CONFIGURATION GUIDE

RUCKUS FastIron Layer 2 Switching Configuration Guide, 10.0.20

Supporting FastIron Software Release 10.0.20

Part Number: 53-1005803-02
Publication Date: 04 October 2024

Export Restrictions

These products and associated technical data (in print or electronic form) may be subject to export control laws of the United States of America. It is your responsibility to determine the applicable regulations and to comply with them. The following notice is applicable for all products or technology subject to export control:

These items are controlled by the U.S. Government and authorized for export only to the country of ultimate destination for use by the ultimate consignee or end-user(s) herein identified. They may not be resold, transferred, or otherwise disposed of, to any other country or to any person other than the authorized ultimate consignee or end-user(s), either in their original form or after being incorporated into other items, without first obtaining approval from the U.S. government or as otherwise authorized by U.S. law and regulations.

Disclaimer

THIS CONTENT AND ASSOCIATED PRODUCTS OR SERVICES ("MATERIALS"), ARE PROVIDED "AS IS" AND WITHOUT WARRANTIES OF ANY KIND, WHETHER EXPRESS OR IMPLIED. TO THE FULLEST EXTENT PERMISSIBLE PURSUANT TO APPLICABLE LAW, COMMSCOPE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, TITLE, NON-INFRINGEMENT, FREEDOM FROM COMPUTER VIRUS, AND WARRANTIES ARISING FROM COURSE OF DEALING OR COURSE OF PERFORMANCE. CommScope does not represent or warrant that the functions described or contained in the Materials will be uninterrupted or error-free, that defects will be corrected, or are free of viruses or other harmful components. CommScope does not make any warranties or representations regarding the use of the Materials in terms of their completeness, correctness, accuracy, adequacy, usefulness, timeliness, reliability or otherwise. As a condition of your use of the Materials, you warrant to CommScope that you will not make use thereof for any purpose that is unlawful or prohibited by their associated terms of use.

Limitation of Liability

IN NO EVENT SHALL COMMSCOPE, COMMSCOPE AFFILIATES, OR THEIR OFFICERS, DIRECTORS, EMPLOYEES, AGENTS, SUPPLIERS, LICENSORS AND THIRD PARTY PARTNERS, BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, PUNITIVE, INCIDENTAL, EXEMPLARY OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER, EVEN IF COMMSCOPE HAS BEEN PREVIOUSLY ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, WHETHER IN AN ACTION UNDER CONTRACT, TORT, OR ANY OTHER THEORY ARISING FROM YOUR ACCESS TO, OR USE OF, THE MATERIALS. Because some jurisdictions do not allow limitations on how long an implied warranty lasts, or the exclusion or limitation of liability for consequential or incidental damages, some of the above limitations may not apply to you.

Trademarks

CommScope and the CommScope logo are registered trademarks of CommScope and/or its affiliates in the U.S. and other countries. For additional trademark information see https://www.commscope.com/trademarks. All product names, trademarks, and registered trademarks are the property of their respective owners.

Patent Marking Notice

For applicable patents, see www.cs-pat.com.

Contents

Contact Information, Resources, and Conventions

Contacting RUCKUS Customer Services and Support

The Customer Services and Support (CSS) organization is available to provide assistance to customers with active warranties on their RUCKUS products, and to customers and partners with active support contracts.

For product support information and details on contacting the Support Team, go directly to the RUCKUS Support Portal using https://support.ruckuswireless.com, or go to https://www.ruckusnetworks.com and select **Support**.

What Support Do I Need?

Technical issues are usually described in terms of priority (or severity). To determine if you need to call and open a case or access the self-service resources, use the following criteria:

- Priority 1 (P1)—Critical. Network or service is down and business is impacted. No known workaround. Go to the **Submit a Case** section.
- Priority 2 (P2)—High. Network or service is impacted, but not down. Business impact may be high. Workaround may be available. Go to the **Submit a Case** section.
- Priority 3 (P3)—Medium. Network or service is moderately impacted, but most business remains functional. Click the **CONTACT** tab at the top of the page and explore the **Self-Service Online Help** options.
- Priority 4 (P4)—Low. Requests for information, product documentation, or product enhancements. Click the **CONTACT** tab at the top of the page and explore the **Self-Service Online Help** options.

Open a Case

When your entire network is down (P1), or severely impacted (P2), call the appropriate telephone number listed below to get help:

- Continental United States: 1-855-782-5871
- Canada: 1-855-782-5871
- Europe, Middle East, Africa, Central and South America, and Asia Pacific, toll-free numbers are available at https://support.ruckuswireless.com/contact-us and Live Chat is also available.
- Worldwide toll number for our support organization. Phone charges will apply: +1-650-265-0903

We suggest that you keep a physical note of the appropriate support number in case you have an entire network outage.

Self-Service Resources

The RUCKUS Support Portal at https://support.ruckuswireless.com offers a number of tools to help you to research and resolve problems with your RUCKUS products, including:

- Technical Documentation—https://support.ruckuswireless.com/documents
- Community Forums—https://community.ruckuswireless.com
- Knowledge Base Articles—https://support.ruckuswireless.com/answers
- Software Downloads and Release Notes—https://support.ruckuswireless.com/#products_grid
- Security Bulletins—https://support.ruckuswireless.com/security

Using these resources will help you to resolve some issues, and will provide the Technical Assistance Center (TAC) with additional data from your troubleshooting analysis if you still require assistance through a support case or Return Merchandise Authorization (RMA). If you still require help, open and manage your case at https://support.ruckuswireless.com/case_management.

Document Feedback

RUCKUS is interested in improving its documentation and welcomes your comments and suggestions.

You can email your comments to RUCKUS at #Ruckus-Docs@commscope.com.

When contacting us, include the following information:

- Document title and release number
- Document part number (on the cover page)
- Page number (if appropriate)

For example:

- RUCKUS SmartZone Upgrade Guide, Release 5.0
- Part number: 800-71850-001 Rev A
- Page 7

RUCKUS Product Documentation Resources

Visit the RUCKUS website to locate related documentation for your product and additional RUCKUS resources.

Release Notes and other user documentation are available at https://support.ruckuswireless.com/documents. You can locate the documentation by product or perform a text search. Access to Release Notes requires an active support contract and a RUCKUS Support Portal user account. Other technical documentation content is available without logging in to the RUCKUS Support Portal.

White papers, data sheets, and other product documentation are available at https://www.ruckusnetworks.com.

Online Training Resources

To access a variety of online RUCKUS training modules, including free introductory courses to wireless networking essentials, site surveys, and products, visit the RUCKUS Training Portal at https://commscopeuniversity.myabsorb.com/. The registration is a two-step process described in this video. Create a CommScope account and then register for, and request access for, CommScope University.

