from different customers to a single S-VLAN.

When using specific interface mapping, the C-VLAN interfaces accept untagged and single-tagged packets, while the S-VLAN interfaces accept untagged, single-tagged, and double-tagged packets.

NOTE: The C-VLAN and S-VLAN interfaces accept untagged packets provided that the `native-vlan-id` statement is configured on these interfaces.

Combining Methods and Configuration Restrictions

If you configure multiple methods, the switch gives priority to mapping a specific interface, then to many-to-one bundling, and last to all-in-one bundling. An access interface configured under all-in-one bundle cannot be part of a many-to-one bundle. It can have additional mappings defined, however.

To ensure deterministic results, the following configuration restrictions apply:

- Mapping cannot be defined for untagged vlans.
- An access interface can have multiple customer VLAN ranges, but an interface cannot have overlapping tags across the VLANs.
- An access interface can have a single rule that maps an untagged packet to a VLAN.
- Each interface can have at most one mapping swap rule per VLAN.
- You can push a VLAN tag only on the access ports of a Q-in-Q VLAN. This restriction applies to all three methods of pushing a VLAN tag: that is, all-in-one bundling, many-to-one-bunding, and mapping a specific interface using push.
- You can push different C-VLAN tags for a given S-VLAN on different interfaces. This could potentially result in traffic leaking across VLANs, depending on your configuration.

Routed VLAN Interfaces on Q-in-Q VLANs

Routed VLAN interfaces (RVIs) are supported on Q-in-Q VLANs.

Packets arriving on an RVI that is using Q-in-Q VLANs will get routed regardless of whether the packet is single or double tagged. The outgoing routed packets contain an S-VLAN tag only when exiting a trunk interface; the packets exit the interface untagged when exiting an access interface.

Constraints for Q-in-Q Tunneling and VLAN Translation

Be aware of the following constraints when configuring Q-in-Q tunneling and VLAN translation:

- Q-in-Q tunneling supports only two VLAN tags.
- Q-in-Q tunneling does not support most access port security features. There is no per-VLAN (customer) policing or per-VLAN (outgoing) shaping and limiting with Q-in-Q tunneling unless you configure these security features by using firewall filters.

- With releases of Junos OS Release 13.2X51 previous to Release 13.2X51-D20, you cannot create a regular VLAN on an interface if you have created an S-VLAN or C-VLAN on that interface for Q-in-Q tunneling. This means that you cannot create an integrated routing and bridging (IRB) interface on that interface because regular VLANs are a required part of IRB configuration. With Junos OS Release 13.2X51-D25, you can create a regular VLAN on a trunk interface that has an S-VLAN, which means that you can also create an IRB interface on the trunk. In this case, the regular VLAN and S-VLAN on the same trunk interface cannot share the same VLAN ID. Junos OS Release 13.2X51-D25 does not allow you to create a regular VLAN on an access interface that has a C-VLAN.

- (On non-ELS switches only) Starting with Junos OS Release 14.1X53-D40, integrated routing and bridging (IRB) interfaces are supported on Q-in-Q VLANs—you can configure the IRB interface on the same interface as one used by an S-VLAN, and you can use the same VLAN ID for both the VLAN used by the IRB interface and for the VLAN used as an S-VLAN.

  Packets arriving on an IRB interface that is using Q-in-Q VLANs will get routed regardless of whether the packet is single tagged or double tagged. The outgoing routed packets contain an S-VLAN tag only when exiting a trunk interface; the packets exit the interface untagged when exiting an access interface.

  NOTE: (On non-ELS switches only) You can configure the IRB interface only on S-VLAN (NNI) interfaces, not on C-VLAN (UNI) interfaces.

- Support for QFX5K switches with Q-in-Q interfaces using the `vlan-tags` statement is limited to Layer 2 interfaces. Layer 3 interfaces that are configured with Q-in-Q `vlan-tags` statements might not function as expected.

- QFX10K switches cannot **push** a third or fourth tag on a double-tagged or triple-tagged packet, respectively. For example, if the incoming packet already has two tags, a third S-VLAN tag will not be pushed. If a push operation is configured in such instances, the resulting packet is malformed.

- Most access port security features are not supported with Q-in-Q tunneling and VLAN translation.

- Configuring Q-in-Q tunneling and VLAN rewriting/VLAN translation on the same port is not supported.

- You can configure at most one VLAN rewrite/VLAN translation for a given VLAN and interface. For example, you can create no more than one translation for VLAN 100 on interface xe-0/0/0.

- The combined total of VLANs and rules for Q-in-Q tunneling and VLAN translation cannot exceed 6000. For example, you can configure and commit 4000 VLANs and 2000 rules for Q-in-Q tunneling and VLAN translation. However, you cannot configure 4000 VLANs and 2500 rules for Q-in-Q tunneling and VLAN translation. If you try to commit a configuration that exceeds the limit, you see CLI and syslog errors that inform you about the problem.

- You cannot use the native VLAN ID.
- MAC addresses are learned from S-VLANs, not C-VLANs.
- Broadcast, unknown unicast, and multicast traffic is forwarded to all members in the S-VLAN.
- The following features are not supported with Q-in-Q tunneling:
  - DHCP relay
  - Fibre Channel over Ethernet
  - IP Source Guard
- The following features are not supported with VLAN rewriting/VLAN translation:
  - Fibre Channel over Ethernet
  - Firewall filter applied to a port or VLAN in the output direction
  - Private VLANs
  - VLAN Spanning Tree Protocol
  - Reflective relay

Configuring Q-in-Q Tunneling on QFX Series Switches

Q-in-Q tunneling and VLAN translation allow service providers to create a Layer 2 Ethernet connection between two customer sites. Providers can segregate different customers' VLAN traffic on a link (for example, if the customers use overlapping VLAN IDs) or bundle different customer VLANs into a single service VLAN. Data centers can use Q-in-Q tunneling to isolate customer traffic within a single site or when customer traffic flows between cloud data centers in different geographic locations.

NOTE: In the QFX5110, QFX5120, QFX5200, QFX5210, and EX4650 Junos platforms, you'll need to configure interface devices only with a service provider-style configuration with Q-in-Q tunnelling instead of using a combination of enterprise provider and service provider-style configurations.

Starting in Junos OS Release 19.4R1, the QFX10000 line of switches supports the third and fourth Q-in-Q tags as payload (also known as a pass-through tag) along with the existing two tags (for VLAN matching and operations). The QFX10000 switches support multiple Q-in-Q tags for both Layer 2 bridging and EVPN-VXLAN cases. The Layer 2 access interfaces accept packets with three or four tags

(all tags with the TPID value 0x8100). All the tags beyond the fourth tag (that is, from the fifth tag onward) are considered part of the Layer 3 payload and are forwarded transparently.

NOTE: In a one or two tagged packet, the tags, tag 1 and tag 2 can carry any TPID values such as 0x8100, 0x88a8, 0x9100, and 0x9200.

Before you begin setting up Q-in-Q tunneling, make sure you have created and configured the necessary customer VLANs on the neighboring switches. See "Configuring VLANs on QFX Series Switches without Enhanced Layer 2 Support" on page 152.

To configure Q-in-Q tunneling:

1. Create the service VLAN (S-VLAN) and configure an ID for it:

```
[edit vlans]
user@switch# set s-vlan-name vlan-ids-vlan-ID
```

2. Enable Q-in-Q tunneling on the S-VLAN:

```
[edit vlans]
user@switch# set s-vlan-name dot1q-tunneling
```

3. Set the allowed customer VLANs (C-VLANs) on the S-VLAN (optional). Here, the C-VLANs are identified by a range:

```
[edit vlans]
user@switch# set s-vlan-name dot1q-tunneling customer-vlans range
```

4. Configure a global value for the tag protocol identifier (EtherType) of the service VLAN tags (optional):

```
[edit]
user@switch# set ethernet-switching-options dot1q-tunneling ether-type ether-type-value
```

Depending on your interface configuration, you might need to adjust the MTU value on your trunk or access ports to accommodate the 4 bytes used for the tag added by Q-in-Q tunneling. For example, if you use the default MTU value of 1514 bytes on your access and trunk ports, you need to make one of the following adjustments:

- Reduce the MTU on the access links by at least 4 bytes so that the frames do not exceed the MTU of the trunk link when S-VLAN tags are added.

- Increase the MTU on the trunk link so that the link can handle the larger frame size.

Configuring Q-in-Q Tunneling on EX Series Switches with ELS Support

IN THIS SECTION

NOTE: This task uses Junos OS for EX Series switches with support for the Enhanced Layer 2 Software (ELS) configuration style. If your switch runs software that does not support ELS, see "Configuring Q-in-Q Tunneling on EX Series Switches" on page 963. For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15.

Q-in-Q tunneling enables service providers on Ethernet access networks to segregate or bundle customer traffic into different VLANs by adding another layer of 802.1Q tags. You can configure Q-in-Q tunneling on EX Series switches.

NOTE: You cannot configure 802.1X user authentication on interfaces that have been enabled for Q-in-Q tunneling.

When Q-in-Q tunneling is configured on EX Series switches, trunk interfaces are assumed to be part of the service-provider network and access interfaces are assumed to be part of the customer network. Therefore, this topic also refers to trunk interfaces as service-provider VLAN (S-VLAN) interfaces (network-to-network interfaces [NNI]), and to access interfaces as customer VLAN (C-VLAN) interfaces (user-network interfaces [UNI]).

Before you begin configuring Q-in-Q tunneling, make sure you set up your VLANs. See Configuring VLANs for EX Series Switches with ELS Support (CLI Procedure) or Configuring VLANs for EX Series Switches (J-Web Procedure).

Configure Q-in-Q tunneling by using one of the following methods to map C-VLANs to S-VLANs:

Configuring All-in-One Bundling

You can configure Q-in-Q tunneling by using the all-in-one bundling method, which maps packets from all C-VLAN interfaces on a switch to an S-VLAN.

To configure the all-in-one bundling method on a C-VLAN interface:

1. Enable the transmission of packets with no or a single 802.1Q VLAN tag:

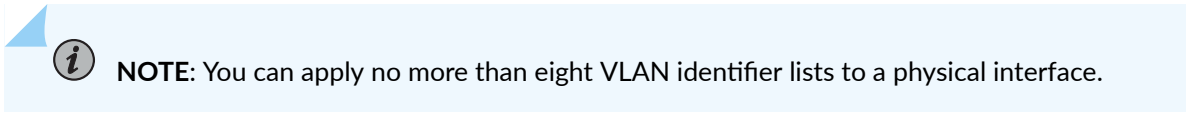
```
[edit interfaces interface-name]
user@switch# set flexible-vlan-tagging
```

2. Enable extended VLAN bridge encapsulation:

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

3. Map packets from all C-VLANs to a logical interface:

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



NOTE: You can apply no more than eight VLAN identifier lists to a physical interface.

4. Enable a C-VLAN interface to send and receive untagged packets:

```
[edit interfaces interface-name]
user@switch# set native-vlan-id vlan-id
```

When specifying a native VLAN ID on a C-VLAN physical interface, the value must be included in the VLAN ID list specified on the C-VLAN logical interface in step 3.

5. Specify that packets traveling from a C-VLAN interface to an S-VLAN interface are tagged with the VLAN ID of the S-VLAN:

```
[edit interfaces interface-name unit logical-unit-number]
user@switch# set input-vlan-map push
```

6. Specify that the 802.1Q S-VLAN tag is removed as packets exit an S-VLAN interface.

```
[edit interfaces interface-name unit logical-unit-number]
user@switch# set output-vlan-map pop
```

7. Configure a name for the S-VLAN, and associate the logical interface configured in step 3 with the S-VLAN:

```
[edit vlans vlan-name]
user@switch# set interface interface-name.logical-unit-number
```

The following configuration on the C-VLAN interface ge-0/0/1 enables Q-in-Q tunneling and maps packets from C-VLANs 100 through 200 to logical interface 10, which is in turn associated with S-VLAN v10. In this sample configuration, a packet originated in C-VLAN 100 includes a tag with the VLAN ID 100. When this packet travels from the interface ge-0/0/1 to the S-VLAN interface, a tag with VLAN ID 10 is added to it. As the packet exits the S-VLAN interface, the tag with the VLAN ID 10 is removed.

```
set interfaces ge-0/0/1 flexible-vlan-tagging
set interfaces ge-0/0/1 encapsulation extended-vlan-bridge
set interfaces ge-0/0/1 unit 10 vlan-id-list 100-200
set interfaces ge-0/0/1 native-vlan-id 150
set interfaces ge-0/0/1 unit 10 input-vlan-map push
set interfaces ge-0/0/1 unit 10 output-vlan-map pop
set vlans v10 interface ge-0/0/1.10
```

To configure the all-in-one bundling method on an S-VLAN interface:

1. Enable the transmission of packets with no, one, or two 802.1Q VLAN tags:

```
[edit interfaces interface-name]
user@switch# set flexible-vlan-tagging
```

2. Enable extended VLAN bridge encapsulation:

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

3. Map packets from the logical interface specified in the C-VLAN interface configuration to the S-VLAN:

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

4. Enable the S-VLAN interface to send and receive untagged packets:

```
[edit interfaces interface-name]
user@switch# set native-vlan-id vlan-id
```

   When specifying a native VLAN ID on an S-VLAN physical interface, the value must match the VLAN ID specified on the S-VLAN logical interface in step 3.

5. Associate the S-VLAN interface with the S-VLAN that was configured in the C-VLAN interface procedure:

```
[edit vlans vlan-name]
user@switch# set interface interface-name.logical-unit-number
```

For example, the following configuration on the S-VLAN interface ge-1/1/1 enables Q-in-Q tunneling and maps packets with a VLAN ID tag of 10 to logical interface 10, which is in turn associated with S-VLAN v10.

```
set interfaces ge-1/1/1 flexible-vlan-tagging
set interfaces ge-1/1/1 encapsulation extended-vlan-bridge
set interfaces ge-1/1/1 unit 10 vlan-id 10
set interfaces ge-1/1/1 native-vlan-id 10
set vlans v10 interface ge-1/1/1.10
```

Configuring Many-to-Many Bundling

You can configure Q-in-Q tunneling by using the many-to-many bundling method, which maps packets from multiple C-VLANs to multiple S-VLANs.

To configure the many-to-many bundling method on a C-VLAN interface:

1. Enable the transmission of packets with no or a single 802.1Q VLAN tag:

```
[edit interfaces interface-name]
user@switch# set flexible-vlan-tagging
```

2. Enable extended VLAN bridge encapsulation:

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

3. Map packets from specified C-VLANs to a logical interface:

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

4. Enable a C-VLAN interface to send and receive untagged packets:

```
[edit interfaces interface-name]
user@switch# set native-vlan-id vlan-id
```

   When specifying a native VLAN ID on a C-VLAN physical interface, the value must be included in the VLAN ID list specified on the C-VLAN logical interface in step 3.

5. Specify that packets traveling from a C-VLAN interface to an S-VLAN interface are tagged with the VLAN ID of the S-VLAN:

```
[edit interfaces interface-name unit logical-unit-number]
user@switch# set input-vlan-map push
```

6. Specify that the 802.1Q S-VLAN tag is removed as packets exit an S-VLAN interface:

```
[edit interfaces interface-name unit logical-unit-number]
user@switch# set output-vlan-map pop
```

7. Configure a name for an S-VLAN, and associate the logical interface configured in step 3 with the S-VLAN:

```
[edit vlans vlan-name]
user@switch# set interface interface-name.logical-unit-number
```

The following configuration on the C-VLAN interface ge-0/0/1 for customer 1 enables Q-in-Q tunneling and maps packets from C-VLANs 100 through 120 to logical interface 10, which is in turn associated with S-VLAN v10.

The configuration on the C-VLAN interface ge-0/0/2 for customer 2 enables Q-in-Q tunneling and maps packets from C- VLANs 30 through 40, 50 through 60, and 70 through 80 to logical interface 30, which is in turn associated with S- VLAN v30.

In this sample configuration, a packet originated in C-VLAN 100 includes a tag with the VLAN ID 100. When this packet travels from the interface ge-0/0/1 to the S-VLAN interface, a tag with a VLAN ID of 10 is added to it. As the packet exits the S-VLAN interface, the tag with the VLAN ID of 10 is removed.

Customer 1

```
set interfaces ge-0/0/1 flexible-vlan-tagging
set interfaces ge-0/0/1 encapsulation extended-vlan-bridge
set interfaces ge-0/0/1 unit 10 vlan-id-list 100-120
set interfaces ge-0/0/1 native-vlan-id 100
set interfaces ge-0/0/1 unit 10 input-vlan-map push
set interfaces ge-0/0/1 unit 10 output-vlan-map pop
set vlans v10 interface ge-0/0/1.10
```

Customer 2

```
set interfaces ge-0/0/2 flexible-vlan-tagging
set interfaces ge-0/0/2 encapsulation extended-vlan-bridge
set interfaces ge-0/0/2 unit 30 vlan-id-list 30-40
set interfaces ge-0/0/2 unit 30 vlan-id-list 50-60
set interfaces ge-0/0/2 unit 30 vlan-id-list 70-80
set interfaces ge-0/0/2 native-vlan-id 30
set interfaces ge-0/0/2 unit 30 input-vlan-map push
set interfaces ge-0/0/2 unit 30 output-vlan-map pop
set vlans v30 interface ge-0/0/2.30
```

To configure the many-to-many bundling method on an S-VLAN interface:

1. Enable the transmission of packets with no, one, or two 802.1Q VLAN tags:

```
[edit interfaces interface-name]
user@switch# set flexible-vlan-tagging
```

2. Enable extended VLAN bridge encapsulation:

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

3. Map packets from each logical interface specified in the C-VLAN interface configuration to an S-VLAN:

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

4. Enable an S-VLAN interface to send and receive untagged packets:

```
[edit interfaces interface-name]
user@switch# set native-vlan-id vlan-id
```

   When specifying a native VLAN ID on an S-VLAN physical interface, the value must match an S-VLAN ID specified on the S-VLAN logical interface in step 3.

5. Associate the S-VLAN interface with the S-VLANs that were configured in the C-VLAN interface procedure:

```
[edit vlans vlan-name]
user@switch# set interface interface-name.logical-unit-number
```

For example, the following configuration on the S-VLAN interface ge-1/1/1 enables Q-in-Q tunneling and maps incoming C-VLAN packets to logical interfaces 10 and 30, which are in turn associated with S-VLANs v10 and v30, respectively.

```
set interfaces ge-1/1/1 flexible-vlan-tagging
set interfaces ge-1/1/1 encapsulation extended-vlan-bridge
set interfaces ge-1/1/1 unit 10 vlan-id 10
set interfaces ge-1/1/1 unit 30 vlan-id 30
set interfaces ge-1/1/1 native-vlan-id 10
set vlans v10 interface ge-1/1/1.10
set vlans v30 interface ge-1/1/1.30
```

Configuring a Specific Interface Mapping with VLAN Rewrite Option

You can configure Q-in-Q tunneling by mapping packets from a specified C-VLAN to a specified S-VLAN. In addition, while the packets are transmitted to and from the S-VLAN, you can specify that the 802.1Q C-VLAN tag be removed and replaced with the S-VLAN tag or vice versa.

To configure a specific interface mapping with VLAN rewriting on the C-VLAN interface:

1. Enable the transmission of packets with no or one 802.1Q VLAN tag:

```
[edit interfaces interface-name]
user@switch# set flexible-vlan-tagging
```

2. Enable extended VLAN bridge encapsulation:

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

3. Map packets from a specified C-VLAN to a logical interface:

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

4. Enable the C-VLAN interface to send and receive untagged packets:

```
[edit interfaces interface-name]
user@switch# set native-vlan-id vlan-id
```

   When specifying a native VLAN ID on a C-VLAN physical interface, the value must match the VLAN ID specified on the C-VLAN logical interface in step 3.

5. Specify that the existing 802.1Q C-VLAN tag is removed from packets traveling from a C-VLAN interface to an S-VLAN interface and replaced with the 802.1Q S-VLAN tag:

```
[edit interfaces interface-name unit logical-unit-number]
user@switch# set input-vlan-map swap
```

6. Specify that the existing 802.1Q S-VLAN tag is removed from packets traveling from an S-VLAN interface to a C-VLAN interface and replaced with the 802.1Q C-VLAN tag:

```
[edit interfaces interface-name unit logical-unit-number]
user@switch# set output-vlan-map swap
```

7. Configure a name for the S-VLAN, and associate the logical interface configured in step 3 with the S-VLAN:

```
[edit vlans vlan-name]
user@switch# set interface interface-name.logical-unit-number
```

For example, the following configuration on the C-VLAN interface ge-0/0/1 enables Q-in-Q tunneling and maps incoming packets from C-VLAN 150 to logical interface 200, which is in turn associated with VLAN v200. Also, as packets travel from the C-VLAN interface ge-0/0/1 to an S-VLAN interface, the C-VLAN tag 150 is removed and replaced with the S-VLAN tag 200. As packets travel from an S-VLAN interface to C-VLAN interface ge-0/0/1, the S-VLAN tag 200 is removed and replaced with the C-VLAN tag of 150.

```
set interfaces ge-0/0/1 flexible-vlan-tagging
set interfaces ge-0/0/1 encapsulation extended-vlan-bridge
set interfaces ge-0/0/1 unit 200 vlan-id 150
set interfaces ge-0/0/1 native-vlan-id 150
set interfaces ge-0/0/1 unit 200 input-vlan-map swap
set interfaces ge-0/0/1 unit 200 output-vlan-map swap
set vlans v200 interface ge-0/0/1.200
```

To configure a specific interface mapping with VLAN rewriting on the S-VLAN interface:

1. Enable the transmission of packets with no, one, or two 802.1Q VLAN tags:

```
[edit interfaces interface-name]
user@switch# set flexible-vlan-tagging
```

2. Enable extended VLAN bridge encapsulation:

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

3. Map packets from the logical interface specified in the C-VLAN interface configuration to the S-VLAN:

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

4. Enable the S-VLAN interface to send and receive untagged packets:

```
[edit interfaces interface-name]
user@switch# set native-vlan-id vlan-id
```

   When specifying a native VLAN ID on an S-VLAN physical interface, the value must match the VLAN ID specified on the S-VLAN logical interface in step 3.

5. Associate the S-VLAN interface with the S-VLAN that was configured in the C-VLAN interface procedure: :

```
[edit vlans vlan-name]
user@switch# set interface interface-name.logical-unit-number
```

For example, the following configuration on the S-VLAN interface ge-1/1/1 enables Q-in-Q tunneling and maps packets with VLAN ID 200 to logical interface 200, which is in turn associated with S-VLAN v200.

```
set interfaces ge-1/1/1 flexible-vlan-tagging
set interfaces ge-1/1/1 encapsulation extended-vlan-bridge
set interfaces ge-1/1/1 unit 200 vlan-id 200
set interfaces ge-1/1/1 native-vlan-id 200
set vlans v200 interface ge-1/1/1.200
```

Configuring Q-in-Q Tunneling on EX Series Switches

NOTE: This task uses Junos OS for EX Series switches that does not support the Enhanced Layer 2 Software (ELS) configuration style.

Q-in-Q tunneling allows service providers on Ethernet access networks to segregate or bundle customer traffic into different VLANs by adding another layer of 802.1Q tags. You can configure Q-in-Q tunneling on EX Series switches.

NOTE: You cannot configure 802.1X user authentication on interfaces that have been enabled for Q-in-Q tunneling.

Before you begin configuring Q-in-Q tunneling, make sure you set up your VLANs. See *Configuring VLANs for EX Series Switches* or Configuring VLANs for EX Series Switches (J-Web Procedure).

To configure Q-in-Q tunneling:

1. Enable Q-in-Q tunneling on the S-VLAN:

```
[edit vlans]
user@switch# set s-vlan-name dot1q-tunneling
```

2. Set the allowed C-VLANs on the S-VLAN (optional). Here, the C-VLANs are identified by VLAN range:

```
[edit vlans]
user@switch# set s-vlan-name dot1q-tunneling customer-vlans range
```

3. Change the global Ethertype value (optional):

```
[edit]
user@switch# set ethernet-switching-options dot1q-tunneling ether-type ether-type-value
```

4. Disable MAC address learning on the S-VLAN (optional):

```
[edit vlans]
user@switch# set s-vlan-name no-mac-learning
```

Configuring Q-in-Q Tunneling on ACX Series

SUMMARY

IN THIS SECTION

Q-in-Q Tunneling on ACX Series Overview

Q-in-Q tunneling allows service providers to create a Layer 2 Ethernet connection between two customer sites. Providers can segregate different customers' VLAN traffic on a link (for example, if the customers use overlapping VLAN IDs) or bundle different customer VLANs into a single service VLAN. Service providers can use Q-in-Q tunneling to isolate customer traffic within a single site or to enable customer traffic flows across geographic locations.

Q-in-Q tunneling adds a service VLAN tag before the customer's 802.1Q VLAN tags. The Juniper Networks Junos operating system implementation of Q-in-Q tunneling supports the IEEE 802.1ad standard.

In Q-in-Q tunneling, as a packet travels from a customer VLAN (C-VLAN) to a service provider's VLAN (S-VLAN), another 802.1Q tag for the appropriate S-VLAN is added before the C-VLAN tag. The C-VLAN tag remains and is transmitted through the network. As the packet exits from the S-VLAN space, in the downstream direction, the S-VLAN 802.1Q tag is removed.

In ACX Series routers, you can configure Q-in-Q tunneling by explicitly configuring an input VLAN map with `push` function on customer facing interfaces in a bridge domain.

You can configure Q-in-Q tunneling on aggregated Ethernet interface by configuring input and output VLAN map.

Configuring Q-in-Q Tunneling on ACX Series

To configure Q-in-Q tunneling, you need to configure the logical interface connected to the customer network (user-to-network interfaces (UNI)) and the logical interface connected to the service provider network (network-to-network interface (NNI)).

The following is an example to configure a logical interface connected to a customer network:

```
[edit]
    interface ge-1/0/1 {
        flexible-vlan-tagging;
            encapsulation flexible-ethernet-services;
        unit 0 {
            encapsulation vlan-bridge;
                vlan-id-list 10-20;
                input-vlan-map {
                push;
                vlan-id 500;
                }
                output-vlan-map pop;
            }
    }
```

The following is an example to configure a logical interface connected to a service provider network:

```
[edit]
interface ge-1/0/2; {
    flexible-vlan-tagging;
        encapsulation flexible-ethernet-services;
        unit 0 {
        encapsulation vlan-bridge;
        vlan-id 500;
    }
}
```

The following is an example to configure the bridge domain:

```
[edit]
bridge-domains {
    qnq-stag-500{
        interface ge-1/0/1;
        interface ge-1/0/2;
    }
}
```

You can configure Q-in-Q tunneling on aggregated Ethernet interface connected to the customer network (UNI) and the logical interface connected to the service provider network (NNI).

Configuring Q-in-Q Tunneling Using All-in-One Bundling

You can configure Q-in-Q tunneling using the all-in-one bundling method for provider edge and customer edge devices. Provider edge devices forward all packets that ingress on a C-VLAN interface to an S-VLAN. Packets are forwarded to the S-VLAN regardless of whether they are tagged or untagged prior to ingress. Using this approach saves you the effort of specifying a specific mapping for each C-VLAN. Customer edge devices use L2 configurations that interface with the configured vlan.

For your provider edge device, first configure the S-VLAN and its interface:

1. Assign a logical interface (unit) to be a member of the S-VLAN.

```
[edit vlans vlan-name]
user@switch# interface interface-name.unit-number
```

NOTE: Do not use logical interface unit 0. You must later bind a VLAN tag ID to the unit you specify in this step, and you cannot bind a VLAN tag ID to unit 0. Also note that you do not create a VLAN ID for the S-VLAN. The ID is created automatically for the appropriate logical interface.

2. Enable the interface to transmit packets with two 802.1Q VLAN tags:

```
[edit interfaces interface-name]
user@switch# set flexible-vlan-tagging
```

3. Enable extended VLAN bridge encapsulation on the interface:

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

NOTE: If you configure an enterprise-style configuration such as PVLAN on the same physical interface on which you are configuring Q-in-Q tunneling, use `set encapsulation flexible-ethernet-services` . See *Understanding Flexible Ethernet Services Encapsulation on Switches*.

4. Enable the S-VLAN interface to send and receive untagged packets:

```
[edit interfaces interface-name]
user@switch# set native-vlan-id vlan-id
```

5. Bind the logical interface (unit) of the interface that you specified in step 1 to the automatically created VLAN ID for the S-VLAN:

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

NOTE: If you configured `flexible-ethernet-services`, configure `vlan-bridge` encapsulation on the logical interface:

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

For example, the following configuration makes xe-0/0/0.10 a member of VLAN 10, enables Q-in-Q tunneling on interface xe-0/0/0, enables xe-0/0/0 to accept untagged packets, and binds the VLAN ID of S-VLAN v10 to a logical interface of xe-0/0/0.

```
set vlans v10 interface xe-0/0/0.10
set interfaces xe-0/0/0 flexible-vlan-tagging
set interfaces xe-0/0/0 native-vlan-id 10
set interfaces xe-0/0/0 encapsulation extended-vlan-bridge
set interfaces xe-0/0/0 unit 10 vlan-id 10
```

Now configure all-in-one bundling on a C-VLAN interface:

1. Assign a logical interface (unit) of the C-VLAN interface to be a member of the S-VLAN.

```
[edit vlans vlan-name]
user@switch# set interface interface-name.unit-number
```

2. Enable the interface to transmit packets with 802.1Q VLAN tags :

```
[edit interfaces interface-name]
user@switch# set flexible-vlan-tagging
```

3. Enable extended VLAN bridge encapsulation on the interface:

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

4. Enable the C-VLAN interface to send and receive untagged packets:

```
[edit interfaces interface-name]
user@switch# set native-vlan-id vlan-id
```

5. Configure a logical interface to receive and forward any tagged packet whose VLAN ID tag matches the list of VLAN IDs you specify:

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

CAUTION: You can apply no more than eight VLAN identifier lists to a physical interface. This limitation does not apply to QFX10000 switches.

6. Configure the system to add an S-VLAN tag (outer tag) as packets travel from a C-VLAN interface to the S-VLAN:

```
[edit interfaces interface-name unit logical-unit-number]
user@switch# set input-vlan-map push
```

NOTE: You can configure `vlan-id` on `input-vlan-map`, but doing so is optional.

7. Configure the system to remove the S-VLAN tag when packets are forwarded (internally) from the S-VLAN interface to the C-VLAN interface:

```
[edit interfaces interface-name unit logical-unit-number]
user@switch# set output-vlan-map pop
```

For example, the following configuration makes xe-0/0/1.10 a member of S-VLAN v10, enables Q-in-Q tunneling, maps packets from C-VLANs 100 through 200 to S-VLAN 10, and enables xe-0/0/1 to accept untagged packets. If a packet originates in C-VLAN 100 and needs to be sent across the S-VLAN, a tag with VLAN ID 10 is added to the packet. When a packet is forwarded (internally) from the S-VLAN interface to interface xe-0/0/1, the tag with VLAN ID 10 is removed.

```
set vlans v10 interface xe-0/0/1.10
set interfaces xe-0/0/1 flexible-vlan-tagging
set interfaces xe-0/0/1 encapsulation extended-vlan-bridge
set interfaces xe-0/0/1 unit 10 vlan-id-list 100-200
set interfaces xe-0/0/1 native-vlan-id 10
set interfaces xe-0/0/1 unit 10 input-vlan-map push
set interfaces xe-0/0/1 unit 10 output-vlan-map pop
```

To configure the vlan that you'll use to forward packets from your customer edge device, configure ethernet mode as **ethernet-switching** and add the associated **vlan-id** tag, which then sends the tagged frames.

For example, the following configuration makes xe-0/0/1.0 a member of vlan 100 and enables Q-in-Q tunneling on interface xe-0/0/1.

```
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 100
set vlans vlan100 vlan-id 100
```

Configuring Q-in-Q Tunneling Using Many-to-Many Bundling

You can configure Q-in-Q tunneling using the many-to-many bundling method, which maps packets from multiple C-VLANs to multiple S-VLANs. This method is convenient for mapping a range of C-VLANs without having to specify each one individually. (You can also use this method to configure only one C-VLAN to be mapped to an S-VLAN.)

First configure the S-VLANs and assign them to an interface:

1. Assign a logical interface (unit) to be a member of one of the S-VLANs. Do not use logical interface unit 0.

```
[edit vlans vlan-name]
user@switch# set interface interface-name.unit-number
```

NOTE: Note that you do not create a VLAN ID for the S-VLAN. The ID is created automatically for the appropriate logical interface.