Document Conventions

The following table lists the text conventions that are used throughout this guide.

TABLE 1 Text Conventions

Convention	Description	Example
`monospace`	Identifies command syntax examples	`device(config)# interface ethernet 1/1/6`
bold	User interface (UI) components such as screen or page names, keyboard keys, software buttons, and field names	On the **Start** menu, click **All Programs**.
italics	Publication titles	Refer to the *RUCKUS Small Cell Release Notes* for more information.

Notes, Cautions, and Safety Warnings

Notes, cautions, and warning statements may be used in this document. They are listed in the order of increasing severity of potential hazards.

NOTE
A NOTE provides a tip, guidance, or advice, emphasizes important information, or provides a reference to related information.

ATTENTION
An ATTENTION statement indicates some information that you must read before continuing with the current action or task.

CAUTION
A CAUTION statement alerts you to situations that can be potentially hazardous to you or cause damage to hardware, firmware, software, or data.

DANGER
A DANGER statement indicates conditions or situations that can be potentially lethal or extremely hazardous to you. Safety labels are also attached directly to products to warn of these conditions or situations.

Command Syntax Conventions

Bold and italic text identify command syntax components. Delimiters and operators define groupings of parameters and their logical relationships.

Convention	Description
bold text	Identifies command names, keywords, and command options.
italic text	Identifies a variable.
[ ]	Syntax components displayed within square brackets are optional. Default responses to system prompts are enclosed in square brackets.
{**x**| **y**| **z**}	A choice of required parameters is enclosed in curly brackets separated by vertical bars. You must select one of the options.
x|**y**	A vertical bar separates mutually exclusive elements.
< >	Nonprinting characters, for example, passwords, are enclosed in angle brackets.
...	Repeat the previous element, for example, *member*[*member*...].
\	Indicates a “soft” line break in command examples. If a backslash separates two lines of a command input, enter the entire command at the prompt without the backslash.

About This Document

New in This Document

The following table describes information added or modified in this guide for FastIron 10.0.20.

TABLE 2 Summary of Enhancements in FastIron Release 10.0.20a

Feature	Description	Reference
Anycast Gateway with VXLAN	Anycast gateway with Virtual Extensible Local Area Network (VXLAN) allows the same gateway IP addresses to be used across all Virtual Tunnel End Points (VTEPs) in a VXLAN network. This ensures that every VTEP can act as the default gateway for the devices connected to it.	Anycast Gateway with VXLAN on page 359
BGP EVPN Multihoming[1]	For BGP-EVPN Multihoming (MH), a customer device or network (CE) can connect simultaneously to multiple edge nodes. This multi-attachment capability offers redundant, all-active connectivity to the EVPN core network and extends this redundancy across the entire network connected to the core.	BGP EVPN Multihoming on page 389
Rapid Spanning Tree Protocol (RSTP)	Support for RSTP (IEEE 802.1w) 2004 standard is introduced.	Rapid Spanning Tree Protocol on page 175
Stacking support for MCT on ICX 7550	Support for MCT on ICX 7550 switches including stacking is introduced	MCT Board Type Compatibility on page 97
Updates to address defects	**Updated**: Minor updates on content throughout to address defects.	All chapters
Minor editorial updates	**Updated**: Minor editorial updates were made throughout the Configuration Guide.	All chapters.

TABLE 3 Key Features and Enhancements in *FastIron 10.0.20*

Feature	Description	Reference
BGP EVPN	BGP Ethernet Virtual Private Network (EVPN) provides an efficient control plane protocol for VXLAN. When BGP EVPN is configured for a VXLAN, remote Virtual Tunnel End Points (VTEPs) and Virtual Network Identifiers (VNIs) do not need to be manually configured, and remote MAC addresses can be learned via the control plane. VTEPs in a VxLAN network can use BGP EVPN to exchange locally learned Layer 2 and Layer 3 destinations with remote VTEPs.	BGP EVPN on page 363

[1] A maximum of two Virtual Tunnel End Points (VTEPs) are supported to participate in multihoming (Dualhoming).

TABLE 3 Key Features and Enhancements in *FastIron 10.0.20* (continued)

Feature	Description	Reference
IPoE S-tag and C-Tag VLANs	When configuring selective Q-in Q tunneling, customer VLANs (CVLANs) can be added or replaced, enhancing traffic management capabilities in environments where complex VLAN handling is required. When Internet Protocol over Ethernet (IPoE) service-tagged (S-tag) and customer-tagged (C-tag) VLANs are enabled, traffic from different clients or tenants can be segregated, ensuring the efficient use of available VLAN IDs and simplifying the network configuration.	Adding or Replacing a Customer VLAN for Q-in-Q Tunneling on page 306
Layer 2 Trace Route Utility	This feature traces the traffic path through a specified device in a VLAN.	Layer 2 Trace Route Utility on page 328
Assigning different VLAN IDs to reserved VLANs	The **reserved-vlan-map** command now allows VLAN 4090 to be re-assigned in addition to VLANs 4091 and 4092.	Assigning a Different VLAN ID to Default and Reserved VLANs on page 280 Viewing Reassigned VLAN IDs for Reserved VLANs on page 281
Updates to address defects	**Updated**: Minor updates on content throughout to address defects.	All chapters
Minor editorial updates	**Updated**: Minor editorial updates were made throughout the Configuration Guide.	All chapters

Supported Hardware

This guide supports the following RUCKUS products:

- RUCKUS ICX 8200 Switches
- RUCKUS ICX 7850 Switches
- RUCKUS ICX 7650 Switches
- RUCKUS ICX 7550 Switches

For information about what models and modules these devices support, refer to the hardware installation guide for the specific product family.

Remote Fault Notification

Remote Fault Notification on 1 Gbps Fiber Connections

NOTE
Remote Fault Notification (RFN) is only available for 1 Gbps ethernet fiber ports. It is not available for 10G/100G ports or 1 Gbps ethernet copper ports.

For fiber-optic connections, you can optionally configure a transmit port to notify the receive port on the remote device whenever the transmit port becomes disabled.

When you enable this feature, the transmit port notifies the remote port whenever the fiber cable is either physically disconnected or has failed. When this occurs and the feature is enabled, the device disables the link and turns off both LEDs associated with the ports.

For more information about the parameters supported with the gig-default command, see "Changing the Gbps fiber negotiation mode" section in the *RUCKUS FastIron Monitoring Configuration Guide*.

By default, RFN is enabled. You can configure RFN as follows:

- On a trunk group
- On an individual interface

Enabling Remote Fault Notification

RFN configures the transmit port to notify the remote port whenever the fiber cable is either physically disconnected or has failed.

RFN is enabled (set to **auto-gig** option) by default.

1. Enter interface configuration mode to enable RFN on a particular interface.

   ```
   device(config)# interface ethernet 1/1/1
   ```

2. Re-enable RFN if it was disabled using the **neg-off** option.

   ```
   device(config-if-e1000-1/1/1)# gig-default auto-gig
   ```

   To disable RFN, use the **gig-default neg-off** command.

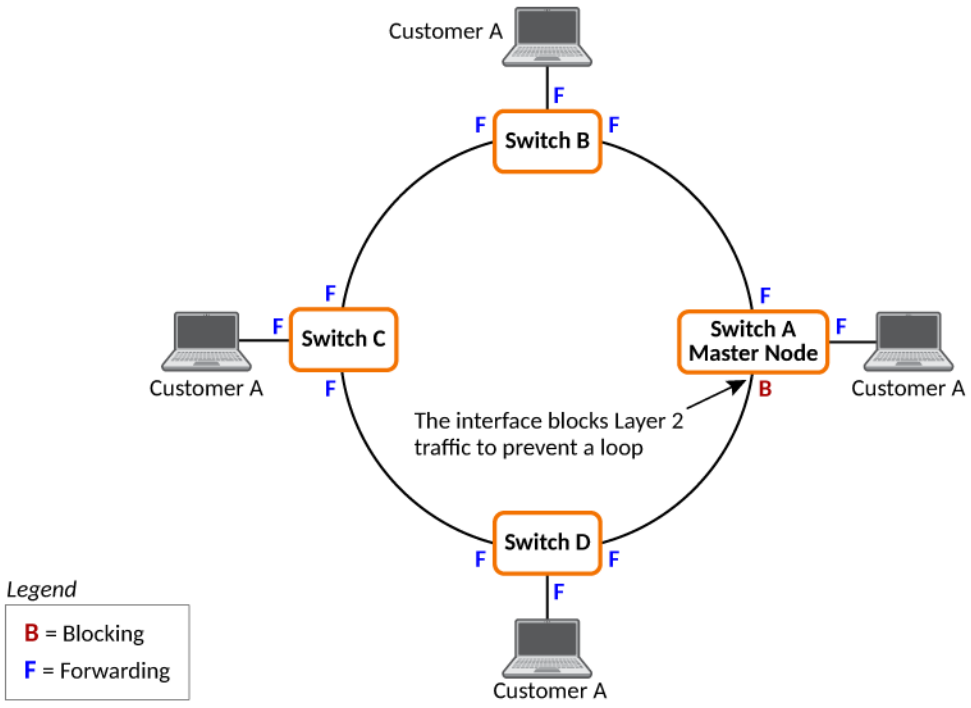
Metro Ring Protocol

Metro Ring Protocol Overview

Metro Ring Protocol (MRP) is a RUCKUS proprietary protocol that prevents Layer 2 loops and provides fast reconvergence in Layer 2 ring topologies. It is an alternative to the Spanning Tree Protocol (STP) and is especially useful in Metropolitan Area Networks (MAN) where using STP has the following drawbacks:

- STP allows a maximum of seven nodes. Metro rings can easily contain more nodes than this.
- STP has a slow reconvergence time, taking many seconds or even minutes. MRP can detect and heal a break in the ring in sub-second time.

FIGURE 1 Metro Ring - Normal State



The ring in Figure 1 shows an example of the MPR metro ring, which consists of four MRP nodes (RUCKUS switches). Each node has two interfaces with the ring. Each node also is connected to a separate customer network. The nodes forward Layer 2 traffic to and from the customer networks

through their interfaces with the ring. The ring interfaces are all in a port-based VLAN. Each customer interface can be in the same VLAN as the ring or in a separate VLAN.

One node is configured as the master node of the ring. One of the two interfaces on the master node is configured as the primary interface; the other is the secondary interface. The primary interface originates Ring Health Packets (RHPs), which are used to monitor the health of the ring. An RHP is forwarded on the ring to the next interface until it reaches the secondary interface of the master node. The secondary interface blocks the packet to prevent a Layer 2 loops.

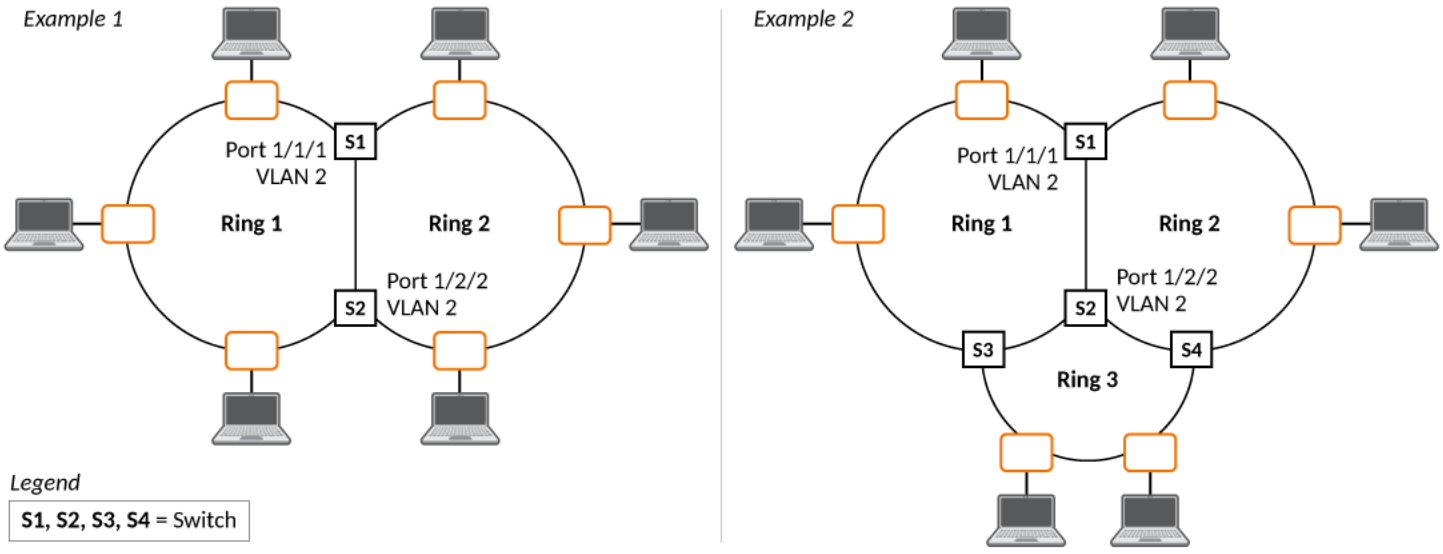
Metro Ring Protocol Configuration Notes

- When you configure Metro Ring Protocol (MRP), RUCKUS recommends that you disable one of the ring interfaces before beginning the ring configuration. Disabling an interface prevents a Layer 2 loop from occurring while you are configuring MRP on the ring nodes. Once the MRP is configured and enabled on all the nodes, you can re-enable the interface.
- The above configurations can be configured as MRP masters or MRP members (for different rings).
- If you configure MRP on a device running Layer 3 software, then restart the device running Layer 2 software, the MRP configuration gets deleted.
- PVST+ packets received on MRP ring interfaces will be forwarded and will cause network disruption. Enabling STP Protection will prevent the MRP ring from flooding the PVST+ packets inside the ring.