2. Repeat step 1 for the other S-VLANs.
3. Enable the physical interface to transmit packets with two 802.1Q VLAN tags:

```
[edit interfaces interface-name]
user@switch# set flexible-vlan-tagging
```

4. Enable extended VLAN bridge encapsulation on the interface:

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

5. Enable the S-VLAN interface to send and receive untagged packets:

```
[edit interfaces interface-name]
user@switch# set native-vlan-id vlan-id
```

6. Bind one of the logical units of the interface to the VLAN ID for one of the S-VLANs.

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

7. Repeat step 6 to bind the automatically-created VLAN IDs for the other S-VLANs to the other logical units of the interface:

For example, the following configuration creates S-VLANs v10 and v30 and associates them with interface xe-0/0/0.10, enables Q-in-Q tunneling, enables xe-0/0/0 to accept untagged packets, and maps incoming C-VLAN packets to S-VLANs v10 and v30.

```
set vlans v10 interface xe-0/0/0.10
set vlans v30 interface xe-0/0/0.10
set interfaces xe-0/0/0 flexible-vlan-tagging
set interfaces xe-0/0/0 native-vlan-id 10
set interfaces xe-0/0/0 encapsulation extended-vlan-bridge
set interfaces xe-0/0/0 unit 10 vlan-id 10
set interfaces xe-0/0/0 unit 30 vlan-id 30
```

To configure the many-to-many bundling method on a C-VLAN interface, perform the following steps for each customer:

1. Assign a logical interface (unit) of one C-VLAN interface to be a member of one S-VLAN.

   ```
   [edit vlans vlan-name]
   user@switch# set interface interface-name.unit-number
   ```

2. Repeat step 1 to assign another C-VLAN interface (physical interface) to be a member of another S-VLAN.

3. Enable the interface to transmit packets with 802.1Q VLAN tags:

   ```
   [edit interfaces interface-name]
   user@switch# set flexible-vlan-tagging
   ```

4. Enable extended VLAN bridge encapsulation on the interface:

   ```
   [edit interfaces interface-name]
   user@switch# encapsulation extended-vlan-bridge
   ```

5. Enable the C-VLAN interface to send and receive untagged packets:

   ```
   [edit interfaces interface-name]
   user@switch# set native-vlan-id vlan-id
   ```

6. For each physical interface, configure a logical interface (unit) to receive and forward any tagged packet whose VLAN ID tag matches the list of VLAN IDs you specify:

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

   To configure only one C-VLAN to be mapped to an S-VLAN, specify only one VLAN ID after *vlan-id-list*.

   CAUTION: You can apply no more than eight VLAN identifier lists to a physical interface. This limitation does not apply to QFX10000 switches.

7. For each physical interface, configure the system to add an S-VLAN tag (outer tag) as packets travel from the C-VLAN interface to the S-VLAN:

```
[edit interfaces interface-name unit logical-unit-number]
user@switch# set input-vlan-map push
```

8. For each physical interface, configure the system to remove the S-VLAN tag when packets are forwarded from the S-VLAN interface to the C-VLAN interface:

```
[edit interfaces interface-name unit logical-unit-number]
user@switch# set output-vlan-map pop
```

For example, the following configuration makes xe-0/0/1.10 a member of S-VLAN v10, enables Q-in-Q tunneling, and maps packets from C-VLANs 10 through 20 to S-VLAN 10. The configuration for customer 2 makes xe-0/0/2.30 a member of S-VLAN v30, enables Q-in-Q tunneling, and maps packets from C-VLANs 30 through 40, 50 through 60, and 70 through 80 to S-VLAN 30. Both interfaces are configured to accept untagged packets.

If a packet originates in C-VLAN 10 and needs to be sent over the S-VLAN, a tag with a VLAN ID 10 is added to the packet. If a packet is forwarded internally from the S-VLAN interface to xe-0/0/1.10, the tag with VLAN ID 10 is removed. The same principles apply to the C-VLANs configured on interface xe-0/0/2.

NOTE: Notice that you can use the same tag value for an S-VLAN and C-VLAN. For example, the configuration for customer 1 maps C-VLAN ID 10 to S-VLAN ID 10. C-VLAN and S-VLAN tags use separate name spaces, so this configuration is allowed.

Configuration for customer 1:

```
set vlans v10 interface xe-0/0/1.10
set interfaces xe-0/0/1 flexible-vlan-tagging
set interfaces xe-0/0/1 encapsulation extended-vlan-bridge
set interfaces xe-0/0/1 unit 10 vlan-id-list 10-20
set interfaces xe-0/0/1 native-vlan-id 15
set interfaces xe-0/0/1 unit 10 input-vlan-map push
set interfaces xe-0/0/1 unit 10 output-vlan-map pop
```

Configuration for customer 2:

```
set vlans v30 interface xe-0/0/2.30
set interfaces xe-0/0/2 flexible-vlan-tagging
set interfaces xe-0/0/2 encapsulation extended-vlan-bridge
set interfaces xe-0/0/2 unit 30 vlan-id-list 30-40
set interfaces xe-0/0/2 unit 30 vlan-id-list 50-60
set interfaces xe-0/0/2 unit 30 vlan-id-list 70-80
set interfaces xe-0/0/2 native-vlan-id 75
set interfaces xe-0/0/2 unit 30 input-vlan-map push
set interfaces xe-0/0/2 unit 30 output-vlan-map pop
```

Configuring a Specific Interface Mapping with VLAN ID Translation Option

You can configure Q-in-Q tunneling by mapping packets from a specified C-VLAN to a specified S-VLAN. In addition, you can configure the system to replace a C-VLAN tag with an S-VLAN tag or replace an S-VLAN tag with a C-VLAN tag (instead of double tagging). This is call VLAN translation or VLAN rewriting. VLAN translation is particularly useful if a service provider's Layer 2 network that connects a customer's sites does not support double tagged packets.

When you use VLAN translation, both ends of the link normally must be able to swap the tags appropriately. That is, both ends of the link must be configured to swap the C-VLAN tag for the S-VLAN tag and swap the S-VLAN tag for the C-VLAN tag so that traffic in both directions is tagged appropriately while in transit and after arrival.

First configure the S-VLAN and its interface:

1. Assign a logical interface to be a member of the S-VLAN. Do not use unit 0.

```
[edit vlans vlan-name]
user@switch# set interface interface-name.unit-number
```

> **NOTE**: Note that you do not create a VLAN ID for the S-VLAN. The ID is created automatically for the appropriate logical interface.

2. Enable the interface to transmit packets with 802.1Q VLAN tags:

```
[edit interfaces interface-name]
user@switch# set flexible-vlan-tagging
```

3. Enable the S-VLAN interface to send and receive untagged packets:

```
[edit interfaces interface-name]
user@switch# set native-vlan-id vlan-id
```

4. Enable extended VLAN bridge encapsulation on the interface:

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

5. Bind the logical interface (unit) of the interface that you specified earlier to the VLAN ID for the S-VLAN:

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

For example, the following configuration creates S-VLAN v200, makes xe-0/0/0.200 a member of that VLAN, enables Q-in-Q tunneling on interface xe-0/0/0, enables xe-0/0/0 to accept untagged packets, and binds a logical interface of xe-0/0/0 to the VLAN ID of VLAN v200.

```
set vlans v200 interface xe-0/0/0.200
set interfaces xe-0/0/0 flexible-vlan-tagging
set interfaces xe-0/0/0 native-vlan-id 150
```

```
set interfaces xe-0/0/0 encapsulation extended-vlan-bridge
set interfaces xe-0/0/0 unit 200 vlan-id 200
```

Now configure a specific interface mapping with optional VLAN ID translation on the C-VLAN interface:

1. Assign a logical interface of the C-VLAN interface to be a member of the S-VLAN.

```
[edit vlans vlan-name]
user@switch# set interface interface-name.unit-number
```

2. Enable the interface to transmit packets with 802.1Q VLAN tags:

```
[edit interfaces interface-name]
user@switch# set flexible-vlan-tagging
```

3. Enable the C-VLAN interface to send and receive untagged packets:

```
[edit interfaces interface-name]
user@switch# set native-vlan-id vlan-id
```

4. Enable extended VLAN bridge encapsulation on the interface:

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

5. Configure a logical interface (unit) to receive and forward any tagged packet whose VLAN ID tag matches the VLAN IDs you specify:

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

6. Configure the system to remove the existing C-VLAN tag and replace it with the S-VLAN tag when packets ingress on the C-VLAN interface and are forwarded to the S-VLAN:

```
[edit interfaces interface-name unit logical-unit-number]
user@switch# set input-vlan-map swap
```

7. Configure the system to remove the existing S-VLAN tag and replace it with the C-VLAN tag when packets are forwarded from the S-VLAN interface to the C-VLAN interface:

```
[edit interfaces interface-name unit logical-unit-number]
user@switch# set output-vlan-map swap
```

8. To configure an S-VLAN and associate it with the appropriate C-VLAN interface:

```
[edit vlans vlan-name]
user@switch# set interface interface-name
```

For example, the following configuration on C-VLAN interface xe-0/0/1.200 enables Q-in-Q tunneling, enables xe-0/0/1 to accept untagged packets, and maps incoming packets from C-VLAN 150 to logical interface 200, which is a member of S-VLAN 200. Also, when packets egress from C-VLAN interface xe-0/0/1 and travel to the S-VLAN interface, the C-VLAN tag of 150 is removed and replaced with the S-VLAN tag of 200. When packets travel from the S-VLAN interface to the C-VLAN interface, the S-VLAN tag of 200 is removed and replaced with the C-VLAN tag of 150.

```
set vlans v200 interface xe-0/0/1.200
set interfaces xe-0/0/1 flexible-vlan-tagging
set interfaces xe-0/0/1 native-vlan-id 150
set interfaces xe-0/0/1 encapsulation extended-vlan-bridge
set interfaces xe-0/0/1 unit 200 vlan-id 200
set interfaces xe-0/0/1 unit 200 output-vlan-map swap
set interfaces xe-0/0/1 unit 200 input-vlan-map swap
```

Example: Setting Up Q-in-Q Tunneling on QFX Series Switches

IN THIS SECTION

Service providers can use Q-in-Q tunneling to transparently pass Layer 2 VLAN traffic between customer sites without removing or changing the customer VLAN tags or class-of-service (CoS) settings. Data centers can use Q-in-Q tunneling to isolate customer traffic within a single site or when customer traffic flows between cloud data centers in different geographic locations.

NOTE: This example uses a Junos OS release that does *not* support the Enhanced Layer 2 Software (ELS) configuration style. If your switch runs software that supports ELS, see Configuring Q-in-Q Tunneling on QFX Series, NFX Series, and EX4600 Switches with ELS Support.

This example describes how to set up Q-in-Q tunneling:

Requirements

This example requires one QFX Series device with Junos OS Release 12.1 or later.

Before you begin setting up Q-in-Q tunneling, make sure you have created and configured the necessary customer VLANs on the neighboring switches. See "Configuring VLANs on QFX Series Switches without Enhanced Layer 2 Support" on page 152.

Overview and Topology

In this service provider network, there are multiple customer VLANs mapped to one service VLAN.

Table 125 on page 978 lists the settings for the sample topology.

Table 125: Components of the Topology for Setting Up Q-in-Q Tunneling

Interface	Description
xe-0/0/11.0	Tagged S-VLAN trunk port
xe-0/0/12.0	Untagged customer-facing access port
xe-0/0/13.0	Untagged customer-facing access port
xe-0/0/14.0	Tagged S-VLAN trunk port

Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly create and configure Q-in-Q tunneling, copy the following commands and paste them into the switch terminal window:

```
[edit]
set vlans service-vlan vlan-id 1000
set vlans service-vlan dot1q-tunneling customer-vlans 1-100
set vlans service-vlan dot1q-tunneling customer-vlans 201-300
set interfaces xe-0/0/11 unit 0 family ethernet-switching port-mode trunk
set interfaces xe-0/0/11 unit 0 family ethernet-switching vlan members 1000
set interfaces xe-0/0/12 unit 0 family ethernet-switching port-mode access
set interfaces xe-0/0/12 unit 0 family ethernet-switching vlan members 1000
set interfaces xe-0/0/13 unit 0 family ethernet-switching port-mode access
set interfaces xe-0/0/13 unit 0 family ethernet-switching vlan members 1000
set interfaces xe-0/0/14 unit 0 family ethernet-switching port-mode trunk
set interfaces xe-0/0/14 unit 0 family ethernet-switching vlan members 1000
set ethernet-switching-options dot1q-tunneling ether-type 0x9100
```

Procedure

Step-by-Step Procedure

To configure Q-in-Q tunneling:

1. Set the VLAN ID for the S-VLAN:

```
[edit vlans]
user@switch# set service-vlan vlan-id 1000
```

2. Enable Q-in-Q tunneling and specify the customer VLAN ranges:

```
[edit vlans]
user@switch# set service-vlan dot1q-tunneling customer-vlans 1-100
user@switch# set service-vlan dot1q-tunneling customer-vlans 201-300
```

3. Set the port mode and VLAN information for the interfaces:

```
[edit interfaces]
user@switch# set xe-0/0/11 unit 0 family ethernet-switching port-mode trunk
user@switch# set xe-0/0/11 unit 0 family ethernet-switching vlan members 1000
user@switch# set xe-0/0/12 unit 0 family ethernet-switching port-mode access
user@switch# set xe-0/0/12 unit 0 family ethernet-switching vlan members 1000
user@switch# set xe-0/0/13 unit 0 family ethernet-switching port-mode access
user@switch# set xe-0/0/13 unit 0 family ethernet-switching vlan members 1000
user@switch# set xe-0/0/14 unit 0 family ethernet-switching port-mode trunk
user@switch# set xe-0/0/14 unit 0 family ethernet-switching vlan members 1000
```

4. Set the Q-in-Q Ethertype value (optional):

```
[edit]
user@switch# set ethernet-switching-options dot1q-tunneling ether-type 0x9100
```

Results

Check the results of the configuration:

```
user@switch> show configuration vlans service-vlan
vlan-id 1000 {
    dot1q-tunneling {
       customer-vlans [ 1-100 201-300 ];
}
user@switch> show configuration interfaces
xe-0/0/11 {
    unit 0 {
        family ethernet-switching {
            port-mode trunk;
            vlan members 1000;
        }
```

```
    }
}
xe-0/0/12 {
    unit 0 {
        family ethernet-switching {
            port-mode access;
            vlan members 1000;
        }
    }
}
xe-0/0/13 {
    unit 0 {
        family ethernet-switching {
            port-mode access;
            vlan members 1000;
        }
    }
}
xe-0/0/14 {
    unit 0 {
        family ethernet-switching {
            port-mode trunk;
            vlan members 1000;
        }
    }
}
user@switch> show ethernet-switching-options
dot1q-tunneling {
    ether-type 0x9100;
}
```

Verification

IN THIS SECTION

- Verifying That Q-in-Q Tunneling Was Enabled | 982

Confirm that the configuration is working properly.

Verifying That Q-in-Q Tunneling Was Enabled

Purpose

Verify that Q-in-Q tunneling was properly enabled.

Action

Use the `show vlans` command:

```
user@switch> show vlans service-vlan extensive
VLAN: service-vlan, Created at: Wed Mar 14 07:17:53 2012
802.1Q Tag: 1000, Internal index: 18, Admin State: Enabled, Origin: Static
Dot1q Tunneling Status: Enabled
Customer VLAN ranges:
                     1-100
                     201-300
Protocol: Port Mode
Number of interfaces: Tagged 2 (Active = 0), Untagged  2 (Active = 0)
      xe-0/0/11.0, tagged, trunk
      xe-0/0/14.0, tagged, trunk
      xe-0/0/12.0, untagged, access
      xe-0/0/13.0, untagged, access
```

Meaning

The output indicates that Q-in-Q tunneling is enabled and that the VLAN is tagged and shows the associated customer VLANs.

Example: Setting Up Q-in-Q Tunneling on EX Series Switches

IN THIS SECTION

Service providers can use Q-in-Q tunneling to transparently pass Layer 2 VLAN traffic from a customer site, through the service provider network, to another customer site without removing or changing the customer VLAN tags or class-of-service (CoS) settings. You can configure Q-in-Q tunneling on EX Series switches.

This example describes how to set up Q-in-Q:

Requirements

This example requires one EX Series switch with Junos OS Release 9.3 or later for EX Series switches.

Before you begin setting up Q-in-Q tunneling, make sure you have created and configured the necessary customer VLANs. See *Configuring VLANs for EX Series Switches* or Configuring VLANs for EX Series Switches (J-Web Procedure).

Overview and Topology

In this service provider network, there are multiple customer VLANs mapped to one service VLAN.

Table 126 on page 983 lists the settings for the example topology.

Table 126: Components of the Topology for Setting Up Q-in-Q Tunneling

Interface	Description
ge-0/0/11.0	Tagged S-VLAN trunk port
ge-0/0/12.0	Untagged customer-facing access port
ge-0/0/13.0	Untagged customer-facing access port
ge-0/0/14.0	Tagged S-VLAN trunk port

Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly create and configure Q-in-Q tunneling, copy the following commands and paste them into the switch terminal window:

```
[edit]
set vlans qinqvlan vlan-id 4001
set vlans qinqvlan dot1q-tunneling customer-vlans 1-100
set vlans qinqvlan dot1q-tunneling customer-vlans 201-300
set interfaces ge-0/0/11 unit 0 family ethernet-switching port-mode trunk
set interfaces ge-0/0/11 unit 0 family ethernet-switching vlan members 4001
set interfaces ge-0/0/12 unit 0 family ethernet-switching port-mode access
set interfaces ge-0/0/12 unit 0 family ethernet-switching vlan members 4001
set interfaces ge-0/0/13 unit 0 family ethernet-switching port-mode access
set interfaces ge-0/0/13 unit 0 family ethernet-switching vlan members 4001
set interfaces ge-0/0/14 unit 0 family ethernet-switching port-mode trunk
set interfaces ge-0/0/14 unit 0 family ethernet-switching vlan members 4001
set ethernet-switching-options dot1q-tunneling ether-type 0x9100
```

Procedure

Step-by-Step Procedure

To configure Q-in-Q tunneling:

1. Set the VLAN ID for the S-VLAN:

```
[edit vlans]
user@switch# set qinqvlan vlan-id 4001
```

2. Enable Q-in-Q tuennling and specify the customer VLAN ranges:

```
[edit vlans]
user@switch# set qinqvlan dot1q-tunneling customer-vlans 1-100
user@switch# set qinqvlan dot1q-tunneling customer-vlans 201-300
```

3. Set the port mode and VLAN information for the interfaces:

```
[edit interfaces]
user@switch# set ge-0/0/11 unit 0 family ethernet-switching port-mode trunk
user@switch# set ge-0/0/11 unit 0 family ethernet-switching vlan members 4001
user@switch# set ge-0/0/12 unit 0 family ethernet-switching port-mode access
user@switch# set ge-0/0/12 unit 0 family ethernet-switching vlan members 4001
user@switch# set ge-0/0/13 unit 0 family ethernet-switching port-mode access
user@switch# set ge-0/0/13 unit 0 family ethernet-switching vlan members 4001
user@switch# set ge-0/0/14 unit 0 family ethernet-switching port-mode trunk
user@switch# set ge-0/0/14 unit 0 family ethernet-switching vlan members 4001
```

4. Set the Q-in-Q Ethertype value:

```
[edit]
user@switch# set ethernet-switching-options dot1q-tunneling ether-type 0x9100
```

Results

Check the results of the configuration:

```
user@switch> show configuration vlans qinqvlan
vlan-id 4001 {
    dot1q-tunneling {
       customer-vlans [ 1-100 201-300 ];
}
user@switch> show configuration interfaces
ge-0/0/11 {
    unit 0 {
        family ethernet-switching {
            port-mode trunk;
            vlan members 4001;
        }
```

```
    }
}
ge-0/0/12 {
    unit 0 {
        family ethernet-switching {
            port-mode access;
            vlan members 4001;
        }
    }
}
ge-0/0/13 {
    unit 0 {
        family ethernet-switching {
            port-mode access;
            vlan members 4001;
        }
    }
}
ge-0/0/14 {
    unit 0 {
        family ethernet-switching {
            port-mode trunk;
            vlan members 4001;
        }
    }
}
user@switch> show ethernet-switching-options
dot1q-tunneling {
    ether-type 0x9100;
}
```

Verification

IN THIS SECTION

- Verifying That Q-in-Q Tunneling Was Enabled | 987

To confirm that the configuration is working properly, perform these tasks:

Verifying That Q-in-Q Tunneling Was Enabled

Purpose

Verify that Q-in-Q tunneling was properly enabled on the switch.

Action

Use the `show vlans` command:

```
user@switch> show vlans qinqvlan extensive
VLAN: qinqvlan, Created at: Thu Sep 18 07:17:53 2008
802.1Q Tag: 4001, Internal index: 18, Admin State: Enabled, Origin: Static
Dot1q Tunneling Status: Enabled
Customer VLAN ranges:
                     1-100
                     201-300
Protocol: Port Mode
Number of interfaces: Tagged 2 (Active = 0), Untagged  4 (Active = 0)
      ge-0/0/11.0, tagged, trunk
      ge-0/0/14.0, tagged, trunk
      ge-0/0/12.0, untagged, access
      ge-0/0/13.0, untagged, access
```

Meaning

The output indicates that Q-in-Q tunneling is enabled and that the VLAN is tagged and shows the associated customer VLANs.

Setting Up a Dual VLAN Tag Translation Configuration on QFX Switches

Starting with Junos OS Release 14.1X53-D40, you can use the dual VLAN tag translation (also known as dual VLAN tag rewrite) feature to deploy switches in service-provider domains, allowing dual-tagged, single-tagged, and untagged VLAN packets to come into or exit from the switch.

The following example configuration shows use of the swap-swap, pop-swap, and swap-push dual tag operations.

```
[edit]
set interfaces ge-0/0/1 unit 503 description UNI-3
set interfaces ge-0/0/1 unit 503 encapsulation vlan-bridge
set interfaces ge-0/0/1 unit 503 vlan-tags outer 503
set interfaces ge-0/0/1 unit 503 vlan-tags inner 504
set interfaces ge-0/0/1 unit 503 input-vlan-map swap-swap
set interfaces ge-0/0/1 unit 503 input-vlan-map vlan-id 500
set interfaces ge-0/0/1 unit 503 input-vlan-map inner-vlan-id 514
set interfaces ge-0/0/1 unit 503 output-vlan-map swap-swap
set interfaces ge-0/0/0 description NNI
set interfaces ge-0/0/0 flexible-vlan-tagging
set interfaces ge-0/0/0 encapsulation flexible-ethernet-services
set interfaces ge-0/0/0 unit 500 description "SVLAN500 port"
set interfaces ge-0/0/0 unit 500 encapsulation vlan-bridge
set interfaces ge-0/0/0 unit 500 vlan-id 500
set interfaces ge-0/0/0 unit 600 description "SVLAN600 port"
set interfaces ge-0/0/0 unit 600 encapsulation vlan-bridge
set interfaces ge-0/0/0 unit 600 vlan-id 600
set interfaces ge-0/0/0 unit 700 description "SVLAN700 port"
set interfaces ge-0/0/0 unit 700 encapsulation vlan-bridge
set interfaces ge-0/0/0 unit 700 vlan-id 700
set interfaces ge-0/0/0 unit 0 family ethernet-switching interface-mode trunk
set interfaces ge-0/0/0 unit 0 family ethernet-switching vlan members v1000
set interfaces ge-0/0/0 unit 1100 description UNI-SVLAN1100
set interfaces ge-0/0/0 unit 1100 encapsulation vlan-bridge
set interfaces ge-0/0/0 unit 1100 vlan-tags outer 1101
set interfaces ge-0/0/0 unit 1100 vlan-tags inner 1102
set interfaces ge-0/0/0 unit 1100 input-vlan-map swap-swap
set interfaces ge-0/0/0 unit 1100 input-vlan-map vlan-id 1100
set interfaces ge-0/0/0 unit 1100 input-vlan-map inner-vlan-id 2101
set interfaces ge-0/0/0 unit 1100 output-vlan-map swap-swap
set interfaces ge-0/0/0 unit 1200 description UNI-SVLAN1200
set interfaces ge-0/0/0 unit 1200 encapsulation vlan-bridge
set interfaces ge-0/0/0 unit 1200 vlan-id 1201
set interfaces ge-0/0/0 unit 1200 input-vlan-map swap-push
set interfaces ge-0/0/0 unit 1200 input-vlan-map inner-vlan-id 2200
set interfaces ge-0/0/0 unit 1200 output-vlan-map pop-swap
set interfaces ge-0/0/2 description UNI
set interfaces ge-0/0/2 flexible-vlan-tagging
```

```
set interfaces ge-0/0/2 encapsulation flexible-ethernet-services
set interfaces ge-0/0/2 unit 0 family ethernet-switching interface-mode trunk
set interfaces ge-0/0/2 unit 0 family ethernet-switching vlan members v1000
set interfaces ge-0/0/2 unit 603 description UNI-3
set interfaces ge-0/0/2 unit 603 encapsulation vlan-bridge
set interfaces ge-0/0/2 unit 603 vlan-tags outer 603
set interfaces ge-0/0/2 unit 603 vlan-tags inner 604
set interfaces ge-0/0/2 unit 603 input-vlan-map swap-swap
set interfaces ge-0/0/2 unit 603 input-vlan-map vlan-id 600
set interfaces ge-0/0/2 unit 603 input-vlan-map inner-vlan-id 614
set interfaces ge-0/0/2 unit 603 output-vlan-map swap-swap
set interfaces ge-0/0/3 description UNI
set interfaces ge-0/0/3 flexible-vlan-tagging
set interfaces ge-0/0/3 encapsulation flexible-ethernet-services
set interfaces ge-0/0/3 unit 0 family ethernet-switching interface-mode trunk
set interfaces ge-0/0/3 unit 0 family ethernet-switching vlan members v1000
set interfaces ge-0/0/3 unit 703 description UNI-3
set interfaces ge-0/0/3 unit 703 encapsulation vlan-bridge
set interfaces ge-0/0/3 unit 703 vlan-tags outer 703
set interfaces ge-0/0/3 unit 703 vlan-tags inner 704
set interfaces ge-0/0/3 unit 703 input-vlan-map swap-swap
set interfaces ge-0/0/3 unit 703 input-vlan-map vlan-id 700
set interfaces ge-0/0/3 unit 703 input-vlan-map inner-vlan-id 714
set interfaces ge-0/0/3 unit 703 output-vlan-map swap-swap
set interfaces ge-0/0/3 unit 701 encapsulation vlan-bridge
set interfaces ge-0/0/3 unit 701 vlan-id 701
set interfaces ge-0/0/3 unit 701 input-vlan-map swap-push
set interfaces ge-0/0/3 unit 701 input-vlan-map inner-vlan-id 780
set interfaces ge-0/0/3 unit 701 output-vlan-map pop-swap
set interfaces ge-0/0/3 unit 1100 description SVLAN1100-NNI
set interfaces ge-0/0/3 unit 1100 encapsulation vlan-bridge
set interfaces ge-0/0/3 unit 1100 vlan-id 1100
set interfaces ge-0/0/3 unit 1200 description SVLAN1200-NNI
set interfaces ge-0/0/3 unit 1200 encapsulation vlan-bridge
set interfaces ge-0/0/3 unit 1200 vlan-id 1200
set vlans SVLAN500 interface ge-0/0/0.500
set vlans SVLAN500 interface ge-0/0/1.503
set vlans SVLAN600 interface ge-0/0/0.600
set vlans SVLAN600 interface ge-0/0/2.603
set vlans SVLAN600 interface ge-0/0/3.701
set vlans SVLAN700 interface ge-0/0/0.700
set vlans SVLAN700 interface ge-0/0/3.703
set vlans v1000 vlan-id 1000
```

```
set vlans SVLAN1100 interface ge-0/0/0.1100
set vlans SVLAN1100 interface ge-0/0/3.1100
set vlans SVLAN1200 interface ge-0/0/3.1200
set vlans SVLAN1200 interface ge-0/0/0.1200
```

NOTE: Dual VLAN tagging (`vlan-tags outer`, `vlan-tags inner`) is not supported on QFX 5000 switches with Layer 3 IFL `family inet`.

```
set interfaces xe-0/0/2 unit 100 vlan-tags outer 1000
set interfaces xe-0/0/2 unit 100 vlan-tags inner 100
```

Support for Swap-Push/Pop-Swap for QFX and EX Switches

Q-in-Q tunneling with L2 swap-push/pop-swap support is a specific scenario in which the customer VLAN (C-VLAN) tag is swapped with the `inner-vlan-id` tag, and the service-provider-defined service VLAN (S-VLAN) tag is pushed on it (for traffic flowing from customer to service provider site). This traffic is sent to the service provider network double-tagged (S-VLAN + C-VLAN). For the traffic flowing from the service provider network to the customer network, the S-VLAN tag is removed, and the C-VLAN tag is replaced with the VLAN ID configured on the UNI logical interface.

The following example shows the swap-push/pop-swap dual tag operations.

1. Swap-push—For incoming-single tagged frame from UNI, the C-VLAN (VLAN ID 100) swaps with the configured inner-vlan-id (200) on logical interface and the S-VLAN (VLAN ID 900) pushes on to the frame. The double-tagged frame egresses out of the NNI.
2. Pop-swap—For incoming double-tagged frame from the NNI, the S-VLAN tag pops (VLAN ID 900) from the frame and the logical interface's VLAN ID 100 replaces the C-VLAN tag. The single-tagged frame egresses out of the UNI.

```
set interfaces ge-0/0/1 description UNI
set interfaces ge-0/0/1 flexible-vlan-tagging
set interfaces ge-0/0/1 encapsulation flexible-ethernet-services
set interfaces ge-0/0/1 unit 100 encapsulation vlan-bridge
set interfaces ge-0/0/1 unit 100 vlan-id 100
set interfaces ge-0/0/1 unit 100 input-vlan-map swap-push
set interfaces ge-0/0/1 unit 100 input-vlan-map vlan-id 900
set interfaces ge-0/0/1 unit 100 input-vlan-map inner-vlan-id 200
set interfaces ge-0/0/1 unit 100 output-vlan-map pop-swap

set interfaces ge-0/0/2 description NNI
```

```
set interfaces ge-0/0/2 flexible-vlan-tagging
set interfaces ge-0/0/2 encapsulation flexible-ethernet-services
set interfaces ge-0/0/2 unit 900 encapsulation vlan-bridge
set interfaces ge-0/0/2 unit 900 vlan-id 900

set vlans vlan-900 interface ge-0/0/1.100
set vlans vlan-900 interface ge-0/0/2.900
```

If you configure the logical interface with a VLAN ID list and the input-vlan-map and output-vlan-map is configured as swap-push/pop-swap, it results in undesired behavior as the traffic regressing out of the UNI has a logical unit number instead of the original customer VLAN ID from VLAN ID list configured.

Verifying That Q-in-Q Tunneling Is Working on Switches

IN THIS SECTION

Purpose

After creating a Q-in-Q VLAN, verify that it is set up properly.

Action

1. Use the `show configuration vlans` command to determine if you successfully created the primary and secondary VLAN configurations:

```
user@switch> show configuration vlans
svlan {
   vlan-id 300;
   dot1q-tunneling {
      customer-vlans [ 101-200 ];
```

```
    }
}
```

2. Use the `show vlans` command to view VLAN information and link status:

```
user@switch> show vlans s-vlan-name extensive
VLAN: svlan, Created at: Thu Oct 23 16:53:20 2008
802.1Q Tag: 300, Internal index: 2, Admin State: Enabled, Origin: Static
Dot1q Tunneling Status: Enabled
Customer VLAN ranges:
                    101-200
Protocol: Port Mode
Number of interfaces: Tagged 1 (Active = 0), Untagged  1 (Active = 0)
      xe-0/0/1, tagged, trunk
      xe-0/0/2, untagged, access
```

Meaning

The output confirms that Q-in-Q tunnling is enabled and that the VLAN is tagged, and lists the customer VLANs that are associated with the tagged VLAN.

Change History Table

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

Release	Description
19.4R1	Starting in Junos OS Release 19.4R1, the QFX10000 line of switches support the third and fourth Q-in-Q tags as payload (also known as a pass-through tag) along with the existing two tags (for VLAN matching and operations).
14.1X53-D40	Starting with Junos OS Release 14.1X53-D40, you can use the dual VLAN tag translation (also known as dual VLAN tag rewrite) feature to deploy switches in service-provider domains, allowing dual-tagged, single-tagged, and untagged VLAN packets to come into or exit from the switch.
14.1X53-D30	Starting with Junos OS 14.1X53-D30, you can configure the same interface to be an S-VLAN/NNI interface and a C-VLAN/UNI interface.

30

CHAPTER

Configuring Redundant Trunk Groups

IN THIS CHAPTER

Redundant Trunk Groups

IN THIS SECTION

Understanding Redundant Trunk Links (Legacy RTG Configuration)

In a typical enterprise network composed of distribution and access layers, a redundant trunk link provides a simple solution for network recovery when a trunk port on a switch goes down. In that case, traffic is routed to another trunk port, keeping network convergence time to a minimum.

NOTE: A Redundant Trunk Group can also have Link Aggregation Groups (LAGs)/ Aggregate Ethernet (AE) interfaces as links. For information on redundant trunk link configurations that include Q-in-Q support and use LAGs with link protection, see "Q-in-Q Support on Redundant Trunk Links Using LAGs with Link Protection" on page 1014.

To configure a redundant trunk link, create a redundant trunk group. The redundant trunk group is configured on the access switch and contains two links: a primary or active link, and a secondary link. If the active link fails, the secondary link automatically starts forwarding data traffic without waiting for normal spanning-tree protocol convergence.

Data traffic is forwarded only on the active link. Data traffic on the secondary link is dropped and shown as dropped packets when you issue the operational mode command `show interfaces` *`interface-name`* `extensive`.

While data traffic is blocked on the secondary link, Layer 2 control traffic is still permitted. For example, an LLDP session can be run between two switches on the secondary link.