MRP Rings with Shared Interfaces (MRP Phase 2)

With MRP Phase 2, MRP rings can be configured to share the same interfaces as long as the interfaces belong to the same VLAN. Figure 2 shows examples of multiple MRP rings that share the same interface.

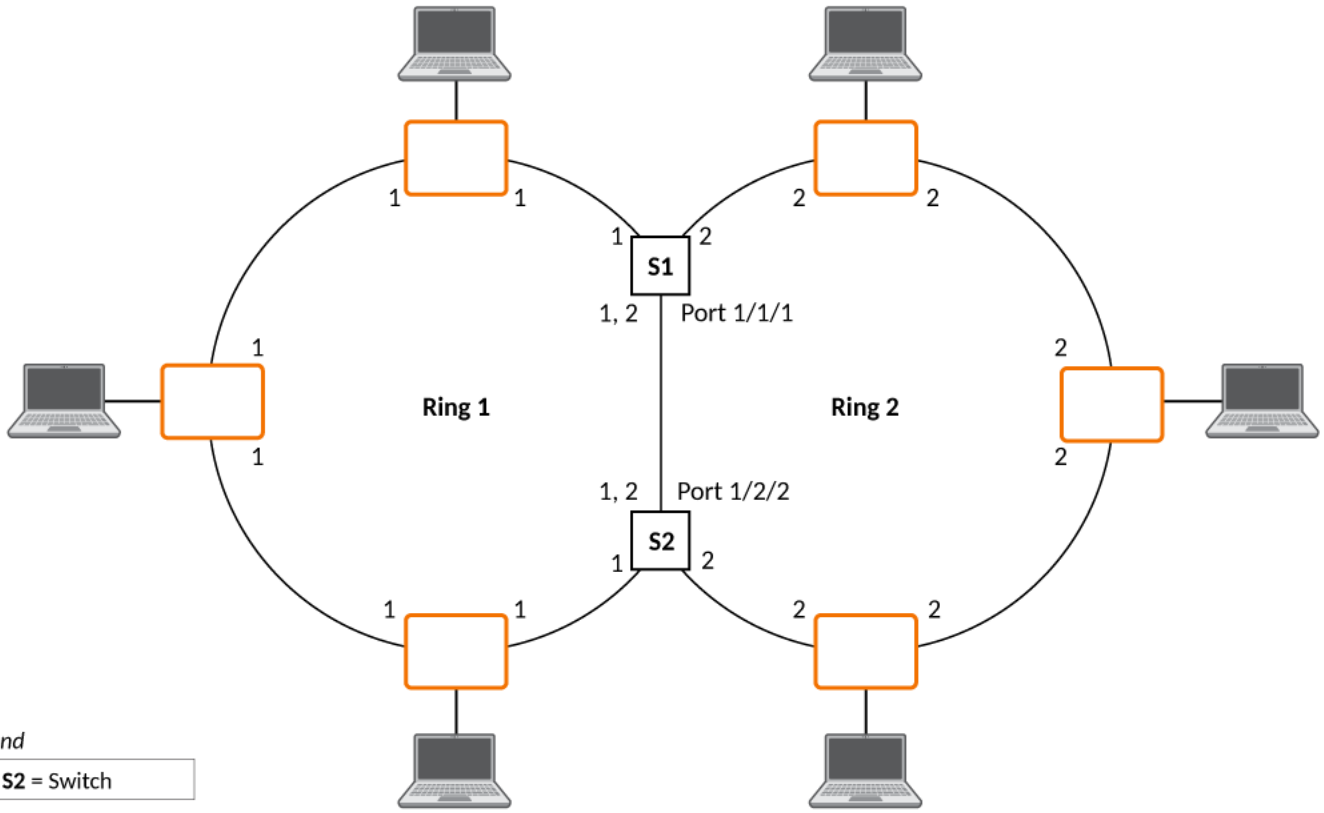
FIGURE 2 Examples of Multiple Rings Sharing the Same Interface - MRP Phase 2



On each node that will participate in the ring, you specify the ring ID and the interfaces that will be used for ring traffic. In a multiple ring configuration, a ring ID determines its priority. The lower the ring ID, the higher priority of a ring.

A ring ID is also used to identify the interfaces that belong to a ring.

FIGURE 3 Interface IDs and Types



For example, in Figure 3, the ID of all interfaces on all nodes on Ring 1 is 1 and all interfaces on all nodes on Ring 2 is 2. Port 1/1/1 on node S1 and Port 1/2/2 on S2 have the IDs of 1 and 2 since the interfaces are shared by Rings 1 and 2.

The ring ID is also used to determine an interface priority. Generally, a ring ID is also the ring priority and the priority of all interfaces on that ring. However, if the interface is shared by two or more rings, then the highest priority (lowest ID) becomes the priority of the interface. For example, in Figure 3, all interfaces on Ring 1, except for Port 1/1/1 on node S1 and Port 1/2/2 on node S2 have a priority of 1. Likewise, all interfaces on Ring 2, except for Port 1/1/1 on node S1 and Port 1/2/2 on node S2 have a priority of 2. Port 1/1/1 on S1 and Port 1/2/2 on S2 have a priority of 1 since 1 is the highest priority (lowest ID) of the rings that share the interface.

If a node has interfaces that have different IDs, the interfaces that belong to the ring with the highest priority become regular ports. Those interfaces that do not belong to the ring with the highest priority become tunnel ports. In Figure 3, nodes S1 and S2 have interfaces that belong to Rings 1 and 2. Those interfaces with a priority of 1 are regular ports. The interfaces with a priority of 2 are the tunnel ports since they belong to Ring 2, which has a lower priority than Ring 1.

Selection of Master Node

Allowing MRP rings to share interfaces limits the nodes that can be designated as the master node. Any node on an MRP ring that does not have a shared interface can be designated as the master node. However, if all nodes on the ring have shared interfaces, nodes that do not have tunnel ports can be designated as the master node of that ring. If none of the nodes meet these criteria, you must change the rings' priorities by reconfiguring the rings' ID.

NOTE
Any node on an MRP ring that has two shared interfaces cannot be elected as the master node.

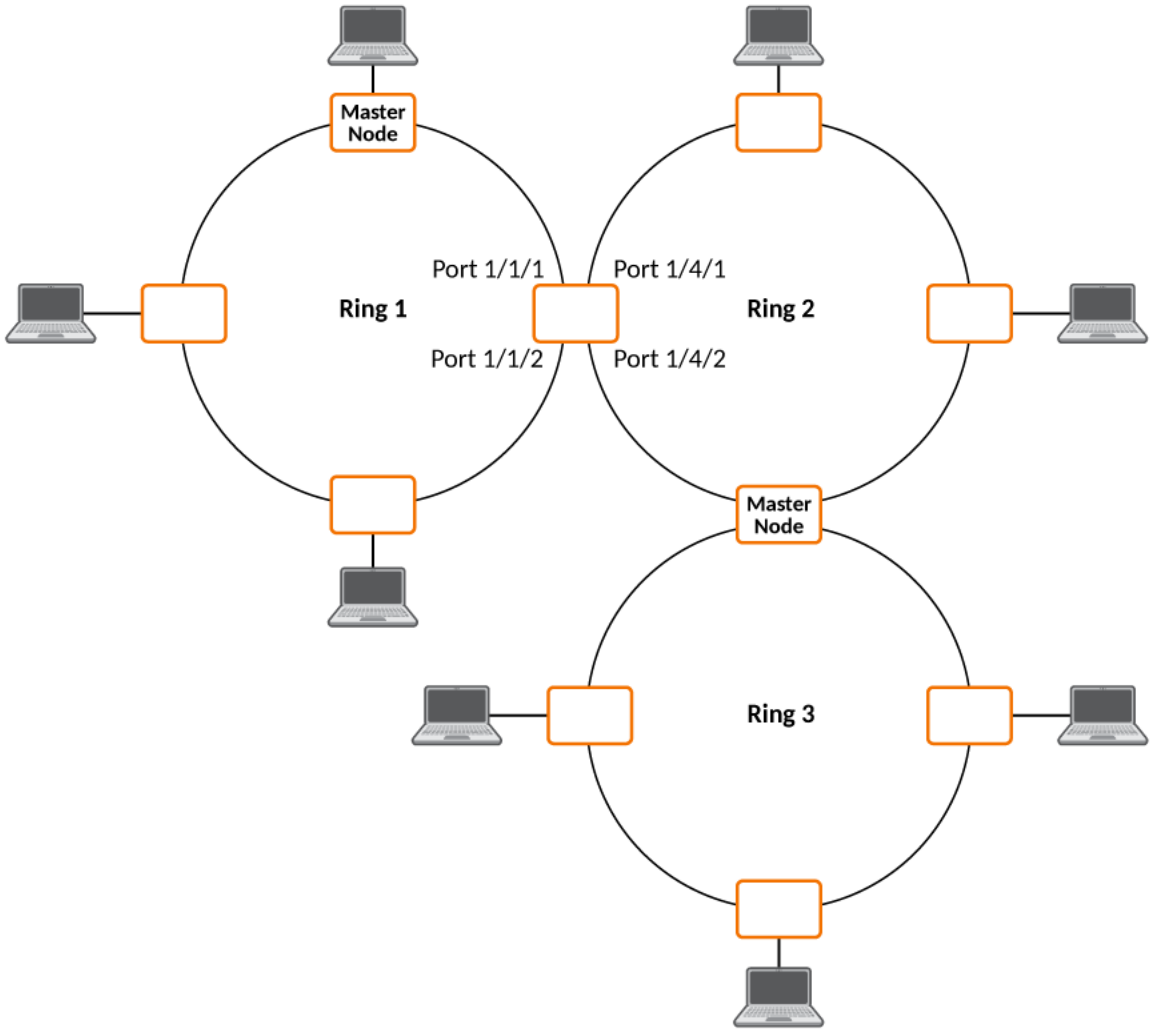
In Figure 3 on page 23, any of the nodes on Ring 1, even S1 or S2, can be a master node since none of its interfaces are tunnel ports. However, in Ring 2, neither S1 nor S2 can be a master node since these nodes contain tunnel ports.

MRP Rings Without Shared Interfaces (MRP Phase 1)

MRP Phase 1 allows you to configure multiple MRP rings, as shown in Figure 4, but the rings cannot share the same link. For example, you cannot configure ring 1 and ring 2 to each have interfaces 1/1/1 and 1/1/2.

Also, when you configure an MRP ring, any node on the ring can be designated as the master node for the ring. A master node can be the master node of more than one ring. (Refer to Figure 4.) Each ring is an independent ring and RHP packets are processed within each ring.

FIGURE 4 Metro Ring - Multiple Rings

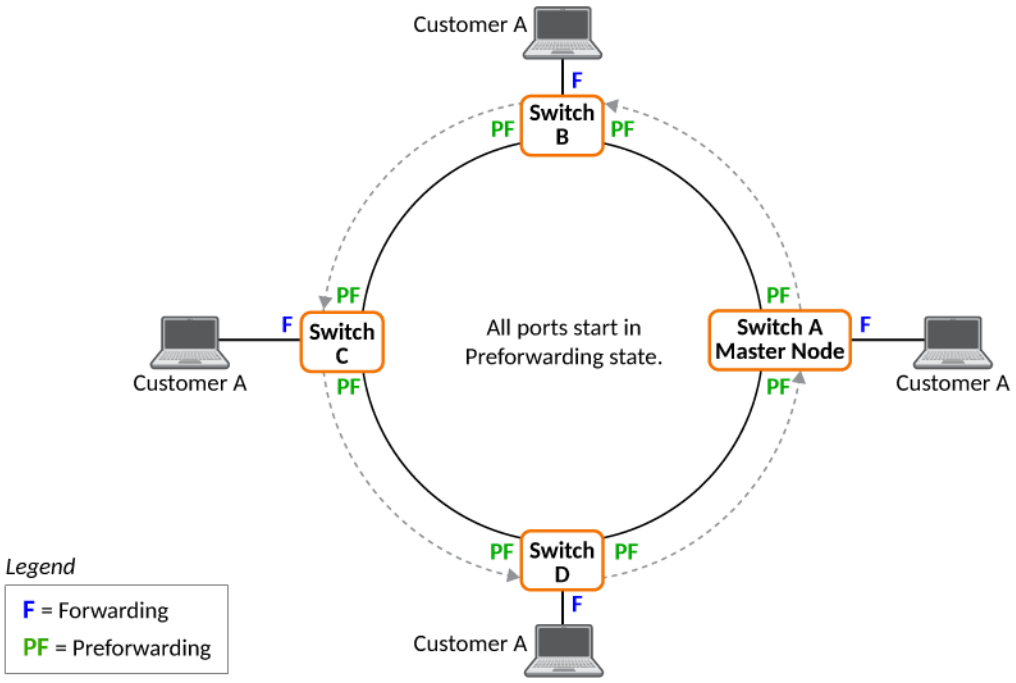


In this example Figure 4, two nodes are each configured with two MRP rings. Any node in a ring can be the master for its ring. A node also can be the master for more than one ring.