Rapid Spanning Tree Protocol (RSTP) is enabled by default on the switches to create a loop-free topology, but an interface is not allowed to be in both a redundant trunk group and in a spanning-tree protocol topology at the same time. You must disable RSTP on an interface if a redundant trunk group is configured on that interface. For example, in Figure 47 on page 995, in addition to disabling RSTP on

the Switch 3 interfaces, you must also disable RSTP on the Switch 1 and Switch 2 interfaces connected to Switch 3. Spanning-tree protocols can, however, continue operating on other interfaces on those switches—for example on the link between Switch 1 and Switch 2.

Figure 47 on page 995 shows three switches in a basic topology for redundant trunk links. Switch 1 and Switch 2 make up the distribution layer, and Switch 3 makes up the access layer. Switch 3 is connected to the distribution layer through trunk ports ge-0/0/9.0 (Link 1) and ge-0/0/10.0 (Link 2). Link 1 and Link 2 are in a redundant trunk group called group1. Link 1 is designated as the primary link. Traffic flows between Switch 3 in the access layer and Switch 1 in the distribution layer through Link 1. While Link 1 is active, Link 2 blocks traffic.

Figure 47: Redundant Trunk Group, Link 1 Active

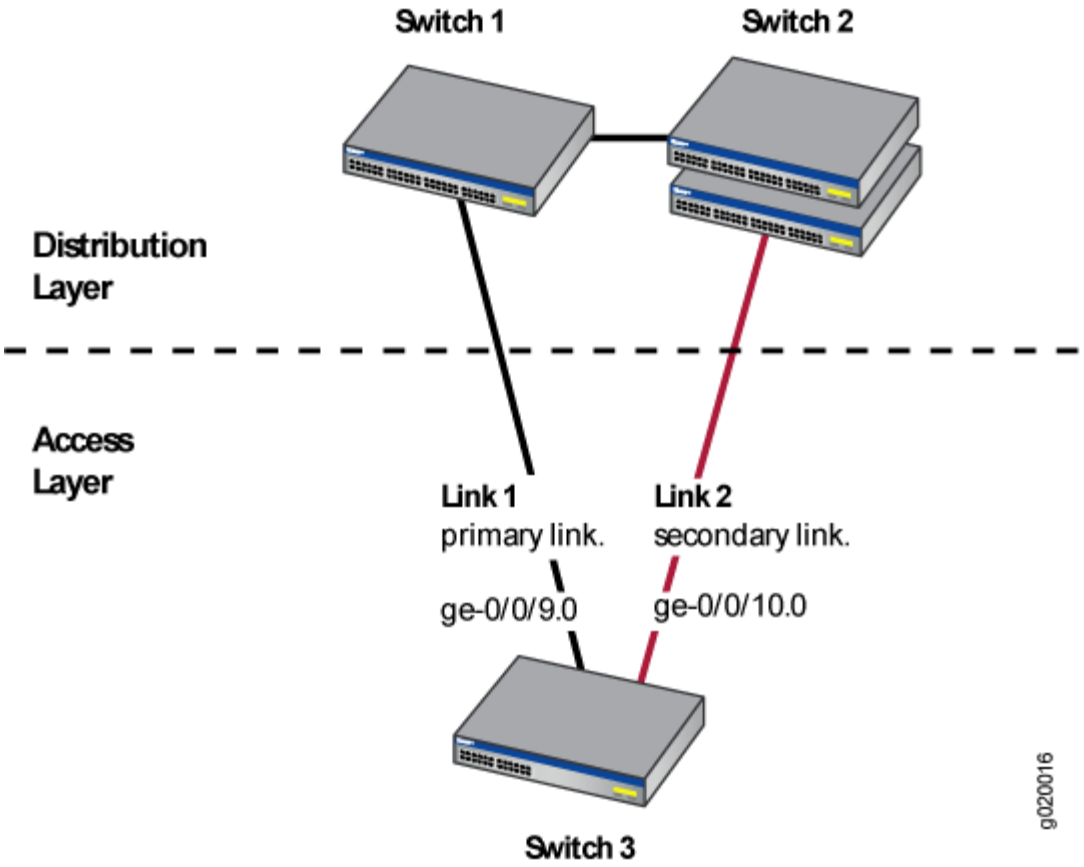
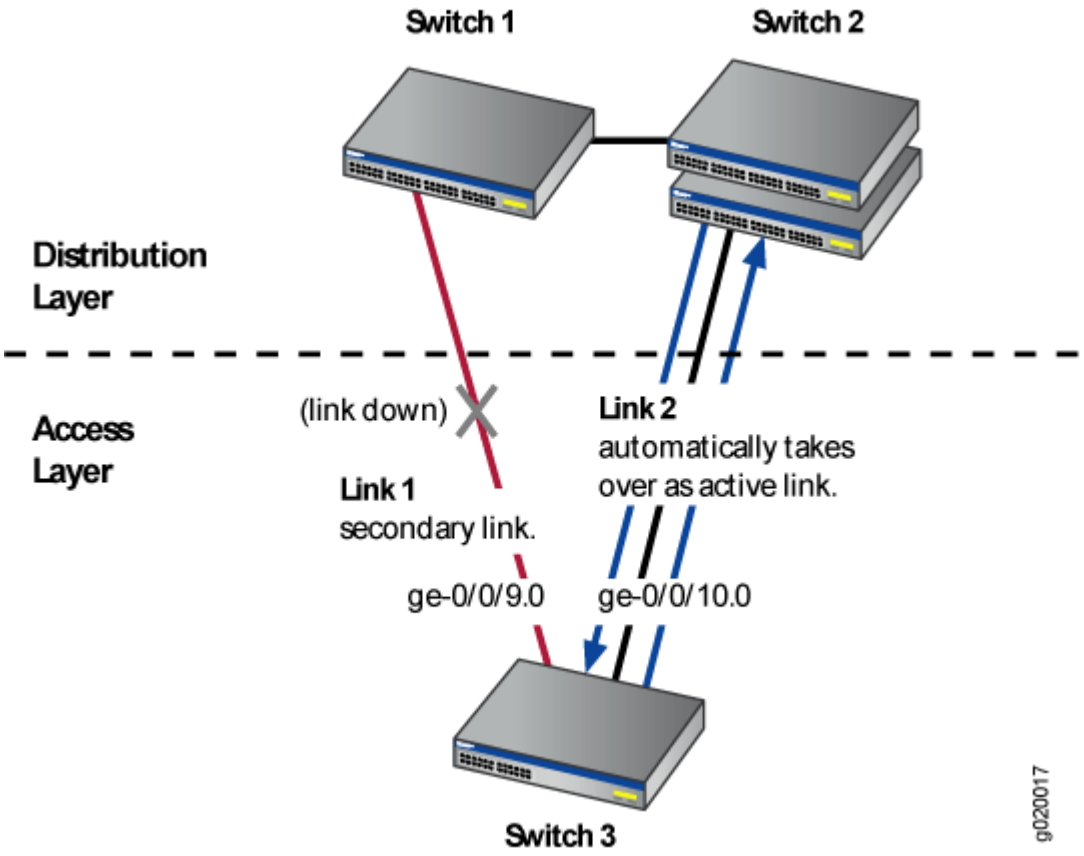


Figure 48 on page 996 illustrates how the redundant trunk link topology works when the primary link goes down.

Figure 48: Redundant Trunk Group, Link 2 Active



When Link 1 between Switch 1 and Switch 3 goes down, Link 2 takes over as the active link. Traffic between the access layer and the distribution layer is then automatically switched to Link 2 between Switch 3 and Switch 2.

Configuring Redundant Trunk Links for Faster Recovery on EX Series Switches

You can manage network convergence by configuring both a primary link and a secondary link on an EX Series switch; this is called a redundant trunk group (RTG). If the primary link in a redundant trunk group fails, it passes its known MAC address locations to the secondary link, which automatically takes over. You can configure a maximum of 16 redundant trunk groups on most standalone switches or on Virtual Chassis. The EX8200 switch and EX8200 Virtual Chassis, however, support up to 254 redundant trunk groups.

Generally, you configure a redundant trunk group by configuring one primary link (and its interface) and one unspecified link (and its interface) to serve as the secondary link. A second type of redundant trunk

group, not shown in the procedure in this topic, consists of two unspecified links (and their interfaces); in this case, neither of the links is primary. In this second case, the software selects an active link by comparing the port numbers of the two links and activating the link with the higher port number. The procedure given here describes configuring a primary/unspecified configuration for a redundant trunk group because that configuration gives you more control and is more commonly used.

Rapid Spanning Tree Protocol (RSTP) is enabled by default on EX Series switches to create a loop-free topology, but an interface is not allowed to be in both a redundant trunk group and in a spanning-tree protocol topology at the same time.

A primary link takes over whenever it is able. You can, however, alter the number of seconds that the primary link waits before reestablishing control by configuring the primary link's preempt cutover timer.

Before you configure the redundant trunk group on the switch, be sure you have:

- Disabled RSTP on all switches that will be linked to your redundant trunk group.
- Configured at least two interfaces with their port mode set to **trunk**; be sure that these two interfaces are not part of any existing RTG. See Configuring Gigabit Ethernet Interfaces (CLI Procedure) .

To configure a redundant trunk group on a switch:

1. Turn off RSTP:

```
[edit]
user@switch# set protocols rstp disable
```

2. Name the redundant trunk group while configuring one primary and one unspecified trunk interface:

```
[edit ethernet-switching-options]
user@switch# set redundant-trunk-group group name interface  interface-name primary
user@switch# set redundant-trunk-group group name interface interface-name
```

3. (Optional) Change the length of time (from the default of 1 second) that a re-enabled primary link waits to take over from an active secondary link:

```
[edit ethernet-switching-options]
set redundant-trunk-group group name preempt-cutover-timer seconds
```

Example: Configuring Redundant Trunk Links for Faster Recovery on Devices with ELS Support

IN THIS SECTION

NOTE: This example uses Junos OS for EX Series switches or QFX Series with support for the Enhanced Layer 2 Software (ELS) configuration style.. For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15.

You can manage network convergence by configuring both a primary link and a secondary link on a switch; this is called a redundant trunk group (RTG). If the primary link in a redundant trunk group fails, it passes its known MAC address locations to the secondary link, which automatically takes over after one minute.

This example describes how to create a redundant trunk group with a primary and a secondary link:

Requirements

This example uses the following hardware and software components:

- Two EX Series or QFX Series distribution switches
- One EX Series or QFX Series access switch
- The appropriate software release for your platform:
  - For EX Series switches: Junos OS Release 13.2X50-D10 or later
  - For the QFX Series: Junos OS Release 13.2X50-D15 or later

Before you configure the redundant trunk links network on the access and distribution switches, be sure you have:

- Configured interfaces ge-0/0/9 and ge-0/0/10 on the access switch, Switch 3, as trunk interfaces.

- Configured one trunk interface on each distribution switch, Switch 1 and Switch 2.
- Connected the three switches as shown in the topology for this example (see Figure 49 on page 1001).

Overview and Topology

IN THIS SECTION

In a typical enterprise network composed of distribution and access layers, a redundant trunk link provides a simple solution for trunk interface network recovery. When a trunk interface fails, data traffic is routed to another trunk interface after one second, thereby keeping network convergence time to a minimum.

This example shows the configuration of a redundant trunk group that includes one primary link (and its interface) and one unspecified link (and its interface) that serves as the secondary link.

A second type of redundant trunk group, not illustrated in the example, consists of two unspecified links (and their interfaces); in this case, neither of the links is primary. The software selects an active link by comparing the port numbers of the two links and activating the link with the higher port number. For example, if the two link interfaces use interfaces ge-0/1/0 and ge-0/1/1, the software activates ge-0/1/1. (In the interface names, the final number is the port number.)

Topology

The two links in a redundant trunk group generally operate the same way, whether they are configured as primary/unspecified or unspecified/unspecified. Data traffic initially passes through the active link but is blocked on the inactive link. While data traffic is blocked on the secondary link, note that Layer 2 control traffic is still permitted if the link is active. For example, an LLDP session can be run between two switches on the secondary link. If the active link either goes down or is disabled administratively, MAC refresh packets (for the MAC addresses that are learnt on the other ports of the VLAN) are sent over the new active link, so that the peer device connected to this new active link learns these MAC addresses and starts forwarding traffic towards the new active link.

The one difference in operation between the two types of redundant trunk groups occurs when a primary link is active, goes down, is replaced by the secondary link, and then reactivates. When a primary link is re-enabled while the secondary link is active, the primary link waits 1 second (you can change the time interval by using the preempt cutover timer to accommodate your network) and then takes over as the active link. In other words, the primary link has priority and is always activated if it is

available. This differs from the behavior of two unspecified links, both of which act as equals. Because the unspecified links are equal, the active link remains active until it either goes down or is disabled administratively.

The example given here illustrates a primary/unspecified configuration for a redundant trunk group because that configuration gives you more control and is more commonly used.

NOTE: Rapid Spanning Tree Protocol (RSTP) is enabled by default on the switches to create a loop-free topology, but an interface is not allowed to be in both a redundant trunk group and in a spanning-tree protocol topology at the same time. You will need to disable RSTP on the two distribution switches in the example, Switch 1 and Switch 2. Spanning-tree protocols can, however, continue operating in other parts of the network —for example, between the distribution switches and also in links between distribution switches and the enterprise core.

Figure 49 on page 1001 displays an example topology containing three switches. Switch 1 and Switch 2 make up the distribution layer, and Switch 3 makes up the access layer. Switch 3 is connected to the distribution layer through trunk interfaces ge-0/0/9.0 (Link 1) and ge-0/0/10.0 (Link 2).

Table 127 on page 1001 lists the components used in this redundant trunk group.

Because RSTP and RTGs cannot operate simultaneously on a switch, you disable RSTP on Switch 1 and Switch 2 in the first configuration task, and you disable RSTP on Switch 3 in the second task.

The second configuration task creates a redundant trunk group called example 1 on Switch 3. The trunk interfaces ge-0/0/9.0 and ge-0/0/10.0 are the two links configured in the second configuration task. You configure the trunk interface ge-0/0/9.0 as the primary link. You configure the trunk interface ge-0/0/10.0 as an unspecified link, which becomes the secondary link by default.

Figure 49: Topology for Configuring the Redundant Trunk Links

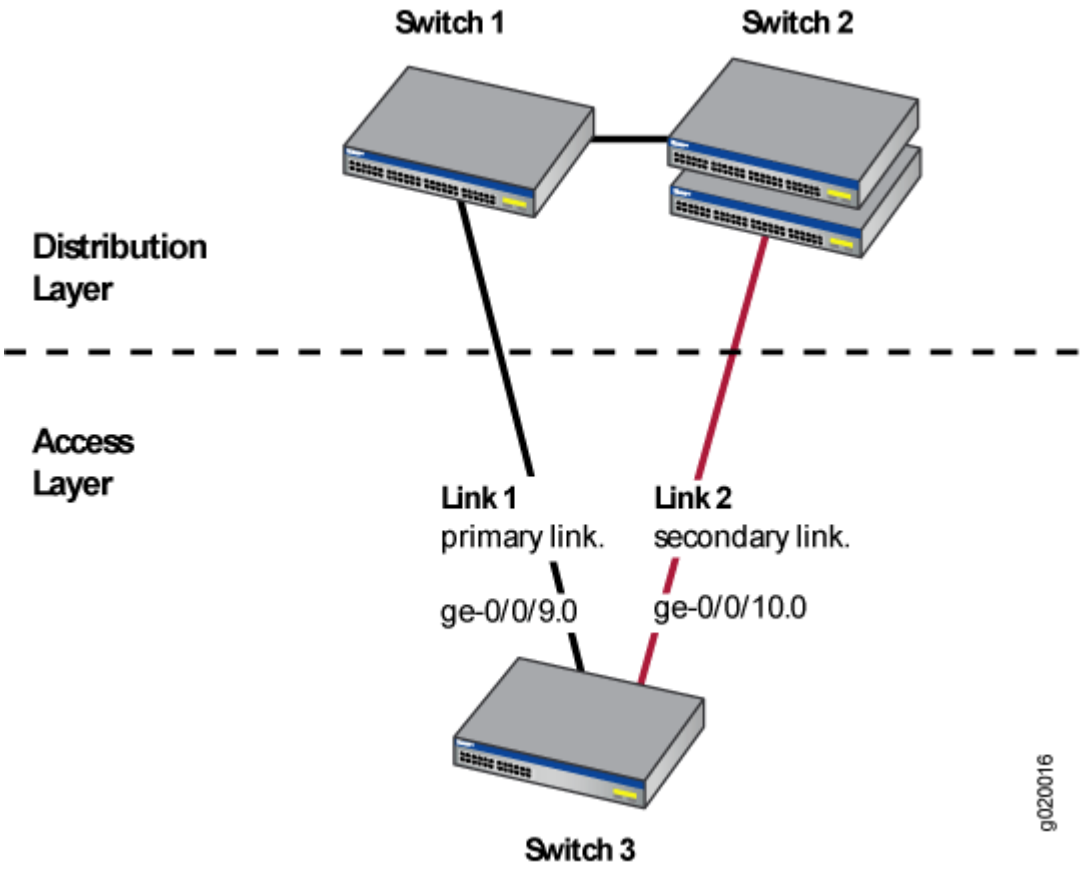


Table 127: Components of the Redundant Trunk Link Topology

Property	Settings
Switch hardware	• Switch 1–1 EX Series or QFX Series distribution switch • Switch 2–1 EX Series or QFX Series distribution switch • Switch 3–1 EX Series or QFX Series access switch
Trunk interfaces	On Switch 3 (access switch): ge-0/0/9.0 and ge-0/0/10.0
Redundant trunk group	rtg0

Disabling RSTP on Switches 1 and 2

IN THIS SECTION

To disable RSTP on Switch 1 and Switch 2, perform this task on each switch:

Procedure

CLI Quick Configuration

To quickly disable RSTP on Switch 1 and Switch 2, copy the following command and paste it into each switch terminal window:

```
[edit]
set protocols rstp disable
```

Step-by-Step Procedure

To disable RSTP on Switch 1 and Switch 2:

1. Disable RSTP on Switch 1 and Switch 2:

```
[edit]
user@switch# set protocols rstp disable
```

Results

Check the results of the configuration:

```
[edit]
user@switch# show
protocols {
    rstp {
```

```
        disable;
    }
}
```

Results

Configuring Redundant Trunk Links on Switch 3

IN THIS SECTION

- Procedure | 1003

To configure redundant trunk links on Switch 3, perform this task:

Procedure

CLI Quick Configuration

To quickly configure the redundant trunk group rtg0 on Switch 3, copy the following commands and paste them into the switch terminal window:

```
[edit]
 set protocols rstp disable
 set switch-options redundant-trunk-group group rtg0 interface ge-0/0/9.0 primary
 set switch-options redundant-trunk-group group rtg0 interface ge-0/0/10.0
 set redundant-trunk-group group rtg0 preempt-cutover-timer 60
```

Step-by-Step Procedure

Configure the redundant trunk group rtg0 on Switch 3.

1. Turn off RSTP:

```
[edit]
user@switch# set protocols rstp disable
```

2. Name the redundant trunk group rtg0 while configuring trunk interface ge-0/0/9.0 as the primary link and ge-0/0/10 as an unspecified link to serve as the secondary link:

```
[edit switch-options]
user@switch#  set redundant-trunk-group group rtg0 interface ge-0/0/9.0 primary
user@switch#  set redundant-trunk-group group rtg0 interface ge-0/0/10.0
```

3. (Optional) Change the time interval (from the default of 1 second) that a re-enabled primary link waits to take over for an active secondary link:

```
[edit switch-options]
user@switch#  set switch-options redundant-trunk-group group rtg0 preempt-cutover-timer 60
```

Results

Check the results of the configuration:

```
[edit]
user@switch# show
switch-options
    redundant-trunk-group {
        group rtg0 {
            preempt-cutover-timer 60;

            interface ge-0/0/9.0 {
                primary;
            }
            interface ge-0/0/10.0;
        }
    }
protocols {
rstp {



        disable;
    }
}
```

Verification

IN THIS SECTION

To confirm that the configuration is set up correctly, perform this task:

Verifying That a Redundant Trunk Group Was Created

Purpose

Verify that the redundant trunk group rtg0 has been created on Switch 1 and that trunk interfaces are members of the redundant trunk group.

Action

List all redundant trunk groups configured on the switch:

```
user@switch> show redundant-trunk-group
Group      Interface   State       Time of last flap          Flap
name                                                          count

rtg0       ge-0/0/9.0  Up/Pri      Never                        0
           ge-0/0/10.0 Up          Never                        0
```

Meaning

The `show redundant-trunk-group` command lists all redundant trunk groups configured on the switch as well as the interface names and their current states (up or down for an unspecified link, and up or down and primary for a primary link). For this configuration example, the output shows that the redundant trunk group rtg0 is configured on the switch. The `Up` beside the interfaces indicates that both link cables are physically connected. The `Pri` beside trunk interface ge-0/0/9.0 indicates that it is configured as the primary link.

Example: Configuring Redundant Trunk Links for Faster Recovery on EX Series Switches

IN THIS SECTION

NOTE: This example uses Junos OS for EX Series switches that does not support the Enhanced Layer 2 Software (ELS) configuration style. If your switch runs software that supports ELS, see "Example: Configuring Redundant Trunk Links for Faster Recovery on Devices with ELS Support" on page 998. For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15.

You can manage network convergence by configuring both a primary link and a secondary link on a switch; this is called a redundant trunk group (RTG). If the primary link in a redundant trunk group fails, it passes its known MAC address locations to the secondary link, which automatically takes over after one minute.

This example describes how to create a redundant trunk group with a primary and a secondary link:

Requirements

This example uses the following hardware and software components:

- Two EX Series distribution switches
- One EX Series access switch
- Junos OS Release 10.4 or later for EX Series switches

Before you configure the redundant trunk links network on the access and distribution switches, be sure you have:

- Configured interfaces **ge-0/0/9** and **ge-0/0/10** on the access switch, Switch 3, as trunk interfaces. See Configuring Gigabit Ethernet Interfaces (CLI Procedure).

- Configured one trunk interface on each distribution switch, Switch 1 and Switch 2.
- Connected the three switches as shown in the topology for this example (see Figure 50 on page 1009).

Overview and Topology

IN THIS SECTION

In a typical enterprise network composed of distribution and access layers, a redundant trunk link provides a simple solution for trunk interface network recovery. When a trunk interface fails, data traffic is routed to another trunk interface after one second, thereby keeping network convergence time to a minimum.

This example shows the configuration of a redundant trunk group that includes one primary link (and its interface) and one unspecified link (and its interface) that serves as the secondary link.

A second type of redundant trunk group, not illustrated in the example, consists of two unspecified links (and their interfaces); in this case, neither of the links is primary. In this second case, the software selects an active link by comparing the port numbers of the two links and activating the link with the higher port number. For example, if the two link interfaces use interfaces **ge-0/1/0** and **ge-0/1/1**, the software activates **ge-0/1/1.** (In the interface names, the final number is the port number.)

Topology

The two links in a redundant trunk group generally operate the same way, whether they are configured as primary/unspecified or unspecified/unspecified. Data traffic initially passes through the active link but is blocked on the inactive link. While data traffic is blocked on the secondary link, note that Layer 2 control traffic is still permitted if the link is active. For example, an LLDP session can be run between two switches on the secondary link. If the active link either goes down or is disabled administratively, MAC refresh packets (for the MAC addresses that are learnt on the other ports of the VLAN) are sent over the new active link, so that the peer device connected to this new active link learns these MAC addresses and starts forwarding traffic towards the new active link.

The one difference in operation between the two types of redundant trunk groups occurs when a primary link is active, goes down, is replaced by the secondary link, and then reactivates. When a primary link is re-enabled while the secondary link is active, the primary link waits 1 second (you can change the length of time using the preempt cutover timer to accommodate your network) and then takes over as the active link. In other words, the primary link has priority and is always activated if it is

available. This differs from the behavior of two unspecified links, which act as equals. Because the unspecified links are equal, the active link remains active until it either goes down or is disabled administratively.

The example given here illustrates a primary/unspecified configuration for a redundant trunk group because that configuration gives you more control and is more commonly used.

NOTE: Rapid Spanning Tree Protocol (RSTP) is enabled by default on EX Series switches to create a loop-free topology, but an interface is not allowed to be in both a redundant trunk group and in a spanning-tree protocol topology at the same time. You will need to disable RSTP on the two distribution switches in the example, Switch 1 and Switch 2. Spanning-tree protocols can, however, continue operating in other parts of the network —for example, between the distribution switches and also in links between distribution switches and the enterprise core.

Figure 50 on page 1009 displays an example topology containing three switches. Switch 1 and Switch 2 make up the distribution layer, and Switch 3 makes up the access layer. Switch 3 is connected to the distribution layer through trunk interfaces **ge-0/0/9.0** (Link 1) and **ge-0/0/10.0** (Link 2).

Table 128 on page 1009 lists the components used in this redundant trunk group.

Because RSTP and RTGs cannot operate simultaneously on a switch, you disable RSTP on Switch 1 and Switch 2 in the first configuration task, and you disable RSTP on Switch 3 in the second task.

The second configuration task creates a redundant trunk group called **example 1** on Switch 3. The trunk interfaces **ge-0/0/9.0** and **ge-0/0/10.0** are the two links configured in the second configuration task. You configure the trunk interface **ge-0/0/9.0** as the primary link. You configure the trunk interface **ge-0/0/10.0** as an unspecified link, which becomes the secondary link by default.

Figure 50: Topology for Configuring the Redundant Trunk Links

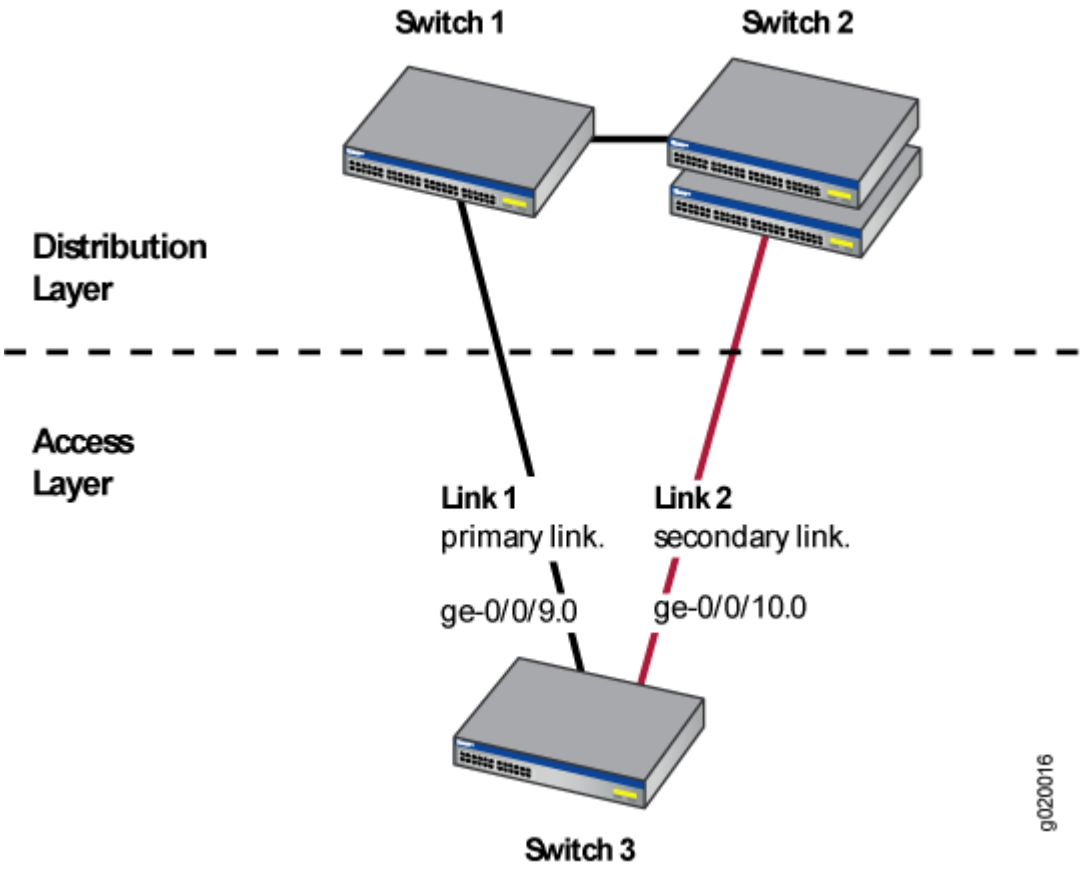


Table 128: Components of the Redundant Trunk Link Topology

Property	Settings
Switch hardware	• Switch 1–1 EX Series distribution switch • Switch 2–1 EX Series distribution switch • Switch 3–1 EX Series access switch
Trunk interfaces	On Switch 3 (access switch): **ge-0/0/9.0** and **ge-0/0/10.0**
Redundant trunk group	**example1**

Disabling RSTP on Switches 1 and 2

IN THIS SECTION

To disable RSTP on Switch 1 and Switch 2, perform this task on each switch:

Procedure

CLI Quick Configuration

To quickly disable RSTP on Switch 1 and Switch 2, copy the following command and paste it into each switch terminal window:

```
[edit]
set protocols rstp disable
```

Step-by-Step Procedure

To disable RSTP on Switch 1 and Switch 2:

1. Disable RSTP on Switch 1 and Switch 2:

```
[edit]
user@switch# set protocols rstp disable
```

Results

Check the results of the configuration:

```
[edit]
user@switch# show
protocols {
    rstp {
```

```
        disable;
    }
}
```

Results

Configuring Redundant Trunk Links on Switch 3

IN THIS SECTION

To configure redundant trunk links on Switch 3, perform this task:

Procedure

CLI Quick Configuration

To quickly configure the redundant trunk group **example1** on Switch 3, copy the following commands and paste them into the switch terminal window:

```
[edit]
 set protocols rstp disable
 set ethernet-switching-options redundant-trunk-group group example1 interface ge-0/0/9.0 primary
 set ethernet-switching-options redundant-trunk-group group example1 interface ge-0/0/10.0
 set ethernet-switching-options redundant-trunk-group group example1 preempt-cutover-timer 60
```

Step-by-Step Procedure

Configure the redundant trunk group **example1** on Switch 3.

1. Turn off RSTP:

```
[edit]
user@switch# set protocols rstp disable
```

2. Name the redundant trunk group example1 while configuring trunk interface **ge-0/0/9.0** as the primary link and **ge-0/0/10** as an unspecified link to serve as the secondary link:

```
[edit ethernet-switching-options]
user@switch#  set redundant-trunk-group group example1 interface ge-0/0/9.0 primary
user@switch#  set redundant-trunk-group group example1 interface ge-0/0/10.0
```

3. (Optional) Change the length of time (from the default of 1 second) that a re-enabled primary link waits to take over for an active secondary link:

```
[edit ethernet-switching-options]
user@switch#  set redundant-trunk-group group example1 preempt-cutover-timer 60
```

Results

Check the results of the configuration:

```
[edit]
user@switch# show
ethernet-switching-options
    redundant-trunk-group {
        group example1 {
            preempt-cutover-timer 60;

            interface ge-0/0/9.0 {
                primary;
            }
            interface ge-0/0/10.0;
        }
    }
protocols {
rstp {



        disable;
    }
}
```

Verification

IN THIS SECTION

To confirm that the configuration is set up correctly, perform this task:

Verifying That a Redundant Trunk Group Was Created

Purpose

Verify that the redundant trunk group **example1** has been created on Switch 1 and that trunk interfaces are members of the redundant trunk group.

Action

List all redundant trunk groups configured on the switch:

```
user@switch> show redundant-trunk-group
Group      Interface   State       Time of last flap          Flap
name                                                          count

example1   ge-0/0/9.0  Up/Pri      Never                        0
           ge-0/0/10.0 Up          Never                        0
```

Meaning

The `show redundant-trunk-group` command lists all redundant trunk groups configured on the switch, both links' interface addresses, and the links' current states (up or down for an unspecified link, and up or down and primary for a primary link). For this configuration example, the output shows that the redundant trunk group **example1** is configured on the switch. The **(Up)** beside the interfaces indicates that both link cables are physically connected. The **(Pri)** beside trunk interface **ge-0/0/9.0** indicates that it is configured as the primary link.

Q-in-Q Support on Redundant Trunk Links Using LAGs with Link Protection

IN THIS SECTION

Understanding Q-in-Q Support on RTGs Using LAGs with Link Protection

Redundant trunk links provide a simple solution for network recovery when a trunk port on a switch goes down. In that case, traffic is routed to another trunk port, keeping network convergence time to a minimum.

NOTE: For information about using redundant trunk links in a legacy redundant trunk groups (RTG) setup—that is, an RTG configuration that does not support Q-in-Q or service-provider configurations—see "Understanding Redundant Trunk Links (Legacy RTG Configuration)" on page 994.

You can use this feature of redundant trunk links (or RTG) with Q-in-Q support using LAGs with link protection in both service provider and enterprise configurations.

This feature of RTG with Q-in-Q support includes support for the following items that are *not* supported in legacy RTG configurations:

- Configuration of flexible VLAN tagging on the same LAG that supports the redundant links configurations
- Multiple redundant-link configurations on one physical interface
- Multicast convergence

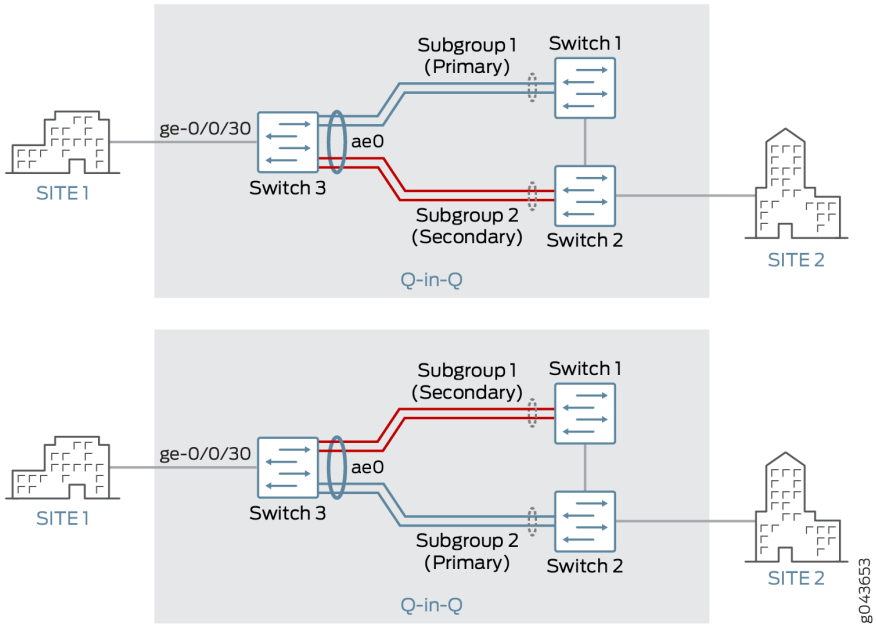
The redundant trunk link configuration (also known as a “redundant trunk group (RTG) configuration”) contains two links: a *primary* or *active* link and a *secondary* link. If the primary link fails, the secondary link automatically starts forwarding data traffic without waiting for normal spanning-tree protocol convergence.

Data traffic is forwarded only on the primary link. Data traffic received on the secondary link is dropped.

While data traffic is blocked on the secondary link, Layer 2 control traffic is still permitted. For example, you can run an LLDP session between two switches on the secondary link.

Rapid Spanning Tree Protocol (RSTP) is enabled by default on the switches to create a loop-free topology, but an interface cannot be in both a redundant trunk link and in a spanning-tree protocol topology at the same time. You must disable RSTP on an interface if a redundant trunk link is configured on that interface. Spanning-tree protocols can, however, continue operating on other interfaces on those switches.

Figure 51: Q-in-Q with Redundant Trunk Links Using LAGs with Link Protection, with Subgroups



The top of Figure 1 shows three switches in a topology for redundant trunk links on a LAG with flexible VLAN tagging. This particular configuration also includes subgroups that contain multiple links—there can be just two subgroups on the LAG, and both subgroups must have the same number of links.

NOTE: The topology shown in Figure 1 applies only to the first of the three configurations described later in this topic. See Configuring Redundant Trunk Links on an LACP LAG (N:N Link Protection with Subgroups). While the remaining configuration tasks share some elements of the first task, some absolute values provided in each task are unique to that task—for example, the ingress interface has a different value in each task.

Switch 3 is connected to Switch 1 through Subgroup 1 and to Switch 2 through Subgroup 2. Subgroups 1 and 2 are in an aggregated Ethernet bundle, or link aggregation group (LAG), with interface name ae0. Subgroup 1 is designated as the primary link, and Subgroup 2 is designated as the secondary link. Traffic flows between Switch 3 and Switch 1 through Subgroup 1. While Subgroup 1 is active, Subgroup 2 blocks data traffic.

The bottom of Figure 1 illustrates how the redundant trunk link topology works when the primary link goes down.

When Subgroup 1 between Switch 1 and Switch 3 goes down, Subgroup 2 takes over as the primary (active) link. Traffic flows between Switch 3 and Switch 2 through Subgroup 2.

NOTE: Here is how *multicast convergence* works in a topology such as the one illustrated in the preceding figure:

- With LAG ae0 being a multicast router port, all IGMP join messages received on Switch 3 are forwarded to Switch 1.
- When the link between Switch 3 and Switch 1 goes down, traffic for the multicast source received on Switch 2 is flooded to all ports in the VLAN.
- When the primary link goes down, an IGMP general query is sent by Switch 3 to all ports in the VLAN, and the IGMP reports received from clients are forwarded to Switch 2, through which learning happens; thus, multicast convergence is achieved.

Configuring Redundant Trunk Links on a LAG with Link Protection and Flexible VLAN Tagging

IN THIS SECTION

There are several variations on the configuration of redundant trunk links on a LAG with link protection and with flexible VLAN tagging.

NOTE: For illustration purposes only, the following configuration tasks show absolute values, such as ge-0/0/30, rather than variables such as *interface-name*.

Configuring Redundant Trunk Links on an LACP LAG (N:N Link Protection with Subgroups)

1. Configure the ingress interface:

```
set interfaces ge-0/0/30 enable
set interfaces ge-0/0/30 flexible-vlan-tagging
set interfaces ge-0/0/30 encapsulation extended-vlan-bridge
set interfaces ge-0/0/30 unit 30 vlan-id 30
set interfaces ge-0/0/30 unit 30 input-vlan-map push
set interfaces ge-0/0/30 unit 30 output-vlan-map pop
set vlans qinqvlan interface ge-0/0/30.30
```

2. Assign interfaces to the LAG, and assign those interfaces to two link-protection subgroups:

```
set interfaces ge-0/0/14 ether-options 802.3ad ae0
set interfaces ge-0/0/15 ether-options 802.3ad ae0
set interfaces ge-0/0/16 ether-options 802.3ad ae0
set interfaces ge-0/0/17 ether-options 802.3ad ae0
set interfaces ge-0/0/14 ether-options 802.3ad link-protection-sub-group subg1
set interfaces ge-0/0/15 ether-options 802.3ad link-protection-sub-group subg1
set interfaces ge-0/0/16 ether-options 802.3ad link-protection-sub-group subg2
set interfaces ge-0/0/17 ether-options 802.3ad link-protection-sub-group subg2
```

3. Assign LACP values, configure redundant trunk links (using the `rtg-config` statement) on the LAG, set one subgroup as primary and one as backup, and configure Q-in-Q on the LAG:

```
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation extended-vlan-bridge
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp periodic fast
set interfaces ae0 aggregated-ether-options lacp link-protection rtg-config
set interfaces ae0 aggregated-ether-options lacp system-id 00:00:00:01:00:02
set interfaces ae0 aggregated-ether-options link-protection-sub-group subg1 primary
```

```
set interfaces ae0 aggregated-ether-options link-protection-sub-group subg2 backup
set interfaces ae0 unit 300 vlan-id 40
set vlans qinqvlan interface ae0.300
```

Configuring Redundant Trunk Links on a Static LAG (1:1 Link Protection)

1. Configure the ingress interface:

```
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 4001 vlan-id-list 1-100
set interfaces ge-1/0/0 unit 4001 input-vlan-map push
set interfaces ge-1/0/0 unit 4001 output-vlan-map pop
```

2. Assign interfaces to the LAG, configure redundant trunk links (using the `rtg-config` statement) on the LAG, and configure Q-in-Q on the LAG:

```
set interfaces ae0 enable
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation extended-vlan-bridge
set interfaces ae0 aggregated-ether-options link-protection rtg-config
set interfaces ae0 aggregated-ether-options minimum-links 1
set interfaces ae0 unit 300 vlan-id 40
set vlans qinqvlan interface ae0.300
set vlans qinqvlan interface ge-1/0/0.4001
```

3. Assign the member interfaces on the LAG and assign the two redundant trunk links to the interfaces; set one of the links as primary and one as backup:

```
set interfaces ge-1/0/22 enable
set interfaces ge-1/0/22 ether-options 802.3ad ae0
set interfaces ge-1/0/22 ether-options 802.3ad primary
set interfaces ge-1/0/23 enable
set interfaces ge-1/0/23 ether-options 802.3ad ae0
set interfaces ge-1/0/23 ether-options 802.3ad backup
```

Configuring Redundant Trunk Links on a LAG with Multiple Logical Interfaces (1:1 Link Protection)

1. Configure the ingress interface:

```
set interfaces ge-0/0/0 flexible-vlan-tagging
set interfaces ge-0/0/0 encapsulation extended-vlan-bridge
set interfaces ge-0/0/0 unit 4001 vlan-id-list 1-100
set interfaces ge-0/0/0 unit 4001 input-vlan-map push
set interfaces ge-0/0/0 unit 4001 output-vlan-map pop
set interfaces ge-0/0/0 unit 5001 vlan-id-list 101-130
set interfaces ge-0/0/0 unit 5001 input-vlan-map push
set interfaces ge-0/0/0 unit 5001 output-vlan-map pop
```

2. Assign interfaces to the LAG, configure redundant trunk links (using the `rtg-config` statement) on the LAG, and configure Q-in-Q on the LAG:

```
set interfaces ae0 enable
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation extended-vlan-bridge
set interfaces ae0 aggregated-ether-options link-protection rtg-config
set interfaces ae0 aggregated-ether-options minimum-links 1
set interfaces ae0 unit 300 vlan-id 40
set interfaces ae0 unit 400 vlan-id 110
set vlans qinqvlan interface ae0.300
set vlans qinqvlan interface ge-0/0/0.4001
set vlans qinqvlan1 interface ae0.400
set vlans qinqvlan1 interface ge-0/0/0.5001
```

3. Assign the member interfaces on the LAG and assign the two redundant trunk links to the *member* interfaces; set one of the links as primary and one as backup:

```
set interfaces ge-0/0/5 enable
set interfaces ge-0/0/5 ether-options 802.3ad ae0
set interfaces ge-0/0/5 ether-options 802.3ad primary
set interfaces ge-0/0/6 enable
set interfaces ge-0/0/6 ether-options 802.3ad ae0
set interfaces ge-0/0/6 ether-options 802.3ad backup
```

SEE ALSO

show mac-refresh

show lacp interfaces

show interfaces ge

Verifying That Redundant Trunk Links Are Available on the LAG and Viewing Active Links

IN THIS SECTION

Purpose

Verify that the redundant trunk links are available on the LAG, and see which interfaces are configured as the primary (active) links.

Action

Use the following `show` commands:

- `show mac-refresh` *`interface-name`*—Display whether redundant trunk links on a LAG with link protection are enabled on the specified interface.
- `show interfaces ge` *`interface-name`* `extensive` or `show interfaces xe` *`interface-name`* `extensive`—On a static LAG, display which interface is set as the primary member.
- `show lacp interfaces`—On an LACP LAG, display which member interfaces are active and which are down.

RELATED DOCUMENTATION

Understanding Redundant Trunk Links (Legacy RTG Configuration) | 994

31

CHAPTER

Configuring Proxy ARP

IN THIS CHAPTER

Proxy ARP

IN THIS SECTION

Understanding Proxy ARP

IN THIS SECTION

You can configure proxy Address Resolution Protocol (ARP) to enable the switch to respond to ARP queries for network addresses by offering its own Ethernet media access control (MAC) address. With proxy ARP enabled, the switch captures and routes traffic to the intended destination.

Proxy ARP is useful in situations where hosts are on different physical networks and you do not want to use subnet masking. Because ARP broadcasts are not propagated between hosts on different physical networks, hosts will not receive a response to their ARP request if the destination is on a different subnet. Enabling the switch to act as an ARP proxy allows the hosts to transparently communicate with each other through the switch. Proxy ARP can help hosts on a subnet reach remote subnets without your having to configure routing or a default gateway.

Benefits of Using Proxy ARP

- Enables the switch to respond to ARP queries for network addresses by offering its own Ethernet media access control (MAC) address.
- Enables the switch to act as an ARP proxy allows the hosts to transparently communicate with each other through the switch.
- Helps hosts on a subnet reach remote subnets without your having to configure routing or a default gateway.

What Is ARP?

Ethernet LANs use ARP to map Ethernet MAC addresses to IP addresses. Each device maintains a cache containing a mapping of MAC addresses to IP addresses. The switch maintains this mapping in a cache that it consults when forwarding packets to network devices. If the ARP cache does not contain an entry for the destination device, the host (the DHCP client) broadcasts an ARP request for that device's address and stores the response in the cache.

Proxy ARP Overview

When proxy ARP is enabled, if the switch receives an ARP request for which it has a route to the target (destination) IP address, the switch responds by sending a proxy ARP reply packet containing its own MAC address. The host that sent the ARP request then sends its packets to the switch, which forwards them to the intended host.

NOTE: For security reasons, the source address in an ARP request must be on the same subnet as the interface on which the ARP request is received.

You can configure proxy ARP for each interface. You can also configure proxy ARP for an integrated routing and bridging (IRB) interface named irb or a *routed VLAN interface* (RVI) named vlan. (On EX Series switches that use Juniper Networks Junos operating system (Junos OS) with support for the Enhanced Layer 2 Software (ELS) configuration style, the feature is known as an IRB interface. On EX Series switches that use Junos OS that does not support ELS, the feature is known as an RVI.)

Two modes of proxy ARP are supported: restricted and unrestricted. Both modes require that the switch have an active route to the destination address of the ARP request.

- Restricted—The switch responds to ARP requests in which the physical networks of the source and target are different and does not respond if the source and target IP addresses are on the same subnet. In this mode, hosts on the same subnet communicate without proxy ARP. We recommend that you use this mode on the switch.

- Unrestricted—The switch responds to all ARP requests for which it has a route to the destination. This is the default mode (because it is the default mode in Juniper Networks Junos operating system (Junos OS) configurations other than those on the switch). We recommend using restricted mode on the switch.

Best Practices for Proxy ARP

We recommend these best practices for configuring proxy ARP on the switches:

- Set proxy ARP to restricted mode.
- Use restricted mode when configuring proxy ARP on RVIs or IRB interfaces.
- If you set proxy ARP to unrestricted, disable gratuitous ARP requests on each interface enabled for proxy ARP.

Configuring Proxy ARP on Devices with ELS Support

NOTE: This task uses Junos OS for EX Series switches and QFX3500 and QFX3600 switches with support for the Enhanced Layer 2 Software (ELS) configuration style.If your switch runs software that does not support ELS, see "Configuring Proxy ARP on Switches" on page 1025 or "Configuring Proxy ARP" on page 1026. For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15.

You can configure proxy Address Resolution Protocol (ARP) on your switch to enable the switch to respond to ARP queries for network addresses by offering its own media access control (MAC) address. With proxy ARP enabled, the switch captures and routes traffic to the intended destination.

To configure proxy ARP on a single interface:

```
[edit interfaces]
user@switch# set interface-name unit logical-unit-number proxy-arp (restricted | unrestricted)
```

BEST PRACTICE: We recommend that you configure proxy ARP in restricted mode. In restricted mode, the switch does not act as a proxy if the source and target IP addresses are on the same subnet. If you decide to use unrestricted mode, disable gratuitous ARP requests on the interface to avoid a situation wherein the switch's response to a gratuitous ARP request appears to the host to be an indication of an IP conflict.

To configure proxy ARP on an integrated routing and bridging (IRB) interface:

```
[edit interfaces]
user@switch# set irb.logical-unit-number proxy-arp restricted
```

Configuring Proxy ARP on Switches

NOTE: This task uses Junos OS for EX Series switches that does not support the Enhanced Layer 2 Software (ELS) configuration style. If your switch runs software that supports ELS, see "Configuring Proxy ARP on Devices with ELS Support" on page 1024. For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15.

You can configure proxy Address Resolution Protocol (ARP) on your EX Series switch to enable the switch to respond to ARP queries for network addresses by offering its own media access control (MAC) address. With proxy ARP enabled, the switch captures and routes traffic to the intended destination.

To configure proxy ARP on a single interface:

```
[edit interfaces]
user@switch# set ge-0/0/3 unit 0 proxy-arp restricted
```

BEST PRACTICE: We recommend that you configure proxy ARP in restricted mode. In restricted mode, the switch is not a proxy if the source and target IP addresses are on the same subnet. If you use unrestricted mode, disable gratuitous ARP requests on the interface to avoid the situation of the switch's response to a gratuitous ARP request appearing to the host to be an indication of an IP conflict:

To configure proxy ARP on a routed VLAN interface (RVI):

```
[edit interfaces]
user@switch# set vlan unit 100 proxy-arp restricted
```

SEE ALSO

Configuring Routed VLAN Interfaces on Switches (CLI Procedure)

Configuring Proxy ARP

You can configure proxy Address Resolution Protocol (ARP) to enable the switch to respond to ARP queries for network addresses by offering its own media access control (MAC) address. With proxy ARP enabled, the switch captures and routes traffic to the intended destination.

To configure proxy ARP on a single interface:

```
[edit interfaces]
user@switch# set xe-0/0/3 unit 0 proxy-arp restricted
```

BEST PRACTICE: We recommend that you configure proxy ARP in restricted mode. In restricted mode, the switch is not a proxy if the source and target IP addresses are on the same subnet. If you use unrestricted mode, disable gratuitous ARP requests on the interface to avoid the situation of the switch's response to a gratuitous ARP request appearing to the host to be an indication of an IP conflict:

To configure proxy ARP on a routed VLAN interface (RVI):

```
[edit interfaces]
user@switch# set vlan unit 100 proxy-arp restricted
```

SEE ALSO

Understanding Integrated Routing and Bridging | 765

Verifying That Proxy ARP Is Working Correctly

IN THIS SECTION

Purpose

Verify that the switch is sending proxy ARP messages.

Action

List the system statistics for ARP:

```
user@switch> show system statistics arp
arp:
         90060 datagrams received
         34 ARP requests received
         610 ARP replies received
         2 resolution request  received
         0 unrestricted proxy requests
         0 restricted proxy requests
         0 received proxy requests
         0 unrestricted proxy requests not proxied
         0 restricted proxy requests not proxied
         0 datagrams with bogus interface
         0 datagrams with incorrect length
         0 datagrams for non-IP protocol
         0 datagrams with unsupported op code
         0 datagrams with bad protocol address length
         0 datagrams with bad hardware address length
         0 datagrams with multicast source address
         0 datagrams with multicast target address
         0 datagrams with my own hardware address
         0 datagrams for an address not on the interface
```

```
          0 datagrams with a broadcast source address
          294 datagrams with source address duplicate to mine
          89113 datagrams which were not for me
          0 packets discarded waiting for resolution
          0 packets sent after waiting for resolution
          309 ARP requests sent
          35 ARP replies sent
          0 requests for memory denied
          0 requests dropped on entry
          0 requests dropped during retry
          0 requests dropped due to interface deletion
          0 requests on unnumbered interfaces
          0 new requests on unnumbered interfaces
          0 replies for from unnumbered interfaces
          0 requests on unnumbered interface with non-subnetted donor
          0 replies from unnumbered interface with non-subnetted donor
```

Meaning

The statistics show that two proxy ARP requests were received. The `unrestricted proxy requests not proxied` and `restricted proxy requests not proxied` fields indicate that all the unproxied ARP requests received have been proxied by the switch.

CHAPTER 32

Configuring Layer 2 Interfaces on Security Devices

IN THIS CHAPTER

Layer 2 Interfaces on Security Devices

IN THIS SECTION

Understanding Layer 2 Interfaces on Security Devices

Layer 2 logical interfaces are created by defining one or more logical units on a physical interface with the family address type **ethernet-switching**. If a physical interface has a **ethernet-switching** family *logical interface*, it cannot have any other family type in its logical interfaces. A logical interface can be configured in one of the following modes:

- Access mode—Interface accepts untagged packets, assigns the specified VLAN identifier to the packet, and forwards the packet within the VLAN that is configured with the matching VLAN identifier.
- Trunk mode—Interface accepts any packet tagged with a VLAN identifier that matches a specified list of VLAN identifiers. Trunk mode interfaces are generally used to interconnect switches. To configure a VLAN identifier for untagged packets received on the physical interface, use the `native-vlan-id` option. If the `native-vlan-id` option is not configured, untagged packets are dropped.

NOTE: Multiple trunk mode logical interfaces can be defined, as long as the VLAN identifiers of a trunk interface do not overlap with those of another trunk interface. The `native-vlan-id` must belong to a VLAN identifier list configured for a trunk interface.

SEE ALSO

Ethernet Switching and Layer 2 Transparent Mode Overview | 5

Example: Configuring Layer 2 Logical Interfaces on Security Devices | 1031

Example: Configuring Layer 2 Logical Interfaces on Security Devices

IN THIS SECTION

This example shows how to configure a Layer 2 logical interface as a trunk port so that the incoming packets can be selectively redirected to a firewall or other security device.

Requirements

Before you begin, configure the VLANs. See "Example: Configuring VLANs on Security Devices" on page 153.

Overview

In this example, you configure logical interface ge-3/0/0.0 as a trunk port that carries traffic for packets tagged with VLAN identifiers 1 through 10; this interface is implicitly assigned to the previously configured VLANs vlan-a and vlan-b. Then you assign a VLAN ID of 10 to any untagged packets received on physical interface ge-3/0/0.

Configuration

IN THIS SECTION

CLI Quick Configuration

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

```
set interfaces ge-3/0/0 unit 0 family ethernet-switching interface-mode trunk vlan members 1-10
set interfaces ge-3/0/0 vlan-tagging native-vlan-id 10
```

Procedure

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure a Layer 2 logical interface as a trunk port:

1. Configure the logical interface.

```
[edit interfaces ge-3/0/0]
user@host# set unit 0 family ethernet-switching interface-mode trunk vlan members 1-10
```

2. Specify a VLAN ID for untagged packets.

```
[edit interfaces ge-3/0/0]
user@host# set vlan-tagging native-vlan-id 10
```

3. If you are done configuring the device, commit the configuration.

```
[edit]
user@host# commit
```

Verification

To verify the configuration is working properly, enter the `show interfaces ge-3/0/0` and `show interfaces ge-3/0/0.0` commands.

SEE ALSO

Understanding Layer 2 Interfaces on Security Devices | 1030

Example: Configuring Layer 2 Security Zones | 1051

Understanding Mixed Mode (Transparent and Route Mode) on Security Devices

Mixed mode supports both transparent mode (Layer 2) and route mode (Layer 3); it is the default mode. You can configure both Layer 2 and Layer 3 interfaces simultaneously using separate security zones.

NOTE: For the mixed mode configuration, you must reboot the device after you commit the changes. However, for SRX5000 line devices, reboot is not required.

SRX4100 and SRX4200 devices support logical system in both transparent and route mode

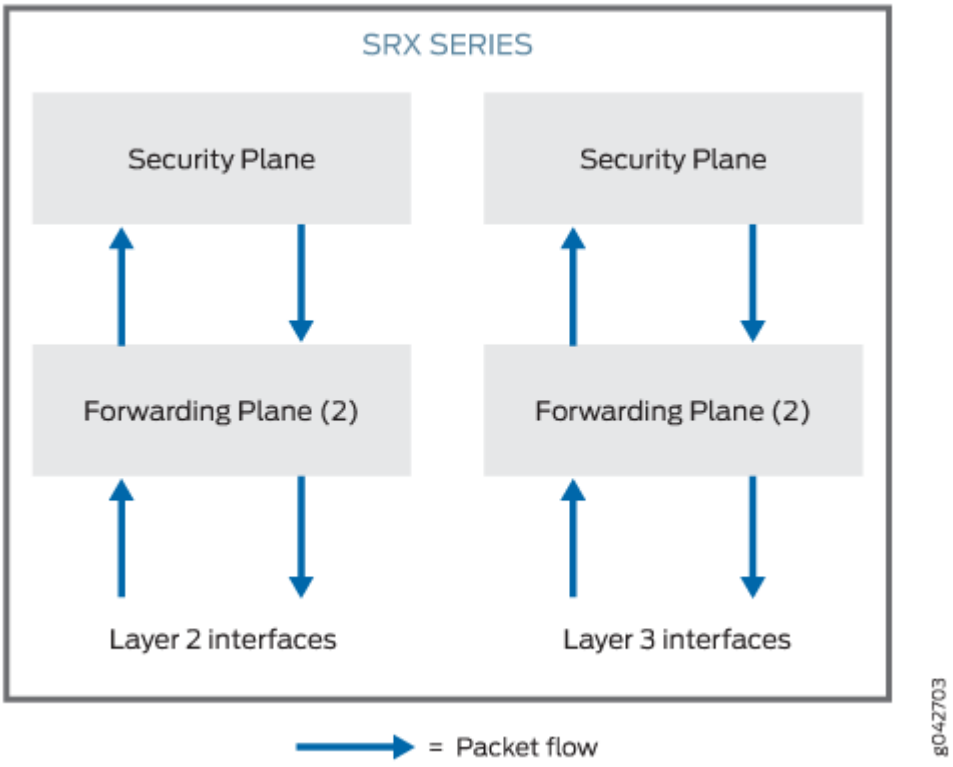
SRX4600 device supports logical system in route mode only

In mixed mode (Transparent and Route Mode):

- There is no routing among IRB interfaces and between IRB interfaces and Layer 3 interfaces.

The device in Figure 52 on page 1034 looks like two separate devices. One device runs in Layer 2 transparent mode and the other device runs in Layer 3 routing mode. But both devices run independently. Packets cannot be transferred between the Layer 2 and Layer 3 interfaces, because there is no routing among IRB interfaces and between IRB interfaces and Layer 3 interfaces.

Figure 52: Architecture of Mixed Transparent and Route Mode



In mixed mode, the Ethernet physical interface can be either a Layer 2 interface or a Layer 3 interface, but the Ethernet physical interface cannot be both simultaneously. However, Layer 2 and Layer 3 families can exist on separate physical interfaces on the same device.

Table 129 on page 1034 lists the Ethernet physical interface types and supported family types.

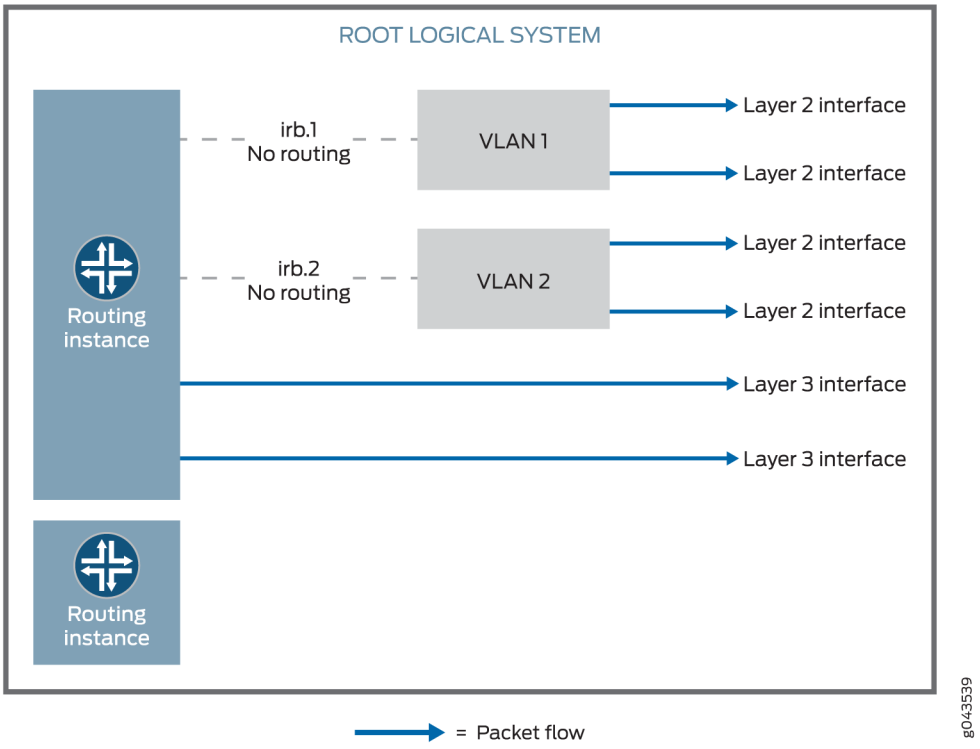
Table 129: Ethernet Physical Interface and Supported Family Types

Ethernet Physical Interface Type	Supported Family Type
Layer 2 Interface	`ethernet-switching`
Layer 3 Interface	`inet` and `inet6`

NOTE: Multiple routing instances are supported.

You can configure both the pseudointerface `irb.x` and the Layer 3 interface under the same default routing instance using either a default routing instance or a user-defined routing instance. See Figure 53 on page 1035.

Figure 53: Mixed Transparent and Route Mode



Packets from the Layer 2 interface are switched within the same VLAN, or they connect to the host through the IRB interface. Packets cannot be routed to another IRB interface or a Layer 3 interface through their own IRB interface.

Packets from the Layer 3 interface are routed to another Layer 3 interface. Packets cannot be routed to a Layer 2 interface through an IRB interface.

Table 130 on page 1036 lists the security features that are supported in mixed mode and the features that are not supported in transparent mode for Layer 2 switching.

Table 130: Security Features Supported in Mixed Mode (Transparent and Route Mode)

Mode Type	Supported	Not Supported
Mixed mode	• Application Layer Gateways (ALGs) • Firewall User Authentication (FWAUTH) • Intrusion Detection and Prevention (IDP) • Screen • AppSecure • Content Security	—
Route mode (Layer 3 interface)	• Network Address Translation (NAT) • VPN	—
Transparent mode (Layer 2 interface)	• Content Security	• Network Address Translation (NAT) • VPN

Starting in Junos OS Release 12.3X48-D10 and Junos OS Release 17.3R1, some conditions apply to mixed-mode operations. Note the conditions here:

- On SRX300, SRX320, SRX340, SRX345, SRX380, SRX550, SRX550HM, and SRX1500 devices, you cannot configure Ethernet switching and virtual private LAN service (VPLS) using mixed mode (Layer 2 and Layer 3).
- On SRX5400, SRX5600, and SRX5800 devices, you do not have to reboot the device when you configure VLAN.

SEE ALSO

Example: Improving Security Services by Configuring an SRX Series Firewall Using Mixed Mode (Transparent and Route Mode) | **1037**

Understanding Secure Wire on Security Devices

Understanding Mixed Mode (Transparent and Route Mode) on Security Devices | **1033**

Example: Improving Security Services by Configuring an SRX Series Firewall Using Mixed Mode (Transparent and Route Mode)

IN THIS SECTION

You can configure an SRX Series Firewall using both transparent mode (Layer 2) and route mode (Layer 3) simultaneously to simplify deployments and to improve security services.

This example shows how to pass the Layer 2 traffic from interface ge-0/0/1.0 to interface ge-0/0/0.0 and Layer 3 traffic from interface ge-0/0/2.0 to interface ge-0/0/3.0.

Requirements

This example uses the following hardware and software components:

- An SRX Series Firewall
- Four PCs

Before you begin:

- Create a separate security zone for Layer 2 and Layer 3 interfaces. See "Understanding Layer 2 Security Zones" on page 1050.

Overview

IN THIS SECTION

In enterprises where different business groups have either Layer 2 or Layer 3 based security solutions, using a single mixed mode configuration simplifies their deployments. In a mixed mode configuration, you can also provide security services with integrated switching and routing.

In addition, you can configure an SRX Series Firewall in both standalone and chassis cluster mode using mixed mode.

In mixed mode (default mode), you can configure both Layer 2 and Layer 3 interfaces simultaneously using separate security zones.

NOTE: For the mixed mode configuration, you must reboot the device after you commit the changes. However, for SRX5000 line devices, reboot is not required.

In this example, first you configure a Layer 2 family type called Ethernet switching to identify Layer 2 interfaces. You set the IP address 10.10.10.1/24 to IRB interface. Then you create zone L2 and add Layer 2 interfaces ge-0/0/1.0 and ge-0/0/0.0 to it.

Next you configure a Layer 3 family type inet to identify Layer 3 interfaces. You set the IP address 192.0.2.1/24 to interface ge-0/0/2.0 and the IP address 192.0.2.3/24 to interface ge-0/0/3. Then you create zone L3 and add Layer 3 interfaces ge-0/0/2.0 and ge-0/0/3.0 to it.

Topology

Figure 54 on page 1039 shows a mixed mode topology.