Ring Initialization

The ring shown in Figure 1 on page 21 shows the port states in a fully initialized ring without any broken links. Figure 5 shows the initial state of the ring, when MRP is first enabled on the ring switches. All ring interfaces on the master node and member nodes begin in the Preforwarding state (PF).

FIGURE 5 Metro Ring - Initial State



MRP uses Ring Health Packets (RHPs) to monitor the health of the ring. An RHP is an MRP protocol packet. The source address is the MAC address of the master node and the destination MAC address is a protocol address for MRP. The master node generates RHPs and sends them on the ring. The state of a ring port depends on the RHPs.

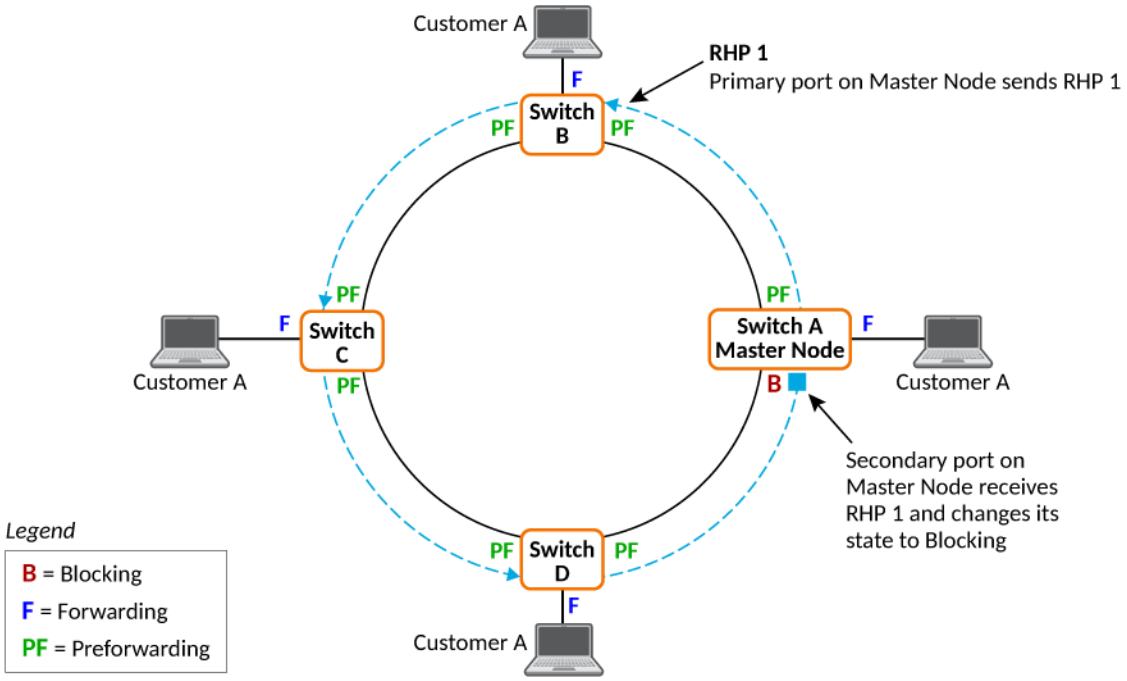
RHP Processing in MRP Phase 1

A ring interface can have one of the following MRP states:

- Preforwarding (PF) - The interface can forward Ring Health Packets (RHPs) but cannot forward data. All ring ports begin in this state when you enable MRP.
- Forwarding (F) - The interface can forward data as well as RHPs. An interface changes from Preforwarding to Forwarding when the port preforwarding time expires. This occurs if the port does not receive an RHP from the master node, or if the forwarding bit in the RHPs received by the port is off. This indicates a break in the ring. The port heals the ring by changing its state to Forwarding. The preforwarding time is the number of milliseconds the port will remain in the Preforwarding state before changing to the Forwarding state, even without receiving an RHP.
- Blocking (B) - The interface cannot forward data. Only the secondary interface on the master node can be Blocking.

Each RHP also has a sequence number. MRP can use the sequence number to determine the round-trip time for RHPs in the ring. Refer to Metro Ring Protocol Diagnostics on page 31.

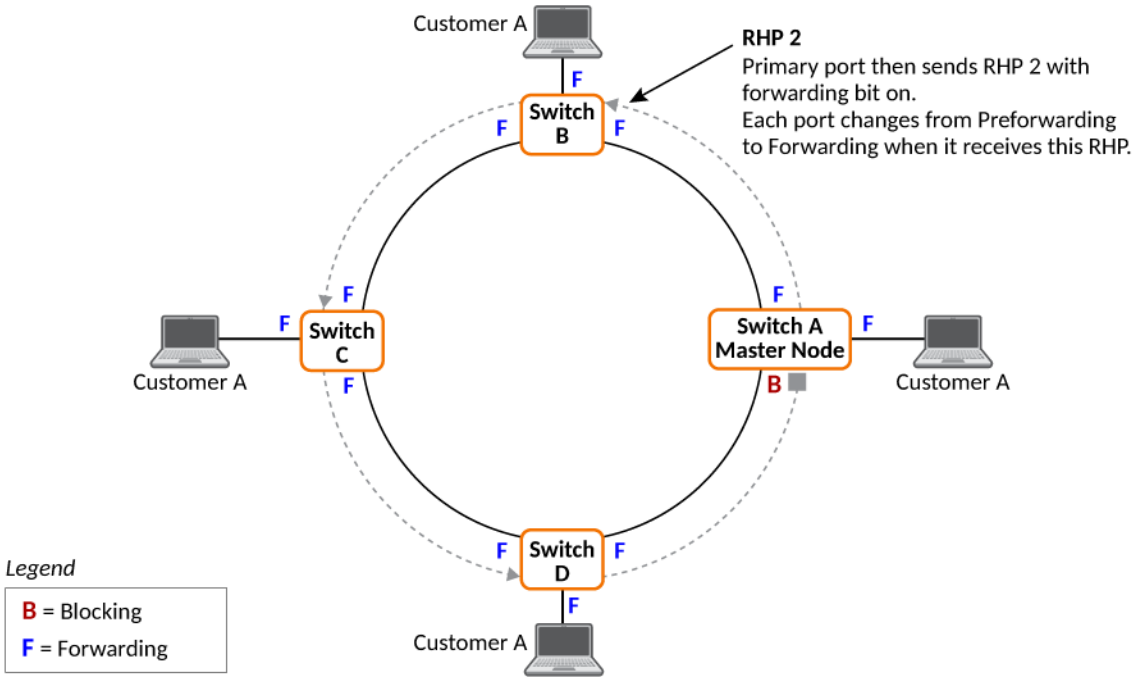
FIGURE 6 Ring Interface RHP1 Processing Example