Figure 54: Mixed Mode Topology

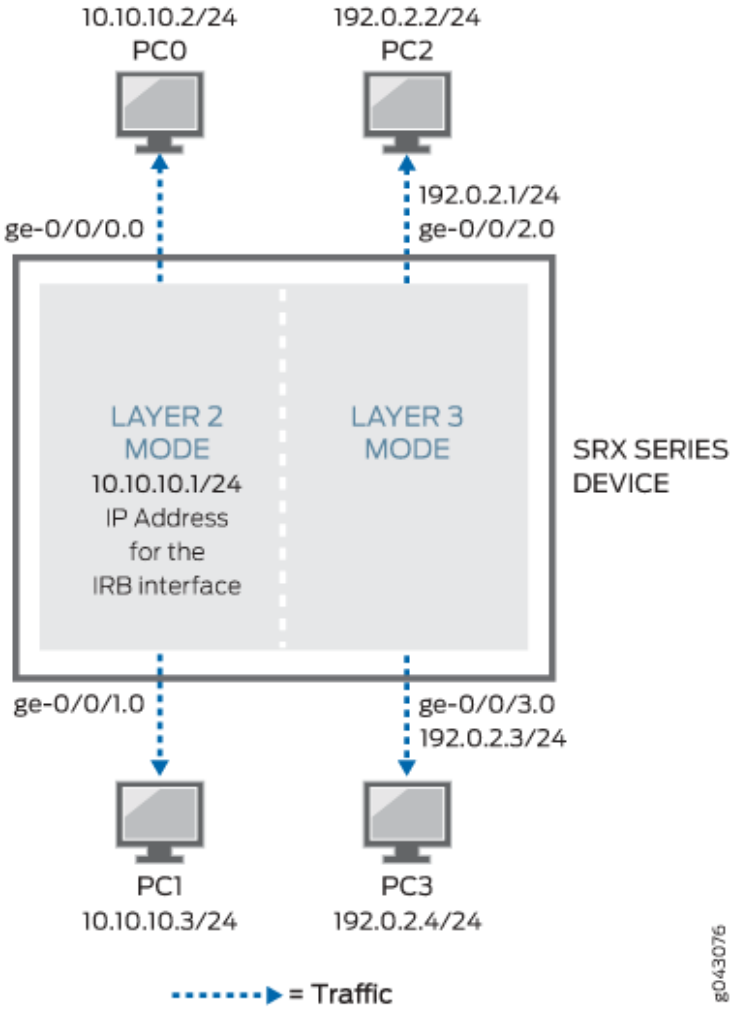


Table 131 on page 1039 shows the parameters configured in this example.

Table 131: Layer 2 and Layer 3 Parameters

Parameter	Description
L2	Layer 2 zone.
ge-0/0/1.0 and ge-0/0/0.0	Layer 2 interfaces added to the Layer 2 zone.

Table 131: Layer 2 and Layer 3 Parameters *(Continued)*

Parameter	Description
L3	Layer 3 zone.
ge-0/0/2.0 and ge-0/0/3.0	Layer 3 interfaces added to the Layer 3 zone.
10.10.10.1/24	IP address for the IRB interface.
192.0.2.1/24 and 192.0.2.3/24	IP addresses for the Layer 3 interface.

Configuration

IN THIS SECTION

- Procedure | 1040

Procedure

CLI Quick Configuration

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

```
set interfaces ge-0/0/0 unit 0 family ethernet-switching interface-mode access
set interfaces ge-0/0/0 unit 0 family ethernet-switching vlan members 10
set interfaces ge-0/0/1 unit 0 family ethernet-switching interface-mode access
set interfaces ge-0/0/1 unit 0 family ethernet-switching vlan members 10
set protocols l2-learning global-mode transparent-bridge
set interfaces irb unit 10 family inet address 10.10.10.1/24
set security zones security-zone L2 interfaces ge-0/0/1.0
set security zones security-zone L2 interfaces ge-0/0/0.0
set vlans vlan-10 vlan-id 10
```

```
set vlans vlan-10 l3-interface irb.10
set interfaces ge-0/0/2 unit 0 family inet address 192.0.2.1/24
set interfaces ge-0/0/3 unit 0 family inet address 192.0.2.3/24
set security policies default-policy permit-all
set security zones security-zone L2 host-inbound-traffic system-services any-service
set security zones security-zone L2 host-inbound-traffic protocols all
set security zones security-zone L3 host-inbound-traffic system-services any-service
set security zones security-zone L3 host-inbound-traffic protocols all
set security zones security-zone L3 interfaces ge-0/0/2.0
set security zones security-zone L3 interfaces ge-0/0/3.0
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure Layer 2 and Layer 3 interfaces:

1. Create a Layer 2 family type to configure Layer 2 interfaces.

```
[edit interfaces]
user@host# set ge-0/0/0 unit 0 family ethernet-switching interface-mode access
user@host# set ge-0/0/0 unit 0 family ethernet-switching vlan members 10
user@host# set ge-0/0/1 unit 0 family ethernet-switching interface-mode access
user@host# set ge-0/0/1 unit 0 family ethernet-switching vlan members 10
```

2. Configure Layer 2 interfaces to work under transparent-bridge mode.

```
[edit protocols]
user@host# set l2-learning global-mode transparent-bridge
```

3. Configure an IP address for the IRB interface.

```
[edit interfaces]
user@host# set irb unit 10 family inet address 10.10.10.1/24
```

4. Configure Layer 2 interfaces.

```
[edit security zones security-zone L2 interfaces]
user@host# set ge-0/0/1.0
user@host#  set ge-0/0/0.0
```

5. Configure VLAN.

```
[edit vlans vlan-10]
user@host# set vlan-id 10
user@host# set l3-interface irb.10
```

6. Configure IP addresses for Layer 3 interfaces.

```
[edit interfaces]
user@host# set ge-0/0/2 unit 0 family inet address 192.0.2.1/24
user@host# set ge-0/0/3 unit 0 family inet address 192.0.2.3/24
```

7. Configure the policy to permit the traffic.

```
[edit security policies]
user@host# set default-policy permit-all
```

8. Configure Layer 3 interfaces.

```
[edit security zones security-zone]
user@host# set L2 host-inbound-traffic system-services any-service
user@host# set L2 host-inbound-traffic protocols all
user@host# set L3 host-inbound-traffic system-services any-service
user@host# set L3 host-inbound-traffic protocols all
user@host# set L3 interfaces ge-0/0/2.0
user@host# set L3 interfaces ge-0/0/3.0
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show security policies`, `show vlans`, and `show security zones` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show interfaces
    ge-0/0/0 {
        unit 0 {
            family ethernet-switching {
                interface-mode access;
                vlan {
                    members 10;
                }
            }
        }
    }
    ge-0/0/1 {
        unit 0 {
            family ethernet-switching {
                interface-mode access;
                vlan {
                    members 10;
                }
            }
        }
    }
    ge-0/0/2 {
        unit 0 {
            family inet {
                address 192.0.2.1/24;
            }
        }
    }
    ge-0/0/3 {
        unit 0 {
            family inet {
                address 192.0.2.3/24;
            }
        }
    }
```

```
    irb {
        unit 10 {
            family inet {
                address 10.10.10.1/24;
            }
        }
    }
[edit]
user@host#  show security policies
default-policy {
    permit-all;
}
[edit]
user@host# show vlans
    vlan-10 {
        vlan-id 10;
        l3-interface irb.10;
    }
[edit]
user@host#  show security zones
    security-zone L2 {
        host-inbound-traffic {
            system-services {
                any-service;
            }
            protocols {
                all;
            }
        }
        interfaces {
            ge-0/0/1.0;
            ge-0/0/0.0;
        }
    }
    security-zone L3 {
        host-inbound-traffic {
            system-services {
                any-service;
            }
            protocols {
                all;
            }
        }
```

```
        interfaces {
            ge-0/0/2.0;
            ge-0/0/3.0;
        }
    }
```

If you are done configuring the device, enter `commit` from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying the Layer 2 and Layer 3 Interfaces and Zones

Purpose

Verify that the Layer 2 and Layer 3 interfaces and Layer 2 and Layer 3 zones are created.

Action

From operational mode, enter the `show security zones` command.

```
user@host> show security zones
 Security zone: HOST
  Send reset for non-SYN session TCP packets: Off
  Policy configurable: Yes
  Interfaces bound: 0
  Interfaces:

Security zone: L2
  Send reset for non-SYN session TCP packets: Off
  Policy configurable: Yes
  Interfaces bound: 2
```

```
  Interfaces:
    ge-0/0/0.0
    ge-0/0/1.0

Security zone: L3
  Send reset for non-SYN session TCP packets: Off
  Policy configurable: Yes
  Interfaces bound: 2
  Interfaces:
    ge-0/0/2.0
    ge-0/0/3.0

Security zone: junos-host
  Send reset for non-SYN session TCP packets: Off
  Policy configurable: Yes
  Interfaces bound: 0
  Interfaces:
```

Meaning

The output shows the Layer 2 (L2) and Layer 3 (L3) zone names and the number and names of Layer 2 and Layer 3 interfaces bound to the L2 and L3 zones.

Verifying the Layer 2 and Layer 3 Session

Purpose

Verify that the Layer 2 and Layer 3 sessions are established on the device.

Action

From operational mode, enter the `show security flow session` command.

```
user@host> show security flow session
Session ID: 1, Policy name: default-policy-logical-system-00/2, Timeout: 58, Valid
  In: 10.102.70.75/54395 --> 228.102.70.76/9876;udp, Conn Tag: 0x0, If: ge-0/0/0.0, Pkts: 1209,
Bytes: 1695018,
  Out: 228.102.70.76/9876 --> 10.102.70.75/54395;udp, Conn Tag: 0x0, If: ge-0/0/1.0, Pkts: 0,
Bytes: 0,
```

```
Session ID: 2, Policy name: default-policy-logical-system-00/2, Timeout: 58, Valid
  In: 10.102.70.19/23364 --> 228.102.70.20/23364;udp, Conn Tag: 0x0, If: ge-0/0/0.0, Pkts: 401,
Bytes: 141152,
  Out: 228.102.70.20/23364 --> 10.102.70.19/23364;udp, Conn Tag: 0x0, If: ge-0/0/1.0, Pkts: 0,
Bytes: 0,
```

Meaning

The output shows active sessions on the device and each session's associated security policy.

- **Session ID 1**—Number that identifies the Layer 2 session. Use this ID to get more information about the Layer 2 session such as policy name or number of packets in and out.
- **default-policy-logical-system-00/2**—Default policy name that permitted the Layer 2 traffic.
- **In**—Incoming flow (source and destination Layer 2 IP addresses with their respective source and destination port numbers, session is ICMP, and the source interface for this session is ge-0/0/0.0).
- **Out**—Reverse flow (source and destination Layer 2 IP addresses with their respective source and destination port numbers, session is ICMP, and destination interface for this session is ge-0/0/1.0).
- **Session ID 2**—Number that identifies the Layer 2 session. Use this ID to get more information about the Layer 2 session such as policy name or number of packets in and out.
- **default-policy-logical-system-00/2**—Default policy name that permitted the Layer 2 traffic.
- **In**—Incoming flow (source and destination Layer 2 IP addresses with their respective source and destination port numbers, session is ICMP, and the source interface for this session is ge-0/0/0.0,).
- **Out**—Reverse flow (source and destination Layer 2 IP addresses with their respective source and destination port numbers, session is ICMP, and destination interface for this session is ge-0/0/1.0,).

SEE ALSO

Understanding Mixed Mode (Transparent and Route Mode) on Security Devices | **1033**

Understanding Secure Wire on Security Devices

Change History Table

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

Release	Description
12.3X48-D10	Starting in Junos OS Release 12.3X48-D10 and Junos OS Release 17.3R1, some conditions apply to mixed-mode operations.

33

CHAPTER

Configuring Security Zones and Security Policies on Security Devices

IN THIS CHAPTER

Security Zones and Security Policies on Security Devices

IN THIS SECTION

Understanding Layer 2 Security Zones

A Layer 2 security zone is a zone that hosts Layer 2 interfaces. A security zone can be either a Layer 2 or Layer 3 zone; it can host either all Layer 2 interfaces or all Layer 3 interfaces, but it cannot contain a mix of Layer 2 and Layer 3 interfaces.

The security zone type—Layer 2 or Layer 3—is implicitly set from the first interface configured for the security zone. Subsequent interfaces configured for the same security zone must be the same type as the first interface.

NOTE: You cannot configure a device with both Layer 2 and Layer 3 security zones.

You can configure the following properties for Layer 2 security zones:

- Interfaces—List of interfaces in the zone.
- Policies—Active security policies that enforce rules for the transit traffic, in terms of what traffic can pass through the firewall, and the actions that need to take place on the traffic as it passes through the firewall.
- Screens—A Juniper Networks stateful firewall secures a network by inspecting, and then allowing or denying, all connection attempts that require passage from one security zone to another. For every security zone, and the MGT zone, you can enable a set of predefined screen options that detect and block various kinds of traffic that the device determines as potentially harmful.

> **NOTE:** You can configure the same screen options for a Layer 2 security zone as for a Layer 3 security zone.

- Address books—IP addresses and address sets that make up an address book to identify its members so that you can apply policies to them.
- TCP-RST—When this feature is enabled, the system sends a TCP segment with the reset flag set when traffic arrives that does not match an existing session and does not have the synchronize flag set.

In addition, you can configure a Layer 2 zone for host-inbound traffic. This allows you to specify the kinds of traffic that can reach the device from systems that are directly connected to the interfaces in the zone. You must specify all expected host-inbound traffic because inbound traffic from devices directly connected to the device's interfaces is dropped by default.

SEE ALSO

Ethernet Switching and Layer 2 Transparent Mode Overview | **5**

Understanding Layer 2 Interfaces on Security Devices | **1030**

Example: Configuring Layer 2 Security Zones | **1051**

Example: Configuring Layer 2 Logical Interfaces on Security Devices | **1031**

Example: Configuring Layer 2 Security Zones

IN THIS SECTION

This example shows how to configure Layer 2 security zones.

Requirements

Before you begin, determine the properties you want to configure for the Layer 2 security zone. See "Understanding Layer 2 Security Zones" on page 1050.

Overview

In this example, you configure security zone l2-zone1 to include a Layer 2 logical interface called ge-3/0/0.0 and security zone l2-zone2 to include a Layer 2 logical interface called ge-3/0/1.0. Then you configure l2-zone2 to allow all supported application services (such as SSH, Telnet, and SNMP) as host-inbound traffic.

Configuration

IN THIS SECTION

CLI Quick Configuration

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

```
set security-zone l2-zone1 interfaces ge-3/0/0.0
set security-zone l2-zone2 interfaces ge-3/0/1.0
set security-zone l2-zone2 host-inbound-traffic system-services all
```

Procedure

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure Layer 2 security zones:

1. Create a Layer 2 security zone and assign interfaces to it.

```
[edit security zones]
user@host# set security-zone l2-zone1 interfaces ge-3/0/0.0
user@host# set security-zone l2-zone2 interfaces ge-3/0/1.0
```

2. Configure one of the Layer 2 security zones.

```
[edit security zones]
user@host# set security-zone l2-zone2 host-inbound-traffic system-services all
```

3. If you are done configuring the device, commit the configuration.

```
[edit]
user@host# commit
```

Verification

To verify the configuration is working properly, enter the `show security zones` command.

SEE ALSO

Example: Configuring Security Policies in Transparent Mode | 1054

Example: Configuring Layer 2 Logical Interfaces on Security Devices | 1031

Understanding Security Policies in Transparent Mode

In transparent mode, security policies can be configured only between Layer 2 zones. When packets are forwarded through the VLAN, the security policies are applied between security zones. A security policy for transparent mode is similar to a policy configured for Layer 3 zones, with the following exceptions:

- NAT is not supported.
- IPsec VPN is not supported.
- Application ANY is not supported.

Layer 2 forwarding does not permit any interzone traffic unless there is a policy explicitly configured on the device. By default, Layer 2 forwarding performs the following actions:

- Allows or denies traffic specified by the configured policy.
- Allows Address Resolution Protocol (ARP) and Layer 2 non-IP multicast and broadcast traffic.
- Continues to block all non-IP and non-ARP unicast traffic.

This default behavior can be changed for Ethernet switching packet flow by using either J-Web or the CLI configuration editor:

- Configure the `block-non-ip-all` option to block all Layer 2 non-IP and non-ARP traffic, including multicast and broadcast traffic.
- Configure the `bypass-non-ip-unicast` option to allow all Layer 2 non-IP traffic to pass through the device.

NOTE: You cannot configure both options at the same time.

Starting in Junos OS Release 12.3X48-D10 and Junos OS Release 17.3R1, you can create a separate security zone in mixed mode (the default mode) for Layer 2 and Layer 3 interfaces. However, there is no routing among IRB interfaces and between IRB interfaces and Layer 3 interfaces. Hence, you cannot configure security policies between Layer 2 and Layer 3 zones. You can only configure security policies between the Layer 2 zones or between Layer 3 zones.

SEE ALSO

Example: Configuring Security Policies in Transparent Mode | **1054**

Example: Configuring Layer 2 Security Zones | **1051**

Understanding Mixed Mode (Transparent and Route Mode) on Security Devices | **1033**

Example: Configuring Security Policies in Transparent Mode

IN THIS SECTION

This example shows how to configure security policies in transparent mode between Layer 2 zones.

Requirements

Before you begin, determine the policy behavior you want to include in the Layer 2 security zone. See "Understanding Security Policies in Transparent Mode" on page 1053.

Overview

In this example, you configure a security policy to allow HTTP traffic from the 192.0.2.0/24 subnetwork in the l2–zone1 security zone to the server at 192.0.2.1/24 in the l2–zone2 security zone.

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

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

```
set security policies from-zone l2-zone1 to-zone l2-zone2 policy p1 match source-address
192.0.2.0/24
set security policies from-zone l2-zone1 to-zone l2-zone2 policy p1 match destination-address
192.0.2.1/24
set security policies from-zone l2-zone1 to-zone l2-zone2 policy p1 match application http
set security policies from-zone l2-zone1 to-zone l2-zone2 policy p1 then permit
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure security policies in transparent mode:

1. Create policies and assign addresses to the interfaces for the zones.

```
[edit security policies]
user@host# set from-zone l2-zone1 to-zone l2-zone2 policy p1 match source-address 192.0.2.0/24
user@host# set from-zone l2-zone1 to-zone l2-zone2 policy p1 match destination-address
192.0.2.1/24
```

2. Set policies for the application.

```
[edit security policies]
user@host# set from-zone l2-zone1 to-zone l2-zone2 policy p1 match application http
user@host# set from-zone l2-zone1 to-zone l2-zone2 policy p1 then permit
```

Results

From configuration mode, confirm your configuration by entering the `show security policies` command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
[edit]
    user@host> show security policies
    from-zone l2-zone1 to-zone l2-zone2
{
    policy p1 {
        match {
        source-address 192.0.2.0/24;
        destination-address 192.0.2.1/24;
        application junos-http;
    }
    then {
        permit;
    }
```

```
}
}
```

If you are done configuring the device, enter `commit` from configuration mode.

Verification

IN THIS SECTION

Verifying Layer 2 Security Policies

Purpose

Verify that the Layer 2 security policies are configured properly.

Action

From configuration mode, enter the `show security policies` command.

SEE ALSO

Example: Configuring Layer 2 Security Zones | **1051**

Understanding Firewall User Authentication in Transparent Mode

A firewall user is a network user who must provide a username and password for authentication when initiating a connection across the firewall. Firewall user authentication enables administrators to restrict and permit users accessing protected resources behind a firewall based on their source IP address and other credentials. Junos OS supports the following types of firewall user authentication for transparent mode on the SRX Series Firewall:

- Pass-through authentication—A host or a user from one zone tries to access resources on another zone. You must use an FTP, Telnet, or HTTP client to access the IP address of the protected resource and be authenticated by the firewall. The device uses FTP, Telnet, or HTTP to collect username and

password information, and subsequent traffic from the user or host is allowed or denied based on the result of this authentication.

- Web authentication—Users try to connect, by using HTTP, to an IP address on the IRB interface that is enabled for Web authentication. You are prompted for the username and password that are verified by the device. Subsequent traffic from the user or host to the protected resource is allowed or denied based on the result of this authentication.

SEE ALSO

Authentication and Integrated User Firewalls User Guide

Understanding Integrated Routing and Bridging | **765**

Example: Configuring an IRB Interface on a Security Device | **787**

Change History Table

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

Release	Description
12.3X48-D10	Starting in Junos OS Release 12.3X48-D10 and Junos OS Release 17.3R1, you can create a separate security zone in mixed mode (the default mode) for Layer 2 and Layer 3 interfaces.

34

CHAPTER

Configuring Ethernet Port Switching Modes on Security Devices

IN THIS CHAPTER

Ethernet Port Switching Modes on Security Devices

IN THIS SECTION

Understanding Switching Modes on Security Devices

There are two types of switching modes:

- Switching Mode–The uPIM appears in the list of interfaces as a single interface, which is the first interface on the uPIM. For example, ge-2/0/0. You can optionally configure each uPIM port only for autonegotiation, speed, and duplex mode. A uPIM in switching mode can perform the following functions:
  - Layer 3 forwarding—Routes traffic destined for WAN interfaces and other PIMs present on the chassis.
  - Layer 2 forwarding—Switches intra-LAN traffic from one host on the LAN to another LAN host (one port of uPIM to another port of same uPIM).
- Enhanced Switching Mode–Each port can be configured for switching or routing mode. This usage differs from the routing and switching modes, in which all ports must be in either switching or routing mode. The uPIM in enhanced switching mode provides the following benefits:

  Benefits of enhnanced switch mode:
  - Supports configuration of different types of VLANs and inter-VLAN routing.
  - Supports Layer 2 control plane protocol such as Link Aggregation Control Protocol (LACP).
  - Supports port-based Network Access Control (PNAC) by means of authentication servers.

NOTE: The SRX300 and SRX320 devices support enhanced switching mode only. When you set a multiport uPIM to enhanced switching mode, all the Layer 2 switching

features are supported on the uPIM. (Platform support depends on the Junos OS release in your installation.)

You can set a multiport Gigabit Ethernet uPIM on a device to either switching or enhanced switching mode.

When you set a multiport uPIM to switching mode, the uPIM appears as a single entity for monitoring purposes. The only physical port settings that you can configure are autonegotiation, speed, and duplex mode on each uPIM port, and these settings are optional.

Ethernet Ports Switching Overview for Security Devices

IN THIS SECTION

Certain ports on Juniper Networks devices can function as Ethernet access switches that switch traffic at Layer 2 and route traffic at Layer 3.

You can deploy supported devices in branch offices as an access or desktop switch with integrated routing capability, thus eliminating intermediate access switch devices from your network topology. The Ethernet ports provide switching while the Routing Engine provides routing functionality, enabling you to use a single device to provide routing, access switching, and WAN interfaces.

This topic contains the following sections:

Supported Devices and Ports

Juniper Networks supports switching features on a variety of Ethernet ports and devices (see Table 132 on page 1062). Platform support depends on the Junos OS release in your installation. The following ports and devices are included:

- Onboard Ethernet ports (Gigabit and Fast Ethernet built-in ports) on the SRX300, SRX320, SRX320 PoE, SRX340, SRX345, SRX550M and SRX1500 devices.
- Multiport Gigabit Ethernet XPIM on the SRX650 device.

Table 132: Supported Devices and Ports for Switching Features

Device	Ports
SRX100 devices	Onboard Fast Ethernet ports (fe-0/0/0 and fe-0/0/7)
SRX210 devices	Onboard Gigabit Ethernet ports (ge-0/0/0 and ge-0/0/1) and 1-Port Gigabit Ethernet SFP Mini-PIM port. Onboard Fast Ethernet ports (fe-0/0/2 and fe-0/0/7)
SRX220 devices	Onboard Gigabit Ethernet ports (ge-0/0/0 through ge-0/0/7) and 1-Port Gigabit Ethernet SFP Mini-PIM port.
SRX240 devices	Onboard Gigabit Ethernet ports (ge-0/0/0 through ge-0/0/15) and 1-Port Gigabit Ethernet SFP Mini-PIM port.
SRX300 devices	Onboard Gigabit Ethernet ports (ge-0/0/0 through ge-0/0/7)
SRX320 devices	Onboard Gigabit Ethernet ports (ge-0/0/0 through ge-0/0/7)
SRX340 devices	Onboard Gigabit Ethernet ports (ge-0/0/0 through ge-0/0/15)
SRX345 devices	Onboard Gigabit Ethernet ports (ge-0/0/0 through ge-0/0/15)
SRX550 devices	Onboard Gigabit Ethernet ports (ge-0/0/0 through ge-0/0/9, Multiport Gigabit Ethernet XPIM modules, and 1-Port Gigabit Ethernet SFP Mini-PIM port.

Table 132: Supported Devices and Ports for Switching Features *(Continued)*

Device	Ports
SRX550M devices	Onboard Gigabit Ethernet ports (ge-0/0/0 through ge-0/0/9 and Multiport Gigabit Ethernet XPIM modules.
SRX650 devices	Multiport Gigabit Ethernet XPIM modules **NOTE**: On SRX650 devices, Ethernet switching is not supported on Gigabit Ethernet interfaces (ge-0/0/0 through ge-0/0/3 ports).
SRX1500 devices	Onboard Gigabit Ethernet ports (ge-0/0/0 through ge-0/0/19)

On the SRX100. SRX220, SRX240, SRX300, SRX320, SRX340 and SRX345 devices, you can set the onboard Gigabit Ethernet ports to operate as either switched ports or routed ports. (Platform support depends on the Junos OS release in your installation.)

Integrated Bridging and Routing

Integrated bridging and routing (IRB) provides support for simultaneous Layer 2 switching and Layer 3 routing within the same VLAN. Packets arriving on an interface of the VLAN are switched or routed based on the destination MAC address of the packet. Packets with the router's MAC address as the destination are routed to other Layer 3 interfaces.

Link Layer Discovery Protocol and LLDP-Media Endpoint Discovery

Devices use Link Layer Discovery Protocol (LLDP) and LLDP-Media Endpoint Discovery (MED) to learn and distribute device information about network links. The information allows the device to quickly identify a variety of systems, resulting in a LAN that interoperates smoothly and efficiently.

LLDP-capable devices transmit information in Type Length Value (TLV) messages to neighbor devices. Device information can include specifics, such as chassis and port identification and system name and system capabilities. The TLVs leverage this information from parameters that have already been configured in the Junos OS.

LLDP-MED goes one step further, exchanging IP-telephony messages between the device and the IP telephone. These TLV messages provide detailed information about Power over Ethernet (PoE) policy. The PoE Management TLVs let the device ports advertise the power level and power priority needed.

For example, the device can compare the power needed by an IP telephone running on a PoE interface with available resources. If the device cannot meet the resources required by the IP telephone, the device could negotiate with the telephone until a compromise on power is reached.

The following basic TLVs are supported:

- Chassis Identifier—The MAC address associated with the local system.
- Port identifier—The port identification for the specified port in the local system.
- Port Description—The user-configured port description. The port description can be a maximum of 256 characters.
- System Name—The user-configured name of the local system. The system name can be a maximum of 256 characters.
- Switching Features Overview—This information is not configurable, but taken from the software.
- System Capabilities—The primary function performed by the system. The capabilities that system supports; for example, Ethernet switching or router. This information is not configurable, but based on the model of the product.
- Management Address—The IP management address of the local system.

The following LLDP-MED TLVs are supported:

- LLDP-MED Capabilities—A TLV that advertises the primary function of the port. The values range from 0 through 15:
  - 0—Capabilities
  - 1—Network policy
  - 2—Location identification
  - 3—Extended power through medium-dependent interface power-sourcing equipment (MDI-PSE)
  - 4—Inventory
  - 5–15—Reserved
- LLDP-MED Device Class Values:
  - 0—Class not defined
  - 1—Class 1 device
  - 2—Class 2 device
  - 3—Class 3 device

- 4—Network connectivity device
- 5–255— Reserved

> **NOTE**: Starting in Junos OS Release 15.1X49-D60 and Junos OS Release 17.3R1, Link Layer Discovery Protocol (LLDP) and LLDP-Media Endpoint Discovery (MFD) are enabled on SRX300, SRX320, SRX340, SRX345, SRX550M and SRX1500 devices.

- Network Policy—A TLV that advertises the port VLAN configuration and associated Layer 2 and Layer 3 attributes. Attributes include the policy identifier, application types, such as voice or streaming video, 802.1Q VLAN tagging, and 802.1p priority bits and Diffserv code points.
- Endpoint Location—A TLV that advertises the physical location of the endpoint.
- Extended Power via MDI—A TLV that advertises the power type, power source, power priority, and power value of the port. It is the responsibility of the PSE device (network connectivity device) to advertise the power priority on a port.

LLDP and LLDP-MED must be explicitly configured on uPIMs (in enhanced switching mode) on base ports on SRX100, SRX210, SRX240, SRX300, SRX320, SRX340, and SRX345 devices, and Gigabit Backplane Physical Interface Modules (GPIMs) on SRX650 devices. (Platform support depends on the Junos OS release in your installation.) To configure LLDP on all interfaces or on a specific interface, use the `lldp` statement at the [`set protocols`] hierarchy level. To configure LLDP-MED on all interfaces or on a specific interface, use the `lldp-med` statement at the [`set protocols`] hierarchy level.

Types of Switch Ports

The ports, or interfaces, on a switch operate in either access mode or trunk mode.

An interface in access mode connects to a network device, such as a desktop computer, an IP telephone, a printer, a file server, or a security camera. The interface itself belongs to a single VLAN. The frames transmitted over an access interface are normal Ethernet frames.

Trunk interfaces handle traffic for multiple VLANs, multiplexing the traffic for all those VLANs over the same physical connection. Trunk interfaces are generally used to interconnect switches to one another.

uPIM in a Daisy Chain

You cannot combine multiple uPIMs to act as a single integrated switch. However, you can connect uPIMs on the same chassis externally by physically connecting a port on one uPIM to a port on another uPIM in a daisy-chain fashion.

Two or more uPIMs daisy-chained together create a single switch with a higher port count than either individual uPIM. One port on each uPIM is used solely for the connection. For example, if you daisy-

chain a 6-port uPIM and an 8-port uPIM, the result operates as a 12-port uPIM. Any port of a uPIM can be used for daisy chaining.

Configure the IP address for only one of the daisy-chained uPIMs, making it the primary uPIM. The secondary uPIM routes traffic to the primary uPIM, which forwards it to the Routing Engine. This results in some increase in latency and packet drops due to oversubscription of the external link.

Only one link between the two uPIMs is supported. Connecting more than one link between uPIMs creates a loop topology, which is not supported.

Q-in-Q VLAN Tagging

Q-in-Q tunneling, defined by the IEEE 802.1ad standard, allows service providers on Ethernet access networks to extend a Layer 2 Ethernet connection between two customer sites.

In Q-in-Q tunneling, as a packet travels from a customer VLAN (C-VLAN) to a service provider's VLAN, a service provider-specific 802.1Q tag is added to the packet. This additional tag is used to segregate traffic into service-provider-defined service VLANs (S-VLANs). The original customer 802.1Q tag of the packet remains and is transmitted transparently, passing through the service provider's network. As the packet leaves the S-VLAN in the downstream direction, the extra 802.1Q tag is removed.

NOTE: When Q-in-Q tunneling is configured for a service provider's VLAN, all Routing Engine packets, including packets from the *routed VLAN interface*, that are transmitted from the customer-facing access port of that VLAN will always be untagged.

There are three ways to map C-VLANs to an S-VLAN:

- All-in-one bundling—Use the `dot1q-tunneling` statement at the [`edit vlans`] hierarchy level to map without specifying customer VLANs. All packets from a specific access interface are mapped to the S-VLAN.
- Many-to-one bundling—Use the `customer-vlans` statement at the [`edit vlans`] hierarchy level to specify which C-VLANs are mapped to the S-VLAN.
- Mapping C-VLAN on a specific interface—Use the `mapping` statement at the [`edit vlans`] hierarchy level to map a specific C-VLAN on a specified access interface to the S-VLAN.

Table 133 on page 1067 lists the C-VLAN to S-VLAN mapping supported on SRX Series Firewalls. (Platform support depends on the Junos OS release in your installation.)

Table 133: Supported Mapping Methods

Mapping	SRX210	SRX240	SRX300	SRX320	SRX340	SRX345	SRX550 M	SRX650
All-in-one bundling	Yes	Yes	No	No	Yes	Yes	Yes	Yes
Many-to-one bundling	No	No	No	No	Yes	Yes	Yes	Yes
Mapping C-VLAN on a specific interface	No	No	No	No	Yes	Yes	Yes	Yes

NOTE: VLAN translation is supported on SRX300 and SRX320 devices and these devices do not support Q-in-Q tunneling.

NOTE: On SRX650 devices, in the dot1q-tunneling configuration options, customer VLANs range and VLAN push do not work together for the same S-VLAN, even when you commit the configuration. If both are configured, then VLAN push takes priority over customer VLANs range.

IRB interfaces are supported on Q-in-Q VLANs for SRX210, SRX240, SRX340, SRX345, and SRX650 devices. Packets arriving on an IRB interface on a Q-in-Q VLAN are routed regardless of whether the packet is single or double tagged. The outgoing routed packets contain an S-VLAN tag only when exiting a trunk interface; the packets exit the interface untagged when exiting an access interface. (Platform support depends on the Junos OS release in your installation.)

In a Q-in-Q deployment, customer packets from downstream interfaces are transported without any changes to source and destination MAC addresses. You can disable MAC address learning at both the interface level and the VLAN level. Disabling MAC address learning on an interface disables learning for all the VLANs of which that interface is a member. When you disable MAC address learning on a VLAN, MAC addresses that have already been learned are flushed.

On SRX100, SRX210, SRX240, SRX300, SRX320, SRX340, SRX345, and SRX650 devices (with platform support depending on the Junos OS release in your installation), on the Layer 3 aggregated Ethernet, the following features are not supported:

- Encapsulations (such as CCC, VLAN CCC, VPLS, and PPPoE)
- J-Web
- Starting in Junos OS Release 19.4R2, you can configure the LLDP on redundant Ethernet (reth) interfaces. Use the `set protocol lldp interface <reth-interface>` command to configure LLDP on reth interface.
- On SRX550M devices the aggregate Ethernet (ae) interface with XE member interface cannot be configured with the Ethernet switching family.
- On SRX300, SRX320, SRX340, SRX345, and SRX550M devices, the Q-in-Q support on a Layer 3 interface has the following limitations:
  - Double tagging is not supported on reth and ae interfaces.
  - Multitopology routing is not supported in flow mode and in chassis clusters.
  - Dual tagged frames are not supported on encapsulations (such as CCC, TCC, VPLS, and PPPoE)
  - On Layer 3 logical interfaces, `input-vlan-map`, `output-vlan-map`, `inner-range`, and `inner-list` are not applicable
  - Only TPIDs with 0x8100 are supported, and the maximum number of tags is 2.
  - Dual tagged frames are accepted only for logical interfaces with IPV4 and IPv6 families.
- On SRX100, SRX210, SRX240, SRX300, SRX320, SRX340, SRX345, and SRX650 devices (with platform support depending on the Junos OS release in your installation), on the *routed VLAN interface* (RVI), the following features are not supported:
  - IS-IS (family ISO)
  - Encapsulations (Ether CCC, VLAN CCC, VPLS, PPPoE, and so on) on VLAN interfaces
  - CLNS
  - DVMRP
  - VLAN interface MAC change
  - G-ARP
  - Change VLAN-Id for VLAN interface

Example: Configuring Switching Modes on Security Devices

IN THIS SECTION

Requirements

Before you begin, see "Ethernet Ports Switching Overview for Security Devices" on page 1061.

Overview

IN THIS SECTION

In this example, you configure `chassis` and set the l2-learning protocol to global mode switching. You then set a physical port parameter on the l2-learning protocols.

Topology

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

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

```
set protocols l2-learning global-mode switching
set interfaces ge-0/0/1 unit 0 family ethernet-switching interface-mode access
```

Step-by-Step Procedure

To configure switching mode:

1. Set l2-learning protocol to global mode switching.

```
[edit protocols l2-learning]
user@host# set protocols l2-learning global-mode switching
```

2. Set a physical port parameter on the l2-learning protocols.

```
[edit]
user@host# set interfaces ge-0/0/1 unit 0 family ethernet-switching interface-mode access
```

3. If you are done configuring the device, commit the configuration.

```
[edit]
user@host# commit
```

Results

From configuration mode, confirm your configuration by entering the `show protocols` and `show interfaces` commands. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
[edit]
user@host# show protocols
l2-learning {
    global-mode switching;
}
```

```
[edit]
user@host# show interfaces
ge-0/0/1 {
    unit 0 {
        family ethernet-switching {
            interface-mode access;
        }
    }
}
```

If you are done configuring the device, enter `commit` from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying the Switching Mode

Purpose

Make sure that the switching mode is configured as expected.

Action

From operational mode, enter the `show ethernet-switching global-information` command.

```
user@host> show ethernet-switching global-information
```

```
Global Configuration:

MAC aging interval    : 300
MAC learning          : Enabled
MAC statistics        : Disabled
MAC limit Count       : 16383
MAC limit hit         : Disabled
MAC packet action drop: Disabled
MAC+IP aging interval : IPv4 - 1200 seconds
                        IPv6 - 1200 seconds
MAC+IP limit Count    : 393215
MAC+IP limit reached  : No
LE  aging time        : 1200
LE  VLAN aging time   : 1200
Global Mode           : Switching
RE state              : Master
```

Meaning

The sample output shows that the global mode switching is configured as expected.

Verifying the Ethernet switching on Interface ge-0/0/1

Purpose

Make sure that the Ethernet switching is configured as expected on interface ge-0/0/1.

Action

From operational mode, enter the `show interfaces ge-0/0/1 brief` command.

```
user@host> show interfaces ge-0/0/1 brief
```

```
Physical interface: ge-0/0/1, Enabled, Physical link is Down
  Link-level type: Ethernet, MTU: 1514, LAN-PHY mode, Speed: 1000mbps, Loopback: Disabled,
Source filtering: Disabled, Flow control: Disabled, Auto-negotiation: Enabled, Remote fault:
Online
  Device flags   : Present Running Down
  Interface flags: Hardware-Down SNMP-Traps Internal: 0x0
  Link flags     : None

  Logical interface ge-0/0/1.0
    Flags: Device-Down SNMP-Traps 0x0 Encapsulation: Ethernet-Bridge
    Security: Zone: Null
    eth-switch
```

Meaning

The sample output shows that the Ethernet switching is configured on interface ge-0/0/1 as expected .

Change History Table

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

Release	Description
15.1X49-D60	Starting in Junos OS Release 15.1X49-D60 and Junos OS Release 17.3R1, Link Layer Discovery Protocol (LLDP) and LLDP-Media Endpoint Discovery (MFD) are enabled on SRX300, SRX320, SRX340, SRX345, SRX550M, and SRX1500 devices.

35

CHAPTER

Configuring Ethernet Port VLANs in Switching Mode on Security Devices

IN THIS CHAPTER

Ethernet Port VLANs in Switching Mode on Security Devices

IN THIS SECTION

Understanding VLAN Retagging on Security Devices

VLAN retagging is not supported from Junos OS Release 15.1X49-D40 to Junos OS Release 15.1X49-D60.

Starting in Junos OS Release 15.1X49-D70, VLAN retagging in switching mode is supported on SRX300, SRX320, SRX340, SRX345, and SRX550M devices.

Starting in Junos OS Release 15.1X49-D80, VLAN retagging in switching mode is supported on SRX1500 devices.

To support VLAN retagging on SRX Series Firewalls, configure `vlan-rewrite` in transparent mode and configure `swap` in switching mode.

The VLAN identifier in packets arriving on a Layer 2 trunk port can be rewritten or *retagged* with a different internal VLAN identifier. VLAN retagging is a symmetric operation; upon exiting the same trunk port, the retagged VLAN identifier is replaced with the original VLAN identifier. VLAN retagging provides a way to selectively screen incoming packets and redirect them to a firewall or other security device without affecting other VLAN traffic.

VLAN retagging can be applied only to interfaces configured as Layer 2 trunk interfaces. These interfaces can include redundant Ethernet interfaces in a Layer 2 transparent mode within a *chassis cluster* configuration.

NOTE: If a trunk port is configured for VLAN retagging, untagged packets received on the port are not assigned a VLAN identifier with the VLAN retagging configuration. To

configure a VLAN identifier for untagged packets received on the physical interface, use the native-vlan-id statement.

To configure VLAN retagging for a Layer 2 trunk interface, specify a one-to-one mapping of the following:

- Incoming VLAN identifier—VLAN identifier of the incoming packet that is to be retagged. This VLAN identifier must not be the same VLAN identifier configured with the native-vlan-id statement for the trunk port.
- Internal VLAN identifier—VLAN identifier for the retagged packet. This VLAN identifier must be in the VLAN identifier list for the trunk port and must not be the same VLAN identifier configured with the native-vlan-id statement for the trunk port.

SEE ALSO

Ethernet Switching and Layer 2 Transparent Mode Overview | 5

Example: Configuring Layer 2 Logical Interfaces on Security Devices | 1031

Configuring VLAN Retagging on a Layer 2 Trunk Interface of a Security Device

VLAN retagging is a feature that works on IEEE standard 802.1Q virtual LAN tagging (VLAN tagging. VLAN retagging for SRX1500 devices is an enterprise style of VLAN retagging, in which a single command is sufficient on top of normal trunk configuration.

1. Create a Layer 2 trunk interface.

```
[edit]
user@host# set interfaces ge-3/0/0 unit 0 family ethernet-switching interface-mode trunk vlan
members 1-10
```

2. Configure VLAN retagging.

```
[edit]
user@host# set interfaces ge-3/0/0 unit 0 family ethernet-switching vlan-rewrite translate 11
2
```

Example: Configuring a Guest VLAN on a Security Device

IN THIS SECTION

This example shows how to configure a guest VLAN for limited network access or for Internet-only access to avoid compromising a company's security.

Guest VLANs are not supported from Junos OS Release 15.1X49-D40 to Junos OS Release 15.1X49-D60.

Requirements

Before you begin, verify that the interfaces that will be used are in switch mode. See "Example: Configuring Switching Modes on Security Devices" on page 1069 and "Understanding Switching Modes on Security Devices" on page 1060.

Overview

In this example, you configure a VLAN called visitor-vlan with a VLAN ID of 300. Then you set protocols and configure visitor-vlan as the guest VLAN.

Configuration

IN THIS SECTION

Procedure

Step-by-Step Procedure

To configure a guest VLAN:

1. Configure a VLAN.

```
[edit]
user@host# set vlans visitor-vlan vlan-id 300
```

2. Specify the guest VLAN.

```
[edit]
user@host# set protocols dot1x authenticator interface all guest-vlan visitor-vlan
```

3. If you are done configuring the device, commit the configuration.

```
[edit]
user@host# commit
```

Verification

To verify the configuration is working properly, enter the `show vlans` and `show protocols dot1x` commands.

36

CHAPTER

Configuring Secure Wire on Security Devices

IN THIS CHAPTER

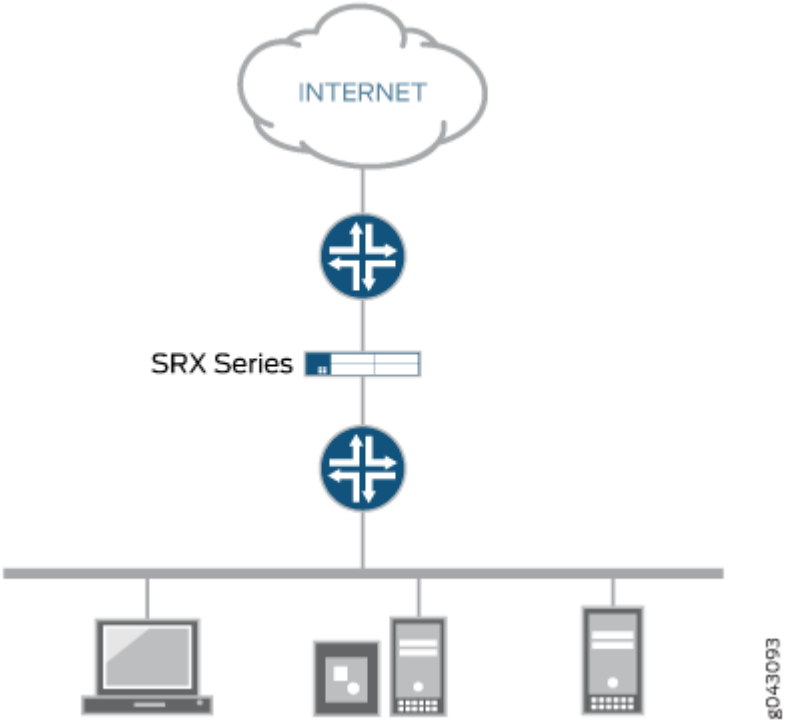
Secure Wire on Security Devices

IN THIS SECTION

Understanding Secure Wire on Security Devices

Traffic that arrives on a specific interface can be forwarded unchanged through another interface. This mapping of interfaces, called *secure wire*, allows an SRX Series to be deployed in the path of network traffic without requiring a change to routing tables or a reconfiguration of neighboring devices. Figure 55 on page 1081 shows a typical in-path deployment of an SRX Series with secure wire.

Figure 55: SRX Series In-Path Deployment with Secure Wire



Secure wire maps two peer interfaces. It differs from transparent and route modes in that there is no switching or routing lookup to forward traffic. As long as the traffic is permitted by a security policy, a packet arriving on one peer interface is immediately forwarded unchanged out of the other peer interface. There is no routing or switching decision made on the packet. Return traffic is also forwarded unchanged.

Secure wire mapping is configured with the `secure-wire` statement at the [`edit security forwarding-options`] hierarchy level; two Ethernet logical interfaces must be specified. The Ethernet logical interfaces must be configured with `family ethernet-switching` and each pair of interfaces must belong to the VLAN(s). The interfaces must be bound to security zones and a security policy configured to permit traffic between the zones.

This feature is available on Ethernet logical interfaces only; both IPv4 and IPv6 traffic are supported. You can configure interfaces for access or trunk mode. Secure wire supports chassis cluster redundant Ethernet interfaces. This feature does not support security features not supported in transparent mode, including NAT and IPsec VPN.

Secure wire supports Layer 7 features like AppSecure, SSL proxy, Content Security and IPS/IDP.

Secure wire is a special case of Layer 2 transparent mode on SRX Series Firewalls that provide point-to-point connections. This means that the two interfaces of a secure wire must ideally be directly

connected to Layer 3 entities, such as routers or hosts. Secure wire interfaces can be connected to switches. However, note that a secure wire interface forwards all arriving traffic to the peer interface only if the traffic is permitted by a security policy.

Secure wire can coexist with Layer 3 mode. While you can configure Layer 2 and Layer 3 interfaces at the same time, traffic forwarding occurs independently on Layer 2 and Layer 3 interfaces.

Secure wire can coexist with Layer 2 transparent mode. If both features exist on the same SRX Series Firewall, you need to configure them in different VLANs.

NOTE: Integrated routing and bridging (IRB) interfaces are not supported with secure wire.

SEE ALSO

Understanding Mixed Mode (Transparent and Route Mode) on Security Devices | **1033**

Example: Simplifying SRX Series Firewall Deployment with Secure Wire over Access Mode Interfaces

IN THIS SECTION

If you are connecting an SRX Series Firewall to other network devices, you can use secure wire to simplify the device deployment in the network. No changes to routing or forwarding tables on the SRX Series Firewall and no reconfiguration of neighboring devices is needed. Secure wire allows traffic to be forwarded unchanged between specified access mode interfaces on an SRX Series Firewall as long as it is permitted by security policies or other security features. Follow this example if you are connecting an SRX Series Firewall to other network devices through access mode interfaces.

This example shows how to configure a secure wire mapping for two access mode interfaces. This configuration applies to scenarios where user traffic is not VLAN tagged.

Requirements

No special configuration beyond device initialization is required before configuring this feature.

Overview

IN THIS SECTION

This example configures the secure wire access-sw that maps interface ge-0/0/3.0 to interface ge-0/0/4.0. The two peer interfaces are configured for access mode. The VLAN ID 10 is configured for the vlan-10 and the access mode interfaces.

NOTE: A specific VLAN ID must be configured for a VLAN.

Topology

Figure 56 on page 1083 shows the access mode interfaces that are mapped in secure wire access-sw.

Figure 56: Secure Wire Access Mode Interfaces

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

> **NOTE**: Starting in Junos OS Release 15.1X49-D10 and Junos OS Release 17.3R1, some Layer 2 CLI configuration statements are enhanced, and some commands are changed. For detailed information about the modified hierarchies, see "Enhanced Layer 2 CLI Configuration Statement and Command Changes for Security Devices" on page 39.
>
> The configuration statements shown below are for Junos OS Release 15.1X49-D10 or higher and Junos OS Release 17.3R1.

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

> **NOTE**: In switching mode, ethernet-switching interface must not be in security zone. You must enable transparent mode for ethernet-switching interfaces to be allowed in security zones by using the *global-mode (Protocols)* command.

```
set vlans vlan-10 vlan-id 10
set interfaces ge-0/0/3 unit 0 family ethernet-switching interface-mode access
set interfaces ge-0/0/3 unit 0 family ethernet-switching vlan members 10
set interfaces ge-0/0/4 unit 0 family ethernet-switching interface-mode access
set interfaces ge-0/0/4 unit 0 family ethernet-switching vlan members 10
set security forwarding-options secure-wire access-sw interface [ge-0/0/3.0 ge-0/0/4.0]
set security zones security-zone trust interfaces ge-0/0/3.0
set security zones security-zone untrust interfaces ge-0/0/4.0
set security address-book book1 address mail-untrust 203.0.113.1
set security address-book book1 attach zone untrust
set security address-book book2 address mail-trust 192.168.1.1
set security address-book book2 attach zone trust
set security policies from-zone trust to-zone untrust policy permit-mail match source-address
mail-trust
set security policies from-zone trust to-zone untrust policy permit-mail match destination-
address mail-untrust
set security policies from-zone trust to-zone untrust policy permit-mail match application junos-
mail
set security policies from-zone trust to-zone untrust policy permit-mail then permit
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure a secure wire mapping for access mode interfaces:

1. Configure the VLAN.

```
[edit vlans vlan-10]
user@host# set vlan-id 10
```

2. Configure the access mode interfaces.

```
[edit interfaces ]
user@host# set ge-0/0/3 unit 0 family ethernet-switching interface-mode access
user@host# set ge-0/0/4 unit 0 family ethernet-switching interface-mode access
user@host# set ge-0/0/3 unit 0 family ethernet-switching vlan members 10
user@host# set ge-0/0/4 unit 0 family ethernet-switching vlan members 10
```

3. Configure the secure wire mapping.

```
[edit security forwarding-options]
user@host# set secure-wire access-sw interface [ge-0/0/3.0 ge-0/0/4.0]
```

4. Configure security zones.

```
[edit security zones]
user@host# set security-zone trust interfaces ge-0/0/3.0
user@host# set security-zone untrust interfaces ge-0/0/4.0
```

5. Create address book entries. Attach security zones to the address books.

```
[edit security address-book book1]
user@host# set address mail-untrust 203.0.113.1
user@host# set attach zone untrust
```

```
[edit security address-book book1]
user@host# set address mail-trust 192.168.1.1
user@host# set attach zone trust
```

6. Configure a security policy to permit mail traffic.

```
[edit security policies from-zone trust to-zone untrust]
user@host# set policy permit-mail match source-address mail-trust
user@host# set policy permit-mail match destination-address mail-untrust
user@host# set policy permit-mail match application junos-mail
user@host# set policy permit-mail then permit
```

Results

From configuration mode, confirm your configuration by entering the `show vlans`, `show interfaces`, `show security forwarding-options`, and `show security zones` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@host# show vlans
    vlan-10 {
        vlan-id 10;
    }
    user@host# show interfaces
    ge-0/0/3 {
        unit 0 {
            family ethernet-switching {
                interface-mode access;
                vlan {
                    members 10;
                }
            }
        }
```

```
    }
    ge-0/0/4 {
        unit 0 {
            family ethernet-switching {
                interface-mode access;
                vlan {
                    members 10;
                }
            }
        }
    }
    user@host# show security forwarding-options
    secure-wire {
        access-sw {
            interface [ ge-0/0/3.0 ge-0/0/4.0 ];
        }
    }
user@host# show security zones
    security-zone trust {
        interfaces {
            ge-0/0/3.0;
        }
    }
    security-zone untrust {
        interfaces {
            ge-0/0/4.0;
        }
    }
user@host# show security policies
from-zone trust to-zone untrust {
    policy permit-mail {
        match {
            source-address mail-trust;
            destination-address mail-untrust;
            application junos-mail;
        }
        then {
            permit;
        }
    }
}
```

If you are done configuring the device, enter `commit` from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying Secure Wire Mapping

Purpose

Verify the secure wire mapping.

Action

From operational mode, enter the `show security forwarding-options secure-wire` command.

```
user@host> show security forward-options secure-wire
Secure wire                    Interface       Link   Interface       Link
access-sw                      ge-0/0/3.0      down   ge-0/0/4.0      down
Total secure wires: 1
```

Verifying the VLAN

Purpose

Verify the VLAN.

Action

From operational mode, enter the `show vlans vlan-10` command.

```
user@host> show vlans vlan-10
Routing instance        VLAN name            Tag     Interfaces
default-switch          vlan-10               10          ge-0/0/3.0
                                                          ge-0/0/4.0
```

Verifying Policy Configuration

Purpose

Verify information about security policies..

Action

From operational mode, enter the `show security policies detail` command.

```
user@host> show security policies detail
Default policy: deny-all
Pre ID default policy: permit-all
Policy: permit-mail, action-type: permit, State: enabled, Index: 4, Scope Policy: 0
  Policy Type: Configured
  Sequence number: 1
  From zone: trust, To zone: untrust
  Source vrf group:
    any
  Destination vrf group:
    any
  Source addresses:
    mail-trust(book2): 192.168.1.1/32
  Destination addresses:
    mail-untrust(book1): 203.0.113.1/32
  Application: junos-mail
    IP protocol: tcp, ALG: 0, Inactivity timeout: 1800
      Source port range: [0-0]
      Destination ports: 25
  Per policy TCP Options: SYN check: No, SEQ check: No, Window scale: No
```

SEE ALSO

Understanding Secure Wire on Security Devices

Ethernet Switching and Layer 2 Transparent Mode Overview

Example: Simplifying SRX Series Firewall Deployment with Secure Wire over Trunk Mode Interfaces

IN THIS SECTION

If you are connecting an SRX Series Firewall to other network devices, you can use secure wire to simplify the device deployment in the network. No changes to routing or forwarding tables on the SRX Series Firewall and no reconfiguration of neighboring devices is needed. Secure wire allows traffic to be forwarded unchanged between specified trunk mode interfaces on an SRX Series Firewall as long as it is permitted by security policies or other security features. Follow this example if you are connecting an SRX Series Firewall to other network devices through trunk mode interfaces.

Requirements

No special configuration beyond device initialization is required before configuring this feature.

Overview

IN THIS SECTION

This example configures the secure wire trunk-sw that maps interface ge-0/1/0.0 to interface ge-0/1/1.0. The two peer interfaces are configured for trunk mode and carry user traffic tagged with

VLAN IDs from 100 to 102. The VLAN ID list 100-102 is configured for the VLAN vlan-100 and the trunk mode interfaces.

NOTE: A specific VLAN ID must be configured for a VLAN.

Topology

Figure 57 on page 1091 shows the trunk mode interfaces that are mapped in secure wire trunk-sw.

Figure 57: Secure Wire Trunk Mode Interfaces

Configuration

IN THIS SECTION

- Procedure | **1091**

Procedure

CLI Quick Configuration

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

```
set vlans vlan-100 vlan members 100-102
set interfaces ge-0/1/0 unit 0 family ethernet-switching interface-mode trunk vlan members
100-102
set interfaces ge-0/1/1 unit 0 family ethernet-switching interface-mode trunk vlan members
100-102
```

```
set security forwarding-options secure-wire trunk-sw interface [ge-0/1/0.0 ge-0/1/1.0]
set security zones security-zone trust interfaces ge-0/1/0.0
set security zones security-zone untrust interfaces ge-0/1/1.0
set security policies default-policy permit-all
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure a secure wire mapping for trunk mode interfaces:

1. Configure the VLAN.

```
[edit vlans vlan-100]
user@host# set vlan members 100-102
```

2. Configure the trunk mode interfaces.

```
[edit interfaces]
user@host# set ge-0/1/0 unit 0 family ethernet-switching interface-mode trunk vlan members
100-102
user@host# set ge-0/1/1 unit 0 family ethernet-switching interface-mode trunk vlan members
100-102
```

3. Configure the secure wire mapping.

```
[edit security forwarding-options]
user@host# set secure-wire trunk-sw interface [ge-0/1/0.0 ge-0/1/1.0]
```

4. Configure security zones.

```
[edit security zones]
user@host# set security-zone trust interfaces ge-0/1/0.0
user@host# set security-zone untrust interfaces ge-0/1/1.0
```

5. Configure a security policy to permit traffic.

```
[edit security policies]
user@host# set default-policy permit-all
```

Results

From configuration mode, confirm your configuration by entering the `show vlans`, `show interfaces`, `show security forwarding-options`, and `show security zones` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@host# show vlans
    vlan-100 {
        vlan members 100-102;
    }
    user@host# show interfaces
    ge-0/1/0 {
        unit 0 {
            family ethernet-switching {
                interface-mode trunk;
                vlan members 100-102;
            }
        }
    }
    ge-0/1/1 {
        unit 0 {
            family ethernet-switching {
                interface-mode trunk;
                vlan members 100-102;
            }
        }
    }
    user@host# show security forwarding-options
    secure-wire trunk-sw {
        interfaces [ge-0/1/0.0 ge-0/1/1.0];
    }
user@host# show security zones
    security-zone trust {
        interfaces {
            ge-0/1/0.0;
```

```
        }
    }
    security-zone untrust {
        interfaces {
            ge-0/1/1.0;
        }
    }
```

If you are done configuring the device, enter `commit` from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying Secure Wire Mapping

Purpose

Verify the secure wire mapping.

Action

From operational mode, enter the `show security forwarding-options secure-wire` command.

```
user@host> show security forward-options secure-wire
Secure wire                    Interface       Link   Interface       Link
trunk-sw                       ge-0/1/0.0      up     ge-0/1/1.0      up
Total secure wires: 1
```

Verifying the VLAN

Purpose

Verify the VLAN.

Action

From operational mode, enter the `show vlans` command.

```
user@host> show vlans
Routing instance        VLAN name            VLAN ID     Interfaces
default-switch          vlan-100-vlan-0100         100         ge-0/1/0.0
                                                               ge-0/1/1.0
default-switch          vlan-100-vlan-0101         101         ge-0/1/0.0
                                                               ge-0/1/1.0
default-switch          vlan-100-vlan-0102         102         ge-0/1/0.0
                                                               ge-0/1/1.0
```

NOTE: VLANs are automatically expanded, with one VLAN for each VLAN ID in the VLAN ID list.

SEE ALSO

Understanding Secure Wire on Security Devices

Example: Simplifying SRX Series Firewall Deployment with Secure Wire over Aggregated Interface Member Links

IN THIS SECTION

If you are connecting an SRX Series Firewall to other network devices, you can use secure wire to simplify the device deployment in the network. No changes to routing or forwarding tables on the SRX Series Firewall and no reconfiguration of neighboring devices is needed. Secure wire allows traffic to be forwarded unchanged between specified aggregated interface member links on an SRX Series Firewall as long as it is permitted by security policies or other security features. Follow this example if you are connecting an SRX Series Firewall to other network devices through aggregated interface member links.

NOTE: LACP is not supported. Secure wire mappings can be configured for member links of link bundles instead of directly mapping aggregated Ethernet interfaces. When the ports, or interfaces on SRX Series Firewall are in trunk mode, the device do not transmit the LACP PDUs and fails the LACP. You must add a native vlan to secure wire interfaces, to bring LACP up.

NOTE: On SRX210, SRX220, SRX240, SRX300, SRX320, SRX340, SRX345, SRX550, and SRX650 devices, when you create an aggregated interface with two or more ports and set the family to Ethernet switching, and if a link in the bundle goes down, the traffic forwarded through the same link will be rerouted two seconds later. This causes an outage for the traffic being sent to the link until reroute is complete.

Requirements

No special configuration beyond device initialization is required before configuring this feature.

Overview

IN THIS SECTION

This example configures secure wires for two aggregated Ethernet interface link bundles with two links each. Two separate secure wires ae-link1 and ae-link2 are configured using one link from each aggregated Ethernet link bundle. This static mapping requires that the two link bundles have the same number of links.

For link bundles, all logical interfaces of the secure wire mappings must belong to the same VLAN. VLAN ID 10 is configured for the VLAN vlan-10 and the logical interfaces. All logical interfaces of a link bundle must belong to the same security zone.

NOTE: A specific VLAN ID or VLAN ID list must be configured for a VLAN.

Topology

Figure 58 on page 1097 shows the aggregated interfaces that are mapped in secure wire configurations.

Figure 58: Secure Wire Aggregated Interfaces

Configuration

IN THIS SECTION

- Procedure | **1097**

Procedure

CLI Quick Configuration

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

```
set vlans vlan-10 vlan-id 10
set interfaces ge-0/0/0 unit 0 family ethernet-switching interface-mode access vlan-id 10
set interfaces ge-0/0/1 unit 0 family ethernet-switching interface-mode access vlan-id 10
set interfaces ge-0/1/0 unit 0 family ethernet-switching interface-mode access vlan-id 10
```

```
set interfaces ge-0/1/1 unit 0 family ethernet-switching interface-mode access vlan-id 10
set security forwarding-options secure-wire ae-link1-sw interface [ge-0/1/0.0 ge-0/1/1.0]
set security forwarding-options secure-wire ae-link2-sw interface [ge-0/0/0.0 ge-0/0/1.0]
set security zones security-zone trust interfaces ge-0/0/0.0
set security zones security-zone trust interfaces ge-0/1/0.0
set security zones security-zone untrust interfaces ge-0/0/1.0
set security zones security-zone untrust interfaces ge-0/1/1.0
set security policies default-policy permit-all
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure a secure wire mapping for aggregated interface member links:

1. Configure the VLAN.

```
[edit vlans vlan-10]
user@host# set vlan-id10
```

2. Configure the interfaces.

```
[edit interfaces ]
user@host# set ge-0/0/0 unit 0 family ethernet-switching interface-mode access vlan-id 10
user@host# set ge-0/0/1 unit 0 family ethernet-switching interface-mode access vlan-id 10
user@host# set ge-0/1/0 unit 0 family ethernet-switching interface-mode access vlan-id 10
user@host# set ge-0/1/1 unit 0 family ethernet-switching interface-mode access vlan-id 10
```

3. Configure the secure wire mappings.

```
[edit security forwarding-options]
user@host# set secure-wire ae-link1-sw interface [ ge-0/1/0.0 ge-0/1/1.0 ]
user@host# set secure-wire ae-link2-sw interface [ ge-0/0/0.0 ge-0/0/1.0 ]
```

4. Configure security zones.

```
[edit security zones]
user@host# set security-zone trust interfaces ge-0/0/0.0
```

```
user@host# set security-zone trust interfaces ge-0/1/0.0
user@host# set security-zone untrust interfaces ge-0/0/1.0
user@host# set security-zone untrust interfaces ge-0/1/1.0
```

5. Configure a security policy to permit traffic.

```
[edit security policies]
user@host# set default-policy permit-all
```

Results

From configuration mode, confirm your configuration by entering the `show vlans`, `show interfaces`, `show security forwarding-options`, and `show security zones` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@host# show vlans
    vlan-10 {
        vlan-id 10;
    }
    user@host# show interfaces
    ge-0/0/0 {
        unit 0 {
            family ethernet-switching {
                interface-mode access;
                vlan-id 10;
            }
        }
    }
    ge-0/0/1 {
        unit 0 {
            family ethernet-switching {
                interface-mode access;
                vlan-id 10;
            }
        }
    }
    ge-0/1/0 {
        unit 0 {
            family ethernet-switching {
                interface-mode access;
```

```
                vlan-id 10;
            }
        }
    }
    ge-0/1/1{
        unit 0 {
            family ethernet-switching {
                interface-mode access;
                vlan-id 10;
            }
        }
    }
    user@host# show security forwarding-options
    secure-wire ae-link1-sw {
        interfaces [ge-0/1/0.0 ge-0/1/1.0];
    }
    secure-wire ae-link2-sw {
        interfaces [ge-0/0/0.0 ge-0/0/1.0];
    }
user@host# show security zones
    security-zone trust {
        interfaces {
            ge-0/0/0.0;
            ge-0/1/0.0;
        }
    }
    security-zone untrust {
        interfaces {
            ge-0/0/1.0;
            ge-0/1/1.0;
        }
    }
```

If you are done configuring the device, enter `commit` from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying Secure Wire Mapping

Purpose

Verify the secure wire mapping.

Action

From operational mode, enter the `show security forwarding-options secure-wire` command.

```
user@host> show security forward-options secure-wire
Secure wire                     Interface       Link   Interface       Link
ae-link1-sw                     ge-0/1/0.0      up     ge-0/1/1.0      up
ae-link2-sw                     ge-0/0/0.0      up     ge-0/0/1.0      up
Total secure wires: 2
```

Verifying the VLAN

Purpose

Verify the VLAN.

Action

From operational mode, enter the `show vlans vlan-10` command.

```
user@host> show vlans vlan-10
Routing instance        VLAN name            VLAN ID     Interfaces
default-switch          vlan-10                    10          ge-0/0/0.0
                                                               ge-0/0/1.0
                                                               ge-0/1/0.0
                                                               ge-0/1/1.0
```

SEE ALSO

Understanding Secure Wire on Security Devices

Example: Simplifying Chassis Cluster Deployment with Secure Wire over Redundant Ethernet Interfaces

IN THIS SECTION

If you are connecting an SRX Series chassis cluster to other network devices, you can use secure wire to simplify the cluster deployment in the network. No changes to routing or forwarding tables on the cluster and no reconfiguration of neighboring devices is needed. Secure wire allows traffic to be forwarded unchanged between specified redundant Ethernet interfaces on the SRX Series chassis cluster as long as it is permitted by security policies or other security features. Follow this example if you are connecting an SRX Series chassis cluster to other network devices through redundant Ethernet interfaces.

Requirements

Before you begin:

- Connect a pair of the same SRX Series Firewalls in a chassis cluster.
- Configure the chassis cluster node ID and cluster ID.
- Set the number of redundant Ethernet interfaces in the chassis cluster.
- Configure the chassis cluster fabric.
- Configure chassis cluster redundancy group (in this example redundancy group 1 is used).

For more information, see the Chassis Cluster User Guide for SRX Series Devices.

Overview

IN THIS SECTION

Secure wire is supported over redundant Ethernet interfaces in a chassis cluster. The two redundant Ethernet interfaces must be configured in the same redundancy group. If failover occurs, both redundant Ethernet interfaces must fail over together.

NOTE: Secure wire mapping of redundant Ethernet link aggregation groups (LAGs) are not supported. LACP is not supported.