When MRP is enabled, all ports begin in the Preforwarding state. The primary interface on the master node, although it is in the Preforwarding state like the other ports, immediately sends an RHP onto the ring. The secondary interface on the master node listens for the RHP.

- If the secondary port receives the RHP, all links in the ring are up and the port changes its state to Blocking. The primary port then sends another RHP with its forwarding bit set on. As each of the member ports receives the RHP, the ports changes their state to Forwarding. Typically, this occurs in sub-second time. The ring very quickly enters the fully initialized state.
- If the secondary port does not receive the RHP by the time the preforwarding time expires, a break has occurred in the ring. The port changes its state to Forwarding. The member ports also change their states from Preforwarding to Forwarding as their preforwarding timers expire. The ring is not intact, but data can still travel among the nodes using the links that are up.

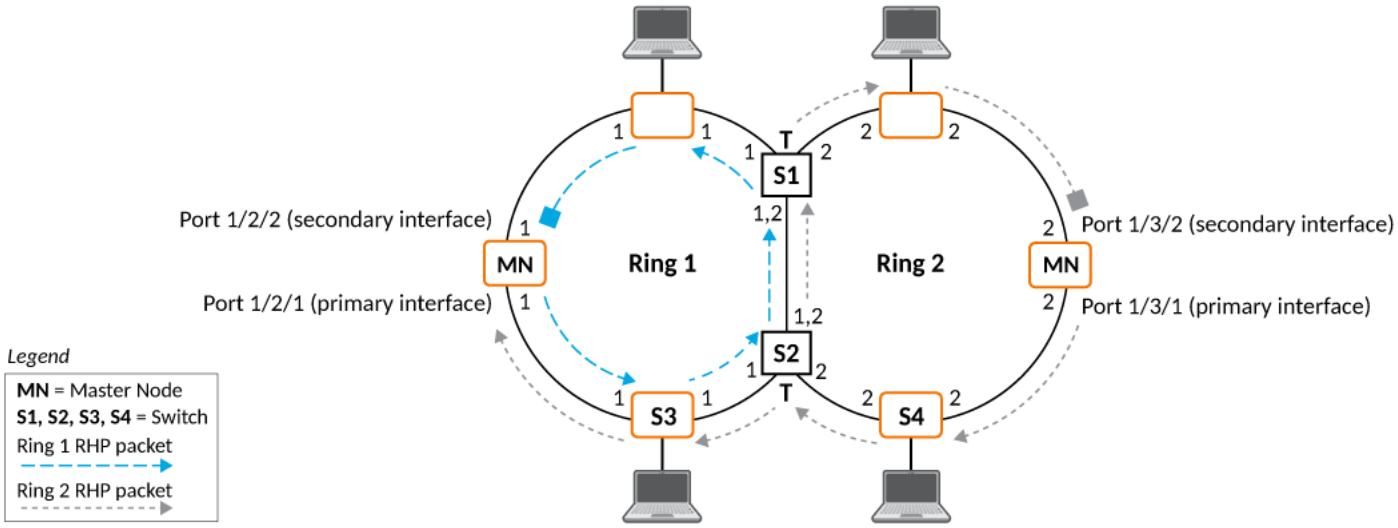
FIGURE 7 Ring Interface RHP2 Processing Example



RHP Processing in MRP Phase 2

Figure 8 shows an example of how RHP packets are processed normally in MRP rings with shared interfaces.

FIGURE 8 Flow of RHP Packets on MRP Rings with Shared Interfaces



Port 1/2/1 on Ring 1 master node is the primary interface of the master node. The primary interface forwards an RHP packet on the ring. Since all the interfaces on Ring 1 are regular ports, the RHP packet is forwarded to all the interfaces until it reaches Port 1/2/2, the secondary interface of the master node. Port 1/2/2 then blocks the packet to complete the process.

On Ring 2, Port 1/3/1, is the primary interface of the master node. It sends an RHP packet on the ring. Since all ports on S4 are regular ports, the RHP packet is forwarded on those interfaces. When the packet reaches S2, the receiving interface is a tunnel port. The port compares the packet priority to its priority. Since the packet priority is the same as the tunnel port priority, the packet is forwarded up the link shared by Rings 1 and 2.