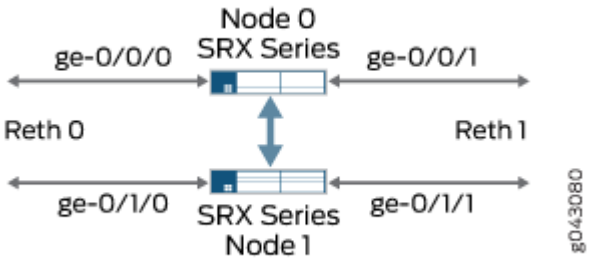
This example configures the secure wire reth-sw that maps ingress interface reth0.0 to egress interface reth1.0. Each redundant Ethernet interface consists of two child interfaces, one on each node of the chassis cluster. The two redundant Ethernet interfaces are configured for access mode. VLAN ID 10 is configured for the VLAN vlan-10 and the redundant Ethernet interfaces.

NOTE: A specific VLAN ID or VLAN ID list must be configured for a VLAN.

Topology

Figure 59 on page 1103 shows the redundant Ethernet interfaces that are mapped in secure wire reth-sw.

Figure 59: Secure Wire Redundant Ethernet Interfaces



Configuration

IN THIS SECTION

- Procedure | 1104

Procedure

CLI Quick Configuration

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

```
set vlans vlan-10 vlan-id 10
set interfaces ge-0/0/0 gigether-options redundant-parent reth0
set interfaces ge-0/0/1 gigether-options redundant-parent reth1
set interfaces ge-0/1/0 gigether-options redundant-parent reth0
set interfaces ge-0/1/1 gigether-options redundant-parent reth1
set interfaces reth0 unit 0 family ethernet-switching interface-mode access vlan-id 10
set interfaces reth1 unit 0 family ethernet-switching interface-mode access vlan-id 10
set interfaces reth0 redundant-ether-options redundancy-group 1
set interfaces reth1 redundant-ether-options redundancy-group 1
set security forwarding-options secure-wire reth-sw interface [reth0.0 reth1.0]
set security zones security-zone trust interfaces reth0.0
set security zones security-zone untrust interfaces reth1.0
set security policies default-policy permit-all
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure a secure wire mapping for chassis cluster redundant Ethernet interfaces:

1. Configure the VLAN.

```
[edit vlans vlan-10]
user@host# set vlan-id 10
```

2. Configure the redundant Ethernet interfaces.

```
[edit interfaces ]
user@host# set ge-0/0/0 gigether-options redundant-parent reth0
user@host# set ge-0/0/1 gigether-options redundant-parent reth1
user@host# set ge-0/1/0 gigether-options redundant-parent reth0
user@host# set ge-0/1/1 gigether-options redundant-parent reth1
user@host#set reth0 unit 0 family ethernet-switching interface-mode access vlan-id 10
user@host#set reth1 unit 0 family ethernet-switching interface-mode access vlan-id 10
user@host# set reth0 redundant-ether-options redundancy-group 1
user@host# set reth1 redundant-ether-options redundancy-group 1
```

3. Configure the secure wire mapping.

```
[edit security forwarding-options]
user@host# set secure-wire reth-sw interface [reth0.0 reth1.0]
```

4. Configure security zones.

```
[edit security zones]
user@host# set security-zone trust interfaces reth0.0
user@host# set security-zone untrust interfaces reth1.0
```

5. Configure a security policy to permit traffic.

```
[edit security policies]
user@host# set default-policy permit-all
```

Results

From configuration mode, confirm your configuration by entering the `show vlans`, `show interfaces`, `show security forwarding-options`, and `show security zones` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@host# show vlans
    vlan-10 {
        vlan-id 10;
    }
    user@host# show interfaces
    ge-0/0/0 {
        gigether-options {
            redundant-parent reth0;
        }
    }
    ge-0/0/1 {
        gigether-options {
            redundant-parent reth1;
        }
    }
    ge-0/1/0 {
        gigether-options {
            redundant-parent reth0;
        }
    }
    ge-0/1/1 {
        gigether-options {
            redundant-parent reth1;
        }
    }
    reth0 {
        redundant-ether-options {
            redundancy-group 1;
        }
        unit 0 {
            family ethernet-switching {
                interface-mode access;
                vlan-id 10;
            }
        }
    }
```

```
    reth1 {
        redundant-ether-options {
            redundancy-group 1;
        }
        unit 0 {
            family ethernet-switching {
                interface-mode access;
                vlan-id 10;
            }
        }
    }
    user@host# show security forwarding-options
    secure-wire reth-sw {
        interfaces [reth0.0 reth1.0];
    }
user@host# show security zones
    security-zone trust {
        interfaces {
            reth0.0;
        }
    }
    security-zone untrust {
        interfaces {
            reth1.0;
        }
    }
```

If you are done configuring the device, enter `commit` from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying Secure Wire Mapping

Purpose

Verify the secure wire mapping.

Action

From operational mode, enter the `show security forwarding-options secure-wire` command.

```
user@host> show security forward-options secure-wire
node0:
--------------------------------------------------------------------------
Secure wire                     Interface       Link   Interface       Link
reth-sw                         reth0.0         up     reth1.0         up

Total secure wires: 1

node1:
--------------------------------------------------------------------------
Secure wire                     Interface       Link   Interface       Link
reth-sw                         reth0.0         up     reth1.0         up

Total secure wires: 1
```

Verifying the VLAN

Purpose

Verify the VLAN.

Action

From operational mode, enter the `show vlan vlan-10` command.

```
user@host> show vlan vlan-10
Routing instance        VLAN Name            VLAN ID     Interfaces
default-switch          vlan-10                    10         reth0.0
                                                              reth1.0
```

SEE ALSO

Understanding Secure Wire on Security Devices

Example: Simplifying Chassis Cluster Deployment with Secure Wire over Aggregated Redundant Ethernet Interfaces

IN THIS SECTION

If you are connecting an SRX Series chassis cluster to other network devices, you can use secure wire to simplify the cluster deployment in the network. No changes to routing or forwarding tables on the cluster and no reconfiguration of neighboring devices is needed. Secure wire allows traffic to be forwarded unchanged between specified redundant Ethernet interfaces on the SRX Series chassis cluster as long as it is permitted by security policies or other security features. Follow this example if you are connecting an SRX Series chassis cluster to other network devices through aggregated redundant Ethernet interfaces.

NOTE: Secure wires cannot be configured for redundant Ethernet interface link aggregation groups (LAGs). For the secure wire mapping shown in this example, there is no LAG configuration on the SRX Series chassis cluster. Each redundant Ethernet interface consists of two child interfaces, one on each node of the chassis cluster. Users on upstream or downstream devices connected to the SRX Series cluster can configure the redundant Ethernet interface child links in LAGs.

Requirements

Before you begin:

- Connect a pair of the same SRX Series Firewalls in a chassis cluster.
- Configure the chassis cluster node ID and cluster ID.

- Set the number of redundant Ethernet interfaces in the chassis cluster.
- Configure the chassis cluster fabric.
- Configure the chassis cluster redundancy group (in this example, redundancy group 1 is used).

For more information, see the Chassis Cluster User Guide for SRX Series Devices.

Overview

IN THIS SECTION

This example configures secure wires for four redundant Ethernet interfaces: reth0, reth1, reth2, and reth3. Each redundant Ethernet interface consists of two child interfaces, one on each node of the chassis cluster. All four redundant Ethernet interfaces must be in the same VLAN—in this example, the VLAN is vlan-0. Two of the redundant Ethernet interfaces, reth0.0 and reth2.0, are assigned to the trust zone, while the other two interfaces, reth1.0 and reth3.0, are assigned to the untrust zone.

This example configures the following secure wires:

- reth-sw1 maps interface reth0.0 to interface reth1.0
- reth-sw2 maps interface reth2.0 to reth3.0

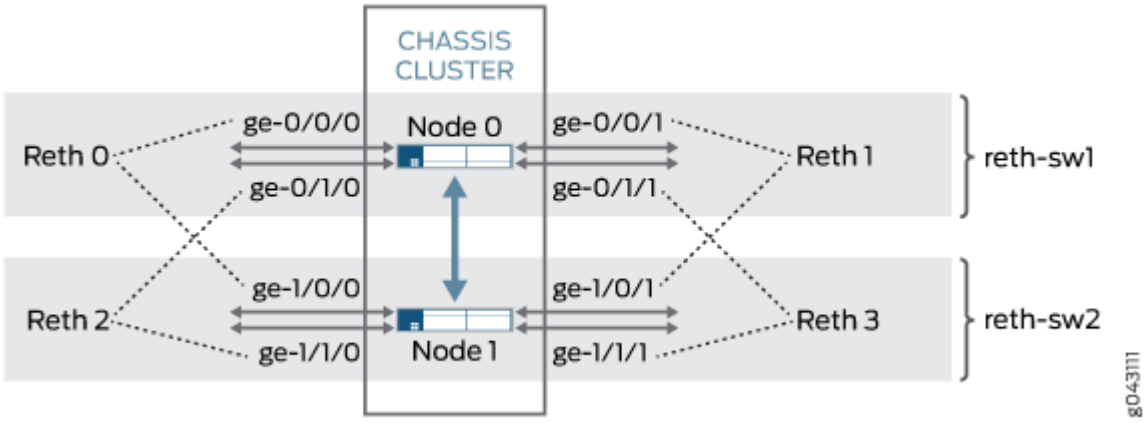
All redundant Ethernet interfaces are configured for access mode. VLAN ID 10 is configured for the VLAN vlan-0 and the redundant Ethernet interfaces.

NOTE: A specific VLAN ID or VLAN ID list must be configured for a VLAN.

Topology

Figure 60 on page 1111 shows the redundant Ethernet interface child links that are mapped in secure wire configurations reth-sw1 and reth-sw2. Each redundant Ethernet interface consists of two child interfaces, one on each node of the chassis cluster.

Figure 60: Secure Wire Redundant Ethernet Interface Child Links



Users on upstream or downstream devices connected to the SRX Series cluster can configure redundant Ethernet interface child links in a LAG as long as the LAG does not span chassis cluster nodes. For example, ge-0/0/0 and ge-0/1/0 and ge-0/0/1 and ge-0/1/1 on node 0 can be configured as LAGs on connected devices. In the same way, ge-1/0/0 and ge-1/1/0 and ge-1/0/1 and ge-1/1/1 on node 1 can be configured as LAGs on connected devices.

Configuration

IN THIS SECTION

- Procedure | 1111

Procedure

CLI Quick Configuration

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

```
set vlans vlan-0 vlan-id 10
set interfaces ge-0/0/0 gigether-options redundant-parent reth0
set interfaces ge-0/0/1 gigether-options redundant-parent reth1
```

```
set interfaces ge-0/1/0 gigether-options redundant-parent reth2
set interfaces ge-0/1/1 gigether-options redundant-parent reth3
set interfaces ge-1/0/0 gigether-options redundant-parent reth0
set interfaces ge-1/0/1 gigether-options redundant-parent reth1
set interfaces ge-1/1/0 gigether-options redundant-parent reth2
set interfaces ge-1/1/1 gigether-options redundant-parent reth3
set interfaces reth0 unit 0 family ethernet-switching interface-mode access vlan-id 10
set interfaces reth1 unit 0 family ethernet-switching interface-mode access vlan-id 10
set interfaces reth2 unit 0 family ethernet-switching interface-mode access vlan-id 10
set interfaces reth3 unit 0 family ethernet-switching interface-mode access vlan-id 10
set interfaces reth0 redundant-ether-options redundancy-group 1
set interfaces reth1 redundant-ether-options redundancy-group 1
set interfaces reth2 redundant-ether-options redundancy-group 1
set interfaces reth3 redundant-ether-options redundancy-group 1
set security forwarding-options secure-wire reth-sw1 interface [reth0.0 reth1.0]
set security forwarding-options secure-wire reth-sw2 interface [reth2.0 reth3.0]
set security zones security-zone trust interfaces reth0.0
set security zones security-zone trust interfaces reth2.0
set security zones security-zone untrust interfaces reth1.0
set security zones security-zone untrust interfaces reth3.0
set security policies default-policy permit-all
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure a secure wire mapping for aggregated interface member links:

1. Configure the VLAN.

```
[edit vlans vlan-0]
user@host# set vlan-id 10
```

2. Configure the redundant Ethernet interfaces.

```
[edit interfaces ]
user@host# set ge-0/0/0 gigether-options redundant-parent reth0
user@host# set ge-0/0/1 gigether-options redundant-parent reth1
user@host# set ge-0/1/0 gigether-options redundant-parent reth2
user@host# set ge-0/1/1 gigether-options redundant-parent reth3
```

```
user@host# set ge-1/0/0 gigether-options redundant-parent reth0
user@host# set ge-1/0/1 gigether-options redundant-parent reth1
user@host# set ge-1/1/0 gigether-options redundant-parent reth2
user@host# set ge-1/1/1 gigether-options redundant-parent reth3
user@host# set reth0 unit 0 family ethernet-switching interface-mode access vlan-id 10
user@host# set reth1 unit 0 family ethernet-switching interface-mode access vlan-id 10
user@host# set reth2 unit 0 family ethernet-switching interface-mode access vlan-id 10
user@host# set reth3 unit 0 family ethernet-switching interface-mode access vlan-id 10
user@host# set reth0 redundant-ether-options redundancy-group 1
user@host# set reth1 redundant-ether-options redundancy-group 1
user@host# set reth2 redundant-ether-options redundancy-group 1
user@host# set reth3 redundant-ether-options redundancy-group 1
```

3. Configure the secure wire mappings.

```
[edit security forwarding-options]
user@host# set secure-wire reth-sw1 interface [reth0.0 reth1.0]
user@host# set secure-wire reth-sw2 interface [reth2.0 reth3.0]
```

4. Configure security zones.

```
[edit security zones]
user@host# set security-zone trust interfaces reth0.0
user@host# set security-zone trust interfaces reth2.0
user@host# set security-zone untrust interfaces reth1.0
user@host# set security-zone untrust interfaces reth3.0
```

5. Configure a security policy to permit traffic.

```
[edit security policies]
user@host# set default-policy permit-all
```

Results

From configuration mode, confirm your configuration by entering the `show vlans`, `show interfaces`, `show security forwarding-options`, and `show security zones` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@host# show vlans
    vlan-0 {
        vlan-id 10;
    }
    user@host# show interfaces
    ge-0/0/0 {
        gigether-options {
            redundant-parent reth0;
        }
    }
    ge-0/0/1 {
        gigether-options {
            redundant-parent reth1;
        }
    }
    ge-0/1/0 {
        gigether-options {
            redundant-parent reth2;
        }
    }
    ge-0/1/1 {
        gigether-options {
            redundant-parent reth3;
        }
    }
    ge-1/0/0 {
        gigether-options {
            redundant-parent reth0;
        }
    }
    ge-1/0/1 {
        gigether-options {
            redundant-parent reth1;
        }
    }
    ge-1/1/0 {
```

```
        gigether-options {
            redundant-parent reth2;
        }
    }
    ge-1/1/1 {
        gigether-options {
            redundant-parent reth3;
        }
    }
    reth0 {
        redundant-ether-options {
            redundancy-group 1;
        }
        unit 0 {
            family ethernet-switching {
                interface-mode access;
                vlan-id 10;
            }
        }
    }
    reth1 {
        redundant-ether-options {
            redundancy-group 1;
        }
        unit 0 {
            family ethernet-switching {
                interface-mode access;
                vlan-id 10;
            }
        }
    }
    reth2 {
        redundant-ether-options {
            redundancy-group 1;
        }
        unit 0 {
            family ethernet-switching {
                interface-mode access;
                vlan-id 10;
            }
        }
    }
    reth3 {
```

```
        redundant-ether-options {
            redundancy-group 1;
        }
        unit 0 {
            family ethernet-switching {
                interface-mode access;
                vlan-id 10;
            }
        }
    }
    user@host# show security forwarding-options
    secure-wire reth-sw1 {
        interfaces [reth0.0 reth1.0];
    }
    secure-wire reth-sw2 {
        interfaces [reth2.0 reth3.0];
    }
user@host# show security zones
    security-zone trust {
        interfaces {
            reth0.0;
            reth2.0;
        }
    }
    security-zone untrust {
        interfaces {
            reth1.0;
            reth3.0;
        }
    }
```

If you are done configuring the device, enter `commit` from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying Secure Wire Mapping

Purpose

Verify the secure wire mapping.

Action

From operational mode, enter the `show security forwarding-options secure-wire` command.

```
user@host> show security forward-options secure-wire
node0:
--------------------------------------------------------------------------
Secure wire                       Interface       Link   Interface       Link
reth-sw1                          reth0.0         up     reth1.0         up
reth-sw2                          reth2.0         up     reth3.0         up

Total secure wires: 2

node1:
--------------------------------------------------------------------------
Secure wire                       Interface       Link   Interface       Link
reth-sw1                          reth0.0         up     reth1.0         up
reth-sw2                          reth2.0         up     reth3.0         up

Total secure wires: 2
```

Verifying VLAN

Purpose

Verify the VLAN.

Action

From operational mode, enter the `show vlans vlan-0` command.

```
user@host> show vlans vlan-0
Routing instance        VLAN name            VLAN ID     Interfaces
default-switch          vlan-0                      10
                                                        reth0.0
                                                        reth1.0
                                                        reth2.0
                                                        reth3.0
```

SEE ALSO

- Understanding Secure Wire on Security Devices

37

CHAPTER

Configuring Reflective Relay on Switches

IN THIS CHAPTER

Reflective Relay on Switches

IN THIS SECTION

Understanding Reflective Relay for Use with VEPA Technology

IN THIS SECTION

Virtual Ethernet Port Aggregator (VEPA) technology aggregates packets generated by virtual machines located on the same server and relays them to a physical switch. The physical switch then provides connectivity between the virtual machines located on the server, so the virtual machines do not communicate with one another. Offloading switching activities from a virtual switch to a physical switch reduces the computing overhead on the virtual servers and takes advantage of the security, filtering, and management features of the physical switch. Reflective relay, also known as “hairpin turn,” enables the physical switch to receive aggregated packets from the virtual machines hosted on the server through the VEPA on the downstream port and send those packets out the same downstream port from which the physical switch received them.

Benefits of VEPA and Reflective Relay

- Reduces the computing overhead on the virtual servers and takes advantage of the security, filtering, and management features of the physical switch.
- Enables the physical switch to receive aggregated packets from the virtual machines hosted on the server through the VEPA on the downstream port and send those packets out the same downstream port from which the physical switch received them.

VEPA

Even though virtual machines are capable of sending packets directly to one another, it is more efficient to pass these aggregated packets from the VEPA to a physical switch. The switch can then send any packets destined for a virtual machine located on the same server to the VEPA.

Reflective Relay

Reflective relay, also known as a “hairpin turn” or “hairpin mode,” returns aggregated packets to the VEPA by using the same downstream port that initially delivered the aggregated packets from the VEPA to the switch. Reflective relay must be configured on the interface located on the physical switch that receives aggregated packets, such as VEPA packets, because some of these packets might need to be sent back to the server if they are destined for another virtual machine on the same server.

Reflective relay only occurs in two situations:

- When the destination address of the packet was learned on that downstream port
- When the destination has not yet been learned

Reflective relay does not otherwise change the operation of the switch. If the interface to which the virtual machine is connected and the MAC address of the virtual machine packet are not yet included in the Ethernet switching table for the virtual machine’s associated VLAN, an entry is added. If the source MAC address of an incoming packet under the respective VLAN is not yet present in the Ethernet switching table, the switch floods the packet on all the other ports that are members of the same VLAN, including the port on which the packet arrived.

SEE ALSO

Understanding Bridging and VLANs on Switches | 140

Configuring Reflective Relay on Switches

Configure reflective relay when a switch port must return packets on a downstream port. For example, configure reflective relay when a switch port receives aggregated virtual machine packets from a technology such as virtual Ethernet port aggregator (VEPA). When these packets are passed through the switch, reflective relay allows the switch to send those packets back on the same interface that was used for delivery.

NOTE: This task uses Junos OS for QFX3500 and QFX3600 switches that do not support the Enhanced Layer 2 Software (ELS) configuration style. If your switch runs software that supports ELS, see "Configuring Reflective Relay on Switches with ELS Support" on page 1129.

Before you begin configuring reflective relay, ensure that you have:

- Configured packet aggregation on the server connected to the port. See your server documentation.
- Configured the port for all VLANs that could be included in aggregated packets..

To configure reflective relay:

1. Configure an Ethernet interface with a port mode of **tagged-acces**:

```
[edit]
user@switch# set interfaces  interface-name unit number family family-type port-mode tagged-
access
```

   For example:

```
[edit]
user@switch# set interfaces xe-0/0/2 unit 0 family ethernet-switching port-mode tagged-access
```

2. Configure the interface for reflective relay:

```
[edit]
user@switch# set interfaces  interface-name unit number family family-type reflective-relay
```

For example:

```
[edit]
user@switch# set interfaces xe-0/0/2 unit 0 family ethernet-switching reflective-relay
```

3. Configure the interface for the VLANs that exist on the VM server:

```
[edit]
user@switch# set interfaces  interface-name  unit number  family family-type vlan members
vlan-names
```

For example:

```
[edit]
user@switch# set interfaces xe-0/0/2 unit 0 family ethernet-switching vlan members
[VLAN_Purple VLAN_Orange VLAN_Blue]
```

Example: Configuring Reflective Relay for Use with VEPA Technology on QFX Switches

IN THIS SECTION

Reflective relay must be configured on a switch that receives virtual machine aggregated packets, such as Virtual Ethernet Port Aggregator (VEPA) packets, because some of these packets might be sent back to the server destined for another virtual machine on the same server. Reflective relay returns those packets to the original device using the same downstream port that delivered the packets to the switch.

NOTE: This example uses Junos OS for QFX3500 and QFX3600 switches that do not support the Enhanced Layer 2 Software (ELS) configuration style. If your switch runs software that does support ELS, see "Example: Configuring Reflective Relay for Use with VEPA Technology on QFX Switches with ELS Support" on page 1131.

This example shows how to configure a switch port interface to return packets sent by VEPA on the downstream interface back to the server using the same downstream interface:

Requirements

This example uses the following hardware and software components:

- One QFX3500 switch
- One server
- Junos OS Release 12.1 or later for the QFX Series

Before you configure reflective relay on a switch port, be sure you have:

- Configured a server with six virtual machines, VM 1 through VM 6.
- Configured the server with three VLANS named VLAN_Purple, VLAN_Orange, and VLAN_Blue and added two virtual machines to each VLAN.
- Configured the same three VLANs named VLAN_Purple, VLAN_Orange, and VLAN_Blue on one interface.
- Installed and configured VEPA to aggregate the virtual machine packets.

Overview and Topology

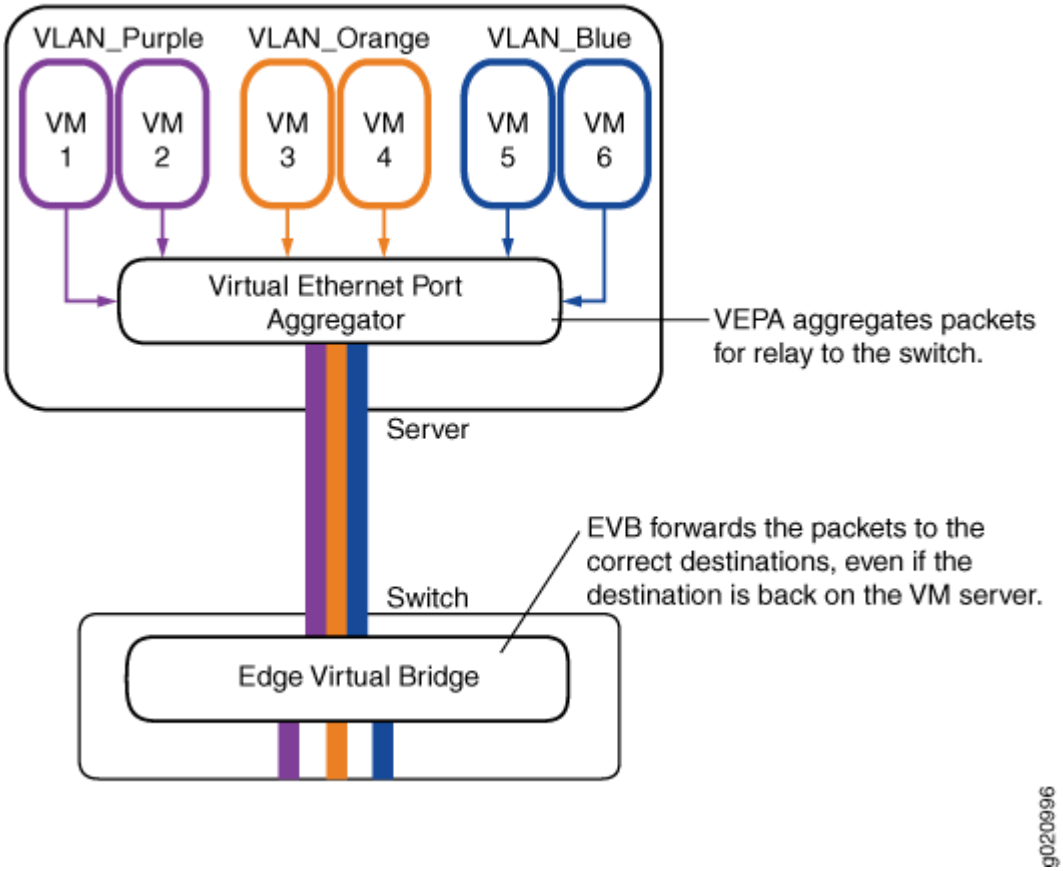
IN THIS SECTION

In this example, illustrated in Figure 61 on page 1125, a switch is connected to one server that is hosting six virtual machines and is configured with a VEPA for aggregating packets. The server's six virtual machines are VM 1 through VM 6, and each virtual machine belongs to one of the three server VLANs, VLAN_Purple, VLAN_Orange, or VLAN_Blue. Instead of the server directly passing packets between virtual machines, packets from any of the three VLANS that are destined for another one of the three

VLANs are aggregated using VEPA technology and passed to the switch for processing. You must configure the switch port to accept these aggregated packets on the downstream interface and to return appropriate packets to the server on the same downstream interface after they are processed. Figure 61 on page 1125 shows the topology for this example.

Topology

Figure 61: Reflective Relay Topology



In this example, you configure the physical Ethernet switch port interface for tagged-access port mode and reflective relay. Configuring tagged-access port mode allows the interface to accept VLAN tagged packets. Configuring reflective relay allows the downstream port to return those packets on the same interface. Table 134 on page 1126 shows the components used in this example.

Table 134: Components of the Topology for Configuring Reflective Relay

Component	Description
QFX3500 switch	Switch that supports reflective relay.
xe-0/0/2	Switch interface to the server.
Server	Server with virtual machines and VEPA technology.
Virtual machines	Six virtual machines located on the server: V1, V2, V3, V4, V5, and V6.
VLANs	Three VLANs: VLAN_Purple, VLAN_Orange, and VLAN_Blue. Each VLAN has two virtual machine members.
VEPA	Virtual Ethernet port aggregator that aggregates virtual machine packets on the server before the resulting single stream is transmitted to the switch.

Configuration

IN THIS SECTION

To configure reflective relay, perform these tasks:

Configuring Reflective Relay on the Port

CLI Quick Configuration

To quickly configure reflective relay, copy the following commands and paste them into the switch window:

```
[edit]
set interfaces xe-0/0/2 unit 0 family ethernet-switching port-mode tagged-access
```

```
set interfaces xe-0/0/2 unit 0 family ethernet-switching reflective-relay
set interfaces xe-0/0/2 unit 0 family ethernet-switching vlan members [VLAN_Blue VLAN_Orange
VLAN_Purple]
```

Step-by-Step Procedure

To configure reflective relay:

1. Configure the tagged-access port mode on the interface:

   NOTE: Configure the port mode as tagged-access otherwise you will receive an error when you commit the configuration.

   ```
   [edit]
   user@switch# set interfaces xe-0/0/2 unit 0 family ethernet-switching port-mode tagged-access
   ```

2. Configure reflective relay on the interface to allow it to both accept and send packets:

   ```
   [edit]
   user@switch# set interfaces xe-0/0/2 unit 0 family ethernet-switching reflective-relay
   ```

3. Configure the interface for the three VLANs on the server:

   ```
   [edit]
   user@switch# set interfaces xe-0/0/2 unit 0 family ethernet-switching vlan members
   [VLAN_Purple VLAN_Orange VLAN_Blue]
   ```

Results

Check the results of the configuration:

```
[edit interfaces xe-0/0/2]
user@switch# show
unit 0 {
        family ethernet-switching {
        port-mode tagged-access;
        reflective-relay;
```

```
        vlan {
            members [ VLAN_Purple VLAN_Orange VLAN_Blue ];
        }
    }
}
```

Verification

IN THIS SECTION

- Verifying That Reflective Relay Is Enabled and Working Correctly | **1128**

To confirm that reflective relay is enabled and working correctly, perform these tasks:

Verifying That Reflective Relay Is Enabled and Working Correctly

Purpose

Verify that reflective relay is enabled and working correctly.

Action

Use the `show ethernet-switching interfaces detail` command to display the reflective relay status:

```
user@switch> show ethernet-switching interfaces xe-0/0/2 detail
Interface: xe-0/0/2, Index: 66, State: down, Port mode: Tagged-access
Reflective Relay Status: Enabled
Ether type for the interface: 0x8100
VLAN membership:
    VLAN_Purple, 802.1Q Tag: 450, tagged, unblocked
    VLAN_Orange, 802.1Q Tag: 460, tagged, unblocked
    VLAN_Blue, 802.1Q Tag: 470, tagged, unblocked
Number of MACs learned on IFL: 0
```

Confirm that reflective relay is working by sending a Layer 2 broadcast message from one virtual machine to another virtual machine located on the same VLAN. Check the switch to verify that the switch sends the packets back on the same interface on which they were received. One way to check

this is to set up port mirroring on the switch interface, connect a traffic generator to the mirrored interface, and use the traffic generator to examine packets.

Alternatively, if you do not have a traffic generator available, you can send traffic between two virtual machines with FTP, Telnet, or SSH, while running the **tcpdump** utility on the receiver virtual machine port to capture reflected packets.

Meaning

The reflective relay status is **Enabled**, meaning that interface **xe-0/0/2** is configured for the tagged-access port mode, which accepts VLAN-tagged packets, and for reflective relay, which accepts and returns packets on the same interface.

When the traffic generator shows packets arriving at the switch and returning to the server on the same interface, reflective relay is working.

SEE ALSO

Configuring VLANs on QFX Series Switches without Enhanced Layer 2 Support | 152

Configuring Reflective Relay on Switches with ELS Support

Configure reflective relay when a switch port must return packets on a downstream port. For example, configure reflective relay when a switch port receives aggregated virtual machine packets from a technology such as virtual Ethernet port aggregator (VEPA). When these packets are passed through the switch, reflective relay allows the switch to send those packets back on the same interface that was used for delivery.

NOTE: This task uses Junos OS for QFX3500 and QFX3600 switches that supports the Enhanced Layer 2 Software (ELS) configuration style. If your switch runs software that does not support ELS, see "Configuring Reflective Relay on Switches" on page 1122.

Before you begin configuring reflective relay, ensure that you have:

- Configured packet aggregation on the server connected to the port. See your server documentation.
- Configured the port for all VLANs that could be included in aggregated packets..

To configure reflective relay:

1. Configure an Ethernet interface with an interface mode of **trunk**:

```
[edit]
user@switch# set interfaces  interface-name unit number family family-type interface-mode
trunk
```

For example:

```
[edit]
user@switch# set interfaces xe-0/0/2 unit 0 family ethernet-switching interface-mode trunk
```

2. Configure the interface for reflective relay:

```
[edit]
user@switch# set interfaces  interface-name unit number family family-type reflective-relay
```

For example:

```
[edit]
user@switch# set interfaces xe-0/0/2 unit 0 family ethernet-switching reflective-relay
```

3. Configure the interface for the VLANs that exist on the VM server:

```
[edit]
user@switch# set interfaces  interface-name  unit number  family family-type vlan members
vlan-names
```

For example:

```
[edit]
user@switch# set interfaces xe-0/0/2 unit 0 family ethernet-switching vlan members
[VLAN_Purple VLAN_Orange VLAN_Blue]
```

Example: Configuring Reflective Relay for Use with VEPA Technology on QFX Switches with ELS Support

IN THIS SECTION

Reflective relay must be configured on a switch that receives virtual machine aggregated packets, such as Virtual Ethernet Port Aggregator (VEPA) packets, because some of these packets might be sent back to the server destined for another virtual machine on the same server. Reflective relay returns those packets to the original device using the same downstream port that delivered the packets to the switch.

NOTE: This example uses Junos OS for QFX3500 and QFX3600 switches with support for the Enhanced Layer 2 Software (ELS) configuration style. If your switch runs software that does not support ELS, see "Example: Configuring Reflective Relay for Use with VEPA Technology on QFX Switches" on page 1123. For ELS details, see "Using the Enhanced Layer 2 Software CLI" on page 15.

This example shows how to configure a switch port interface to return packets sent by VEPA on the downstream interface back to the server using the same downstream interface:

Requirements

This example uses the following hardware and software components:

- One QFX3500 switch
- One server
- Junos OS Release 12.1 or later for the QFX Series

Before you configure reflective relay on a switch port, be sure you have:

- Configured a server with six virtual machines, VM 1 through VM 6.

- Configured the server with three VLANS named VLAN_Purple, VLAN_Orange, and VLAN_Blue and added two virtual machines to each VLAN.
- Configured the same three VLANs named VLAN_Purple, VLAN_Orange, and VLAN_Blue on one interface.
- Installed and configured VEPA to aggregate the virtual machine packets.

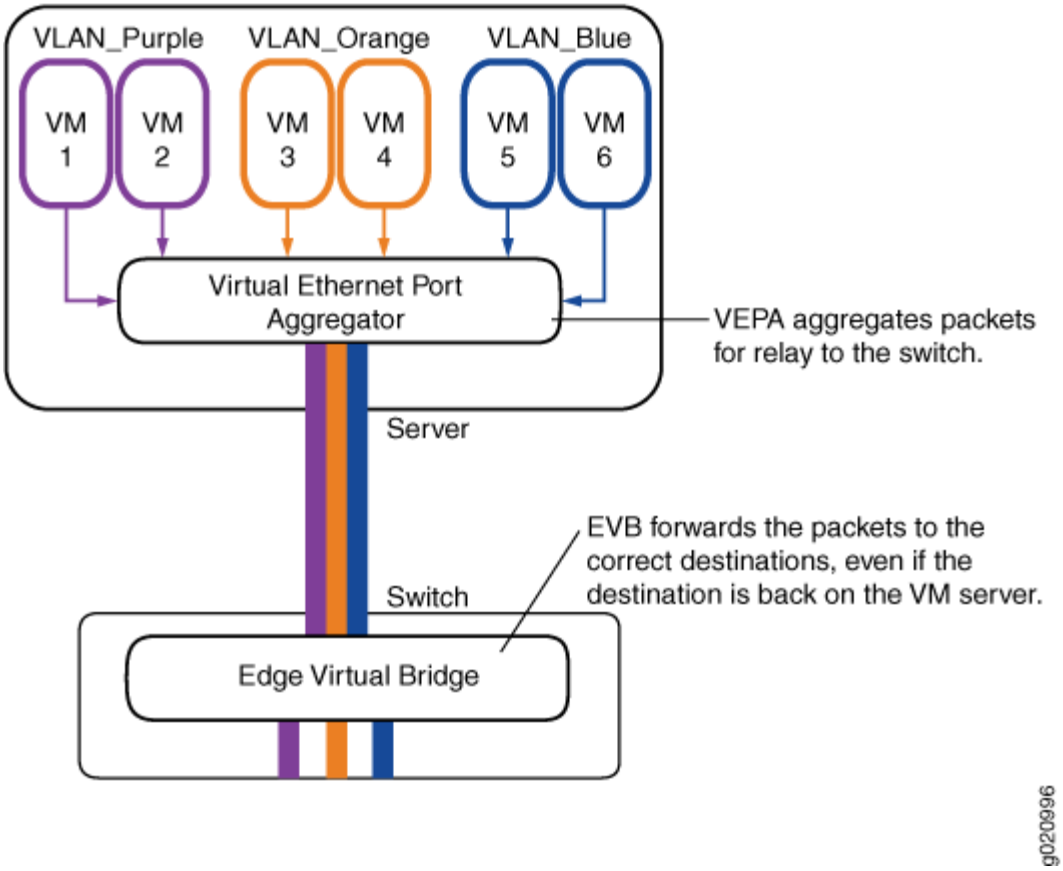
Overview and Topology

IN THIS SECTION

In this example, illustrated in Figure 62 on page 1133, a switch is connected to one server that is hosting six virtual machines and is configured with a VEPA for aggregating packets. The server's six virtual machines are VM 1 through VM 6, and each virtual machine belongs to one of the three server VLANs, VLAN_Purple, VLAN_Orange, or VLAN_Blue. Instead of the server directly passing packets between virtual machines, packets from any of the three VLANS that are destined for another one of the three VLANs are aggregated using VEPA technology and passed to the switch for processing. You must configure the switch port to accept these aggregated packets on the downstream interface and to return appropriate packets to the server on the same downstream interface after they are processed. Figure 62 on page 1133 shows the topology for this example.

Topology

Figure 62: Reflective Relay Topology



In this example, you configure the physical Ethernet switch port interface for trunk interface mode and reflective relay. Configuring trunk port mode allows the interface to accept VLAN tagged packets. Configuring reflective relay allows the downstream port to return those packets on the same interface. Table 135 on page 1133 shows the components used in this example.

Table 135: Components of the Topology for Configuring Reflective Relay

Component	Description
QFX3500 switch	Switch that supports reflective relay. .
xe-0/0/2	Switch interface to the server.

Table 135: Components of the Topology for Configuring Reflective Relay *(Continued)*

Component	Description
Server	Server with virtual machines and VEPA technology.
Virtual machines	Six virtual machines located on the server: V1, V2, V3, V4, V5, and V6.
VLANs	Three VLANs: VLAN_Purple, VLAN_Orange, and VLAN_Blue. Each VLAN has two virtual machine members.
VEPA	Virtual Ethernet port aggregator that aggregates virtual machine packets on the server before the resulting single stream is transmitted to the switch.

Configuration

IN THIS SECTION

To configure reflective relay, perform these tasks:

Configuring Reflective Relay on the Port

CLI Quick Configuration

To quickly configure reflective relay, copy the following commands and paste them into the switch window:

```
[edit]
set interfaces xe-0/0/2 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/2 unit 0 family ethernet-switching reflective-relay
set interfaces xe-0/0/2 unit 0 family ethernet-switching vlan members [VLAN_Blue VLAN_Orange
VLAN_Purple]
```

Step-by-Step Procedure

To configure reflective relay:

1. Configure the trunk interface mode on the interface:

```
[edit]
user@switch# set interfaces xe-0/0/2 unit 0 family ethernet-switching interface-mode trunk
```

2. Configure reflective relay on the interface to allow it to both accept and send packets:

```
[edit]
user@switch# set interfaces xe-0/0/2 unit 0 family ethernet-switching reflective-relay
```

3. Configure the interface for the three VLANs on the server:

```
[edit]
user@switch# set interfaces xe-0/0/2 unit 0 family ethernet-switching vlan members
[VLAN_Purple VLAN_Orange VLAN_Blue]
```

Results

Check the results of the configuration:

```
[edit interfaces xe-0/0/2]
user@switch# show
unit 0 {
        family ethernet-switching {
        interface-mode trunk;
        reflective-relay;
        vlan {
            members [ VLAN_Purple VLAN_Orange VLAN_Blue ];
        }
    }
}
```

Verification

IN THIS SECTION

To confirm that reflective relay is enabled and working correctly, perform these tasks:

Verifying That Reflective Relay Is Enabled and Working Correctly

Purpose

Verify that reflective relay is enabled and working correctly.

Action

Use the `show ethernet-switching interfaces detail` command to display the reflective relay status:

```
user@switch> show ethernet-switching interfaces xe-0/0/2 detail
Interface: xe-0/0/2, Index: 66, State: down, Interface mode: Trunk
Reflective Relay Status: Enabled
Ether type for the interface: 0x8100
VLAN membership:
    VLAN_Purple, 802.1Q Tag: 450, tagged, unblocked
    VLAN_Orange, 802.1Q Tag: 460, tagged, unblocked
    VLAN_Blue, 802.1Q Tag: 470, tagged, unblocked
Number of MACs learned on IFL: 0
```

Confirm that reflective relay is working by sending a Layer 2 broadcast message from one virtual machine to another virtual machine located on the same VLAN. Check the switch to verify that the switch sends the packets back on the same interface on which they were received. One way to check this is to set up port mirroring on the switch interface, connect a traffic generator to the mirrored interface, and use the traffic generator to examine packets.

Alternatively, if you do not have a traffic generator available, you can send traffic between two virtual machines with FTP, Telnet, or SSH, while running the **tcpdump** utility on the receiver virtual machine port to capture reflected packets.

Meaning

The reflective relay status is **Enabled**, meaning that interface **xe-0/0/2** is configured for the trunk interface mode, which accepts VLAN-tagged packets, and for reflective relay, which accepts and returns packets on the same interface.

When the traffic generator shows packets arriving at the switch and returning to the server on the same interface, reflective relay is working.

SEE ALSO

- Configuring Port Mirroring

38
CHAPTER

Configuring Edge Virtual Bridging

IN THIS CHAPTER

Edge Virtual Bridging

IN THIS SECTION

Understanding Edge Virtual Bridging for Use with VEPA Technology on EX Series Switches

IN THIS SECTION

Servers using virtual Ethernet port aggregator (VEPA) do not send packets directly from one virtual machine (VM) to another. Instead, the packets are sent to virtual bridges on an adjacent switch for processing. EX Series switches use edge virtual bridging (EVB) as a virtual bridge to return the packets on the same interface that delivered the packets.

What Is EVB?

EVB is a software capability on a switch running Junos OS that allows multiple virtual machines to communicate with each other and with external hosts in the Ethernet network environment.

What Is VEPA?

VEPA is a software capability on a server that collaborates with an adjacent, external switch to provide bridging support between multiple virtual machines and external networks. The VEPA collaborates with the adjacent switch by forwarding all VM-originated frames to the adjacent switch for frame processing and frame relay (including hairpin forwarding) and by steering and replicating frames received from the VEPA uplink to the appropriate destinations.

Why Use VEPA Instead of VEB?

Even though virtual machines are capable of sending packets directly to one another with a technology called virtual Ethernet bridging (VEB), you typically want to use physical switches for switching because VEB uses expensive server hardware to accomplish the task. Instead of using VEB, you can install VEPA on a server to offload switching functionality to an adjacent, less expensive physical switch. Additional advantages of using VEPA include:

- VEPA reduces complexity and allows higher performance at the server.
- VEPA takes advantage of the physical switch's security and tracking features.
- VEPA provides visibility of inter-virtual-machine traffic to network management tools designed for an adjacent bridge.
- VEPA reduces the amount of network configuration required by server administrators, and as a consequence, reduces work for the network administrator.

How Does EVB Work?

EVB uses two protocols, Virtual Station Interface (VSI) Discovery and Configuration Protocol (VDP) and Edge Control Protocol (ECP), to program policies for each individual virtual switch instance—specifically, EVB maintains the following information for each VSI instance:

- VLAN ID
- VSI type
- VSI type version
- MAC address of the server

VDP is used by the VEPA server to propagate VSI information to the switch. This allows the switch to program policies on individual VSIs and supports virtual machine migration by implementing logic to preassociate a VSI with a particular interface.

ECP is a Link Layer Discovery Protocol (LLDP)-like transport layer that allows multiple upper layer protocols to send and receive protocol data units (PDUs). ECP improves upon LLDP by implementing

sequencing, retransmission and an ack mechanism, while at the same time remaining lightweight enough to be implemented on a single-hop network. ECP is implemented in an EVB configuration when you configure LLDP on interfaces that you have configured for EVB. That is, you configure LLDP, not ECP.

How Do I Implement EVB?

You can configure EVB on a switch when that switch is adjacent to a server that includes VEPA technology. In general, this is what you do to implement EVB:

- The network manager creates a set of VSI types. Each VSI type is represented by a VSI type ID and a VSI version--the network manager can deploy one or more VSI versions at any given time.
- The VM manager configures VSI (which is a virtual station interface for a VM that is represented by a MAC address and VLAN ID pair) . To accomplish this, the VM manager queries available VSI type IDs (VTIDs) and creates a VSI instance consisting of a VSI Instance ID and the chosen VTID. This instance is known as VTDB and contains a VSI manager ID, a VSI type ID, a VSI version, and a VSI instance ID.

SEE ALSO

Understanding Bridging and VLANs on Switches | **140**

Example: Configuring Edge Virtual Bridging for Use with VEPA Technology on an EX Series Switch | **1144**

Configuring Edge Virtual Bridging on an EX Series Switch

Configure edge virtual bridging (EVB) when a switch is connected to a virtual machine (VM) server using virtual Ethernet port aggregator (VEPA) technology. EVB does not convert packets; rather, it ensures that packets from one VM destined for another VM on the same VM server is switched. In other words, when the source and destination of a packet are the same port, EVB delivers the packet properly, which otherwise would not happen.

NOTE: Configuring EVB also enables Virtual Station Interface (VSI) Discovery and Configuration Protocol (VDP).

Before you begin configuring EVB, ensure that you have:

- Configured packet aggregation on the server connected to the port that you will use on the switch for EVB. See your server documentation.

- Configured the EVB interface for all VLANs located on the virtual machines. See *Configuring VLANs for EX Series Switches*.

> **NOTE**: The port security features MAC move limiting and MAC limiting are supported on interfaces that are configured for EVB; however, the port security features IP source guard, dynamic ARP inspection (DAI), and DHCP snooping are not supported by EVB. For more information about these features, see *Port Security Features*.

To configure EVB on the switch:

1. Configure tagged-access mode for the interfaces on which you will enable EVB:

```
[edit interfaces interface-name]
user@switch# set unit 0 family ethernet-switching port-mode tagged-access
```

2. Enable the Link Layer Discovery Protocol (LLDP) on the interfaces on which you will enable EVB:.

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

3. Configure the interfaces for EVB as members of all VLANs located on the virtual machines.

```
[edit protocols]
user@switch# set vlans vlan-name vlan-id vlan-number
```

4. Enable VDP on the interfaces:

```
[edit protocols]
user@switch# set edge-virtual-bridging vsi-discovery interface interface-name
```

5. Define policies for VSI information, including a VSI manager ID, VSI type, VSI version, and VSI instance ID:

```
[edit policy-options]
user@switch# set vsi-policy policy-name from vsi-manager manager-number vsi-type type-
number vsi-version version-number vsi-instance instance-number
user@switch# set vsi-policy policy-name then filter filter-name
```

6. Define the firewall filters you mapped to in the previous step. When each incoming packet matches the filter, the count is incremented by 1. Other possible actions are accept and drop.

```
[edit firewall family ethernet-switching]
user@switch# set filter filter-name term term-name then action
```

7. Associate VSI policies with VDP:

```
[edit protocols]
user@switch# set edge-virtual-bridging vsi-discovery vsi-policy policy-name
```

8. Verify that the virtual machine successfully associated with the switch. After successful association of the VSI Profile with the switch interface, verify the learning of the VM's MAC address on MAC-Table or Forwarding database Table. The learn type of the VM's MAC addresses will be VDP, and upon successful shutdown of VM the corresponding MAC-VLAN entry will get flushed out from FDB table otherwise it will never shutdown.

```
admin@host# run show ethernet-switching table
```

9. Verify that VSI profiles are being learned at the switch:

```
user@switch# show edge-virtual-bridging vsi-profiles
```

10. Check the statistics of ECP packet exchanges between the switch and server:

```
user@switch# show edge-virtual-bridging ecp statistics
```

SEE ALSO

Example: Configuring Edge Virtual Bridging for Use with VEPA Technology on an EX Series Switch | 1144

Understanding Edge Virtual Bridging for Use with VEPA Technology on EX Series Switches | 1139

Example: Configuring Edge Virtual Bridging for Use with VEPA Technology on an EX Series Switch

IN THIS SECTION

Virtual machines (VMs) can use a physical switch that is adjacent to the VMs' server to send packets both to other VMs and to the rest of the network when two conditions have been met:

- Virtual Ethernet packet aggregator (VEPA) is configured on the VM server.
- Edge virtual bridging (EVB) is configured on the switch.

This example shows how to configure EVB on the switch so that packets can flow to and from the virtual machines.

Requirements

This example uses the following hardware and software components:

- One EX4500 or EX8200 switch
- Junos OS Release 12.1 or later for EX Series switches

Before you configure EVB on a switch, be sure you have configured the server with virtual machines, the VLANs, and VEPA:

NOTE: The following are the numbers of components used in this example, but you can use fewer or more to configure the feature.

- On the server, configure six virtual machines, VM 1 through VM 6 as shown in Figure 63 on page 1146. See your server documentation.
- On the server, configure three VLANs named VLAN_Purple, VLAN_Orange, and VLAN_Blue, and add two virtual machines to each VLAN. See your server documentation.

- On the server, install and configure VEPA to aggregate the virtual machine packets.
- On the switch, configure one interface with the same three VLANs as the server (VLAN_Purple, VLAN_Orange, and VLAN_Blue). See *Configuring VLANs for EX Series Switches*.

Overview and Topology

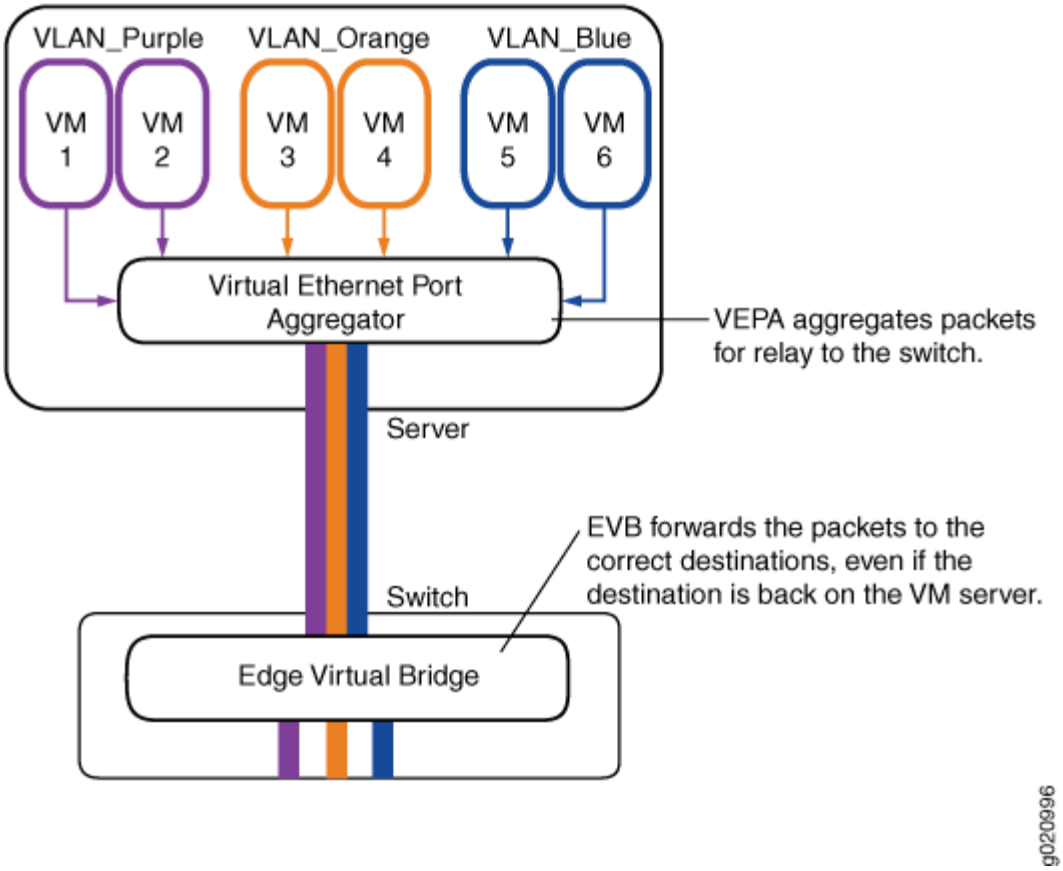
IN THIS SECTION

EVB is a software capability that provides multiple virtual end stations that communicate with each other and with external switches in the Ethernet network environment.

This example demonstrates the configuration that takes place on a switch when that switch is connected to a server with VEPA configured. In this example, a switch is already connected to a server hosting six virtual machines (VMs) and configured with VEPA for aggregating packets. The server's six virtual machines are VM 1 through VM 6, and each virtual machine belongs to one of the three server VLANs—VLAN_Purple, VLAN_Orange, or VLAN_Blue. Because VEPA is configured on the server, no two VMs can communicate directly—all communication between VMs must happen via the adjacent switch. Figure 63 on page 1146 shows the topology for this example.

Edge Virtual Bridging Example Topology

Figure 63: Topology



The VEPA component of the server pushes all packets from any VM, regardless of whether the packets are destined to other VMs on the same server or to any external host, to the adjacent switch. The adjacent switch applies policies to incoming packets based on the interface configuration and then forwards the packets to appropriate interfaces based on the MAC learning table. If the switch has not yet learned a destination MAC, it floods the packet to all interfaces, including the source port on which the packet arrived.

Table 136 on page 1146 shows the components used in this example.

Table 136: Components of the Topology for Configuring EVB

Component	Description
EX Series switch	For a list of switches that support this feature, see EX Series Switch Software Features Overview or EX Series Virtual Chassis Software Features Overview.

Table 136: Components of the Topology for Configuring EVB *(Continued)*

Component	Description
ge-0/0/20	Switch interface to the server.
Server	Server with virtual machines and VEPA technology.
Virtual machines	Six virtual machines located on the server, named VM 1, VM 2, VM 3, VM 4, VM 5, and VM 6.
VLANs	Three VLANs, named VLAN_Purple, VLAN_Orange, and VLAN_Blue. Each VLAN has two virtual machine members.
VEPA	A virtual Ethernet port aggregator (VEPA) is a software capability on a server that collaborates with an adjacent, external switch to provide bridging support between multiple virtual machines and with external networks. The VEPA collaborates with the switch by forwarding all VM-originated frames to the adjacent bridge for frame processing and frame relay (including hairpin forwarding) and by steering and replicating frames received from the VEPA uplink to the appropriate destinations.

NOTE: Configuring EVB also enables Virtual Station Interface (VSI) Discovery and Configuration Protocol (VDP).

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

To quickly configure EVB, copy the following commands and paste them into the switch's CLI at the `[edit]` hierarchy level.

```
set interfaces ge-0/0/20 unit 0 family ethernet-switching port-mode tagged-access
 set protocols lldp interface ge-0/0/20.0
set vlans vlan_purple interface ge-0/0/20.0
set vlans vlan_orange interface ge-0/0/20.0
set vlans vlan_blue interface ge-0/0/20.0
set protocols edge-virtual-bridging vsi-discovery interface ge-0/0/20.0
set policy-options vsi-policy P1 from vsi-manager 98 vsi-type 998 vsi-version 4 vsi-instance
09b11c53-8b5c-4eeb-8f00-c84ebb0bb998
set policy-options vsi-policy P1 then filter f2
set policy-options vsi-policy P3 from vsi-manager 97 vsi-type 997 vsi-version 3 vsi-instance
09b11c53-8b5c-4eeb-8f00-c84ebb0bb997
set policy-options vsi-policy P3 then filter f3
set firewall family ethernet-switching filter f2 term t1 then accept
set firewall family ethernet-switching filter f2 term t1 then count f2_accept
set firewall family ethernet-switching filter f3 term t1 then accept
set firewall family ethernet-switching filter f3 term t1 then count f3_accept
set protocols edge-virtual-bridging vsi-discovery vsi-policy P1
set protocols edge-virtual-bridging vsi-discovery vsi-policy P3
```

Step-by-Step Procedure

To configure EVB on the switch:

1. Configure tagged-access mode for the interfaces on which you will enable EVB:

```
[edit interfaces ge-0/0/20]
user@switch# set unit 0 family ethernet-switching port-mode tagged-access
```

2. Enable the Link Layer Discovery Protocol (LLDP) on the ports interfaces on which you will enable EVB:

```
[edit protocols]
user@switch# set lldp interface ge-0/0/20.0
```

3. Configure the interface as a member of all VLANs located on the virtual machines.

```
[edit]
user@switch# set vlans vlan_purple interface ge-0/0/20.0
user@switch# set vlans vlan_orange interface ge-0/0/20.0
user@switch# set vlans vlan_blue interface ge-0/0/20.0
```

4. Enable the VSI Discovery and Control Protocol (VDP) on the interface:

```
[edit protocols]
user@switch# set edge-virtual-bridging vsi-discovery interface ge-0/0/20.0
```

5. Define policies for VSI information. VSI information is based on a VSI manager ID, VSI type, VSI version, and VSI instance ID:

```
[edit policy-options]
user@switch# set vsi-policy P1 from vsi-manager 98 vsi-type 998 vsi-version 4 vsi-instance
09b11c53-8b5c-4eeb-8f00-c84ebb0bb998
user@switch# set vsi-policy P1 then filter f2
user@switch# set vsi-policy P3 from vsi-manager 97 vsi-type 997 vsi-version 3 vsi-instance
09b11c53-8b5c-4eeb-8f00-c84ebb0bb997
user@switch# set vsi-policy P3 then filter f3
```

6. Two VSI policies were defined in the previous step, each of them mapping to different firewall filters. Define the firewall filters:

```
[edit firewall family ethernet-switching]
user@switch# set filter f2 term t1 then accept
user@switch# set filter f2 term t1 then count f2_accept
user@switch# set filter f3 term t1 then accept
user@switch# set filter f3 term t1 then count f3_accept
```

7. Associate VSI policies with VSI-discovery protocol

```
[edit]
user@switch# set protocols edge-virtual-bridging vsi-discovery vsi-policy P1
user@switch# set protocols edge-virtual-bridging vsi-discovery vsi-policy P3
```

Results

```
user@switch# show protocols
edge-virtual-bridging {
    vsi-discovery {
        interface {
            ge-0/0/20.0;
        }
        vsi-policy {
            P1;
            P3;
        }
    }
}
lldp {
    interface ge-0/0/20.0;
```

```
user@switch# show policy-options
vsi-policy P1 {
    from {
        vsi-manager 98 vsi-type 998 vsi-version 4 vsi-instance 09b11c53-8b5c-4ee
b-8f00-c84ebb0bb998;
    }
    then {
        filter f2;
    }
}
vsi-policy P3 {
    from {
        vsi-manager 97 vsi-type 997 vsi-version 3 vsi-instance 09b11c53-8b5c-4ee
b-8f00-c84ebb0bb997;
    }
    then {
```

```
        filter f3;
    }
}
```

```
user@switch# show vlans
vlan_blue {
    interface {
        ge-0/0/20.0;
    }
}
vlan_orange {
    interface {
        ge-0/0/20.0;
    }
}
vlan_purple {
    interface {
        ge-0/0/20.0;
        interface;
    }
}
```

```
user@switch# show firewall
family ethernet-switching {
        filter f2 {
        term t1 {
            then {
                accept;
                count f2_accept;
            }
        }
    }
    filter f3 {
        term t1 {
            then {
                accept;
                count f3_accept;
            }
        }
```

```
    }
}
```

Verification

IN THIS SECTION

To confirm that EVB is enabled and working correctly, perform these tasks:

Verifying That EVB is Correctly Configured

Purpose

Verify that EVB is correctly configured

Action

```
user@switch# show edge-virtual-bridging
Interface     Forwarding Mode      RTE   Number of VSIs  Protocols
ge-0/0/20.0   Reflective-relay     25    400             ECP, VDP, RTE
```

Meaning

When LLDP is first enabled, an EVB LLDP exchange takes place between switch and server using LLDP. As part of this exchange the following parameters are negotiated: Number of VSIs supported, Forwarding mode, ECP support, VDP support, and Retransmission Timer Exponent (RTE). If the output has values for the negotiated parameters, EVB is correctly configured.

Verifying That the Virtual Machine Successfully Associated With the Switch

Purpose

Verify that the virtual machine successfully associated with the switch. After successful association of VSI Profile with the switch interface, verify the learning of the VM's MAC address on MAC-Table or Forwarding database Table. The learn type of the VM's MAC addresses will be VDP, and upon successful shutdown of VM the corresponding MAC-VLAN entry will get flushed out from FDB table otherwise it will never shutdown.

Action

```
user@switch# run show ethernet-switching table
Ethernet-switching table: 10 entries, 4 learned
VLAN    MAC address         Type     Age     Interfaces
v3     *              Flood        -      All-members
v3      00:02:a6:11:bb:1a    VDP          -      ge-1/0/10.0
v3      00:02:a6:11:cc:1a    VDP          -      ge-1/0/10.0
v3     00:23:9c:4f:70:01    Static       -      Router
v4      *                    Flood        -      All-members
v4      00:02:a6:11:bb:bb    VDP          -      ge-1/0/10.0
v4      00:23:9c:4f:70:01    Static       -      Router
v5      *                    Flood        -      All-members
v5      00:23:9c:4f:70:01    Static       -      Router
v5      52:54:00:d5:49:11    VDP          -      ge-1/0/20.0
```

Verifying That VSI Profiles Are Being Learned at the Switch

Purpose

Verify that VSI profiles are being learned at the switch.

Action

```
user@switch# show edge-virtual-bridging vsi-profiles
Interface: ge-0/0/20.0
Manager: 97, Type: 997, Version: 3, VSI State: Associate
 Instance: 09b11c53-8b5c-4eeb-8f00-c84ebb0bb997
```

```
MAC                     VLAN
00:10:94:00:00:04       3
```

Meaning

Whenever VMs configured for VEPA are started at the server, the VMs start sending VDP messages. As part of this protocol VSI profiles are learned at the switch.

If the output has values for Manager, Type, Version, VSI State, and Instance, VSI profiles are being learned at the switch.

SEE ALSO

Configuring Edge Virtual Bridging on an EX Series Switch | **1141**

Understanding Edge Virtual Bridging for Use with VEPA Technology on EX Series Switches | **1139**

39

CHAPTER

Troubleshooting Ethernet Switching

IN THIS CHAPTER

Troubleshooting Ethernet Switching

IN THIS SECTION

Problem

Description

Sometimes a MAC address entry in the switch's Ethernet switching table is not updated after the device with that MAC address has been moved from one interface to another on the switch. Typically, the switch does not wait for a MAC address expiration when a MAC move operation occurs. As soon as the switch detects the MAC address on the new interface, it immediately updates the table. Many network devices send a gratuitous ARP packet when switching an IP address from one device to another. The switch updates its ARP cache table after receipt of such gratuitous ARP messages, and then it also updates its Ethernet switching table.

Sometimes silent devices, such as syslog servers or SNMP trap receivers that receive UDP traffic but do not return acknowledgment (ACK) messages to the traffic source, fail to send gratuitous ARP packets when a device moves. If such a move occurs when the system administrator is not available to explicitly clear the affected interfaces by issuing the `clear ethernet-switching table` command, the entry for the moved device in the Ethernet switching table is not updated.

Solution

Set up the switch to handle unattended MAC address switchovers.

1. Reduce the system-wide ARP aging timer. (By default, the ARP aging timer is set at 20 minutes. The range of the ARP aging timer is from 1 through 240 minutes.)

```
[edit system arp]
user@switch# set aging-timer 3
```

2. Set the MAC aging timer to the same value as the ARP timer. (By default, the MAC aging timer is set to 300 seconds. The range is 60 to 1,000,000 seconds.)

```
[edit protocols l2-learning]
user@switch# set global-mac-table-aging-time 180
```

The ARP entry and the MAC address entry for the moved device expire within the times specified by the aging timer values. After the entries expire, the switch sends a new ARP message to the IP address of the device. The device responds to the ARP message, thereby refreshing the entries in the switch's ARP cache table and Ethernet switching table.

RELATED DOCUMENTATION

arp

global-mac-table-aging-time

Troubleshooting Ethernet Switching on EX Series Switches

IN THIS SECTION

Troubleshooting issues for Ethernet switching on EX Series switches:

MAC Address in the Switch's Ethernet Switching Table Is Not Updated After a MAC Address Move

IN THIS SECTION

Problem

Description

Sometimes a MAC address entry in the switch's Ethernet switching table is not updated after the device with that MAC address has been moved from one interface to another on the switch. Typically, the switch does not wait for a MAC address expiration when a MAC move operation occurs. As soon as the switch detects the MAC address on the new interface, it immediately updates the table. Many network devices send a gratuitous ARP packet when switching an IP address from one device to another. The switch updates its ARP cache table after receipt of such gratuitous ARP messages, and then it also updates its Ethernet switching table. However, sometimes silent devices, such as SYSLOG servers or SNMP Trap receivers that receive UDP traffic but do not return acknowledgement (ACK) messages to the traffic source, do not send gratuitous ARP packets when a device moves. If such a move occurs when the system administrator is not available to explicitly clear the affected interfaces by issuing the `clear ethernet-switching table` command, the entry for the moved device in the Ethernet switching table is not updated.

Solution

Set up the switch to handle unattended MAC address switchovers.

1. Reduce the system-wide ARP aging timer. (By default, the ARP aging timer is set at 20 minutes.)

```
[edit system arp]
user@switch# set aging-timer 3
```

2. Set the MAC aging timer to the same value as the ARP timer. (By default, the MAC aging timer is set to 300 seconds. The range is 15 to 1,000,000 seconds.)

```
[edit vlans]
user@switch# set vlans sales mac-table-aging-time 180
```

The ARP entry and the MAC address entry for the moved device expire within the times specified by the aging timer values. After the entries expire, the switch sends a new ARP message to the IP address of the device. The device responds to the ARP, thereby refreshing the entries in the switch's ARP cache table and Ethernet switching table

Change History Table

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

Release	Description
8.1	Starting with Junos OS Release 9.4 and later, the range of the ARP aging timer is from 1 through 240 minutes.

RELATED DOCUMENTATION

arp

mac-table-aging-time

40

CHAPTER

Configuration Statements and Operational Commands

IN THIS CHAPTER

Junos CLI Reference Overview

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

- Junos CLI Reference

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

- Configuration Statements
- Operational Commands