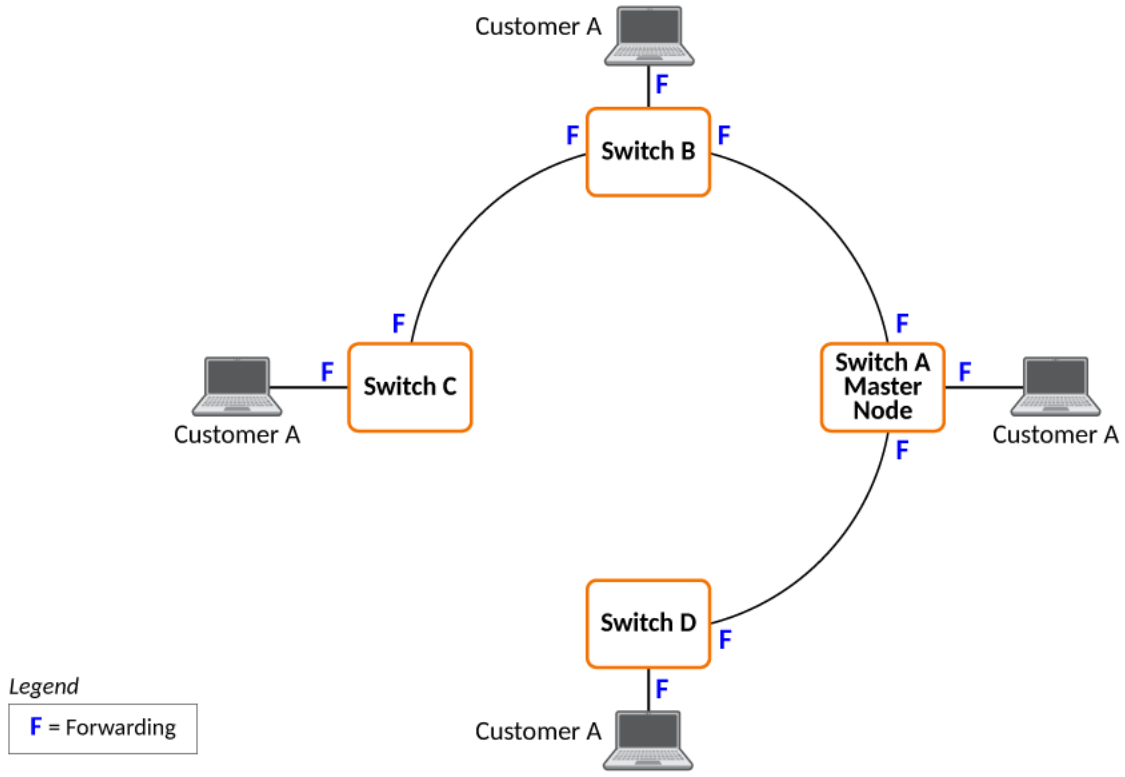
When the RHP packet reaches the interface on node S2 shared by Rings 1 and 2, the packet is forwarded since its priority is less than the interface priority. The packet continues to be forwarded to node S1 until it reaches the tunnel port on S1. That tunnel port determines that the RHP packet priority is equal to the port priority and forwards the packet. The RHP packet is forwarded to the remaining interfaces on Ring 2 until it reaches port 1/3/2, the secondary interface of the master node. Port 1/3/2 then blocks the packet to prevent a loop.

When the RHP packet from Ring 2 reached S2, it was also forwarded from S2 to S3 on Ring 1 since the port on S2 has a higher priority than the RHP packet. The packet is forwarded around Ring 1 until it reaches port 1/2/2, Ring 1 the secondary port. The RHP packet is then blocked by that port.

How Ring Breaks are Detected and Healed

Figure 9 shows ring interface states following a link break. MRP quickly heals the ring and preserves connectivity among the customer networks.

FIGURE 9 Metro Ring - Ring Break



If a break in the ring occurs, MRP heals the ring by changing the states of some of the ring interfaces:

- **Blocking interface** - The Blocking interface on the master node has a dead timer. If the dead time expires before the interface receives one of its ring RHPs, the interface changes state to Preforwarding. Once the secondary interface changes state to Preforwarding:
  - If the interface receives an RHP, the interface changes back to the Blocking state and resets the dead timer.
  - If the interface does not receive an RHP for its ring before the Preforwarding time expires, the interface changes to the Forwarding state, as shown in Figure 9.
- **Forwarding interfaces** - Each member interface remains in the Forwarding state.

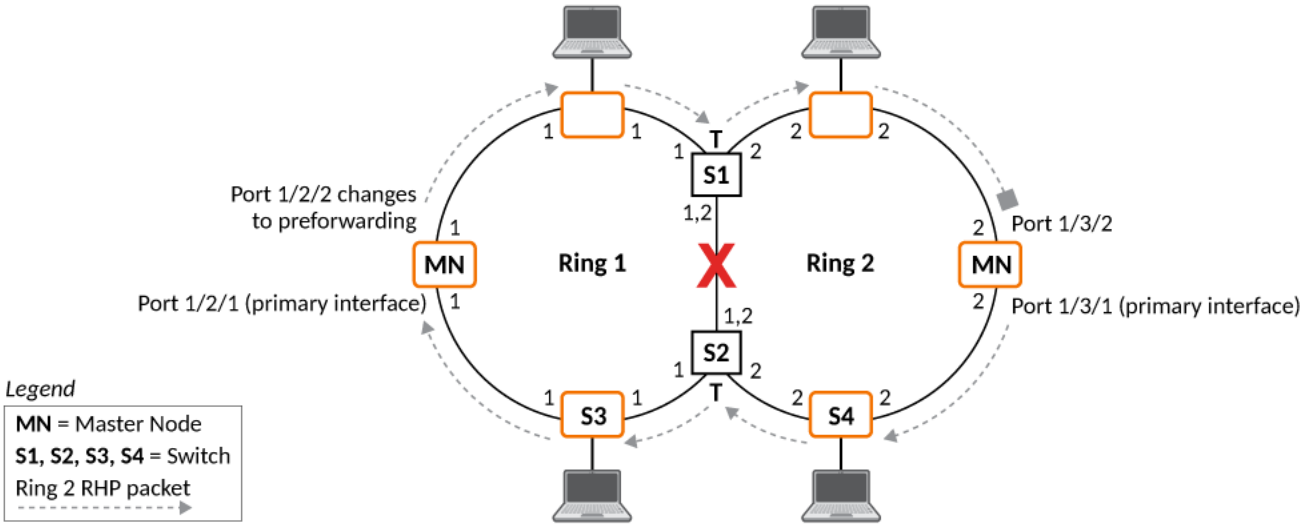
When the broken link is repaired, the link interfaces come up in the Preforwarding state, which allows RHPs to travel through the restored interfaces and reach the secondary interface on the Master node:

- If an RHP reaches the Master node secondary interface, the ring is intact. The secondary interface changes to Blocking. The Master node sets the forwarding bit on in the next RHP. When the restored interfaces receive this RHP, they immediately change state to Forwarding.
- If an RHP does not reach the Master node secondary interface, the ring is still broken. The Master node does not send an RHP with the forwarding bit on. In this case, the restored interfaces remain in the Preforwarding state until the preforwarding timer expires, then change to the Forwarding state.

If the link between shared interfaces breaks (Figure 10), the secondary interface on Ring 1 master node changes to a Preforwarding state. The RHP packet sent by port 1/3/1 on Ring 2 is forwarded through the interfaces on S4, then to S2. The packet is then forwarded through S2 to S3, but not from S2 to S1 since the link between the two nodes is not available. When the packet reaches Ring 1 master node, the packet is forwarded through the secondary interface since it is currently in a Preforwarding state. A secondary interface in Preforwarding state ignores any RHP packet that is not from its ring. The secondary interface changes to blocking mode only when the RHP packet forwarded by its primary interface is returned.

The packet then continues around Ring 1, through the interfaces on S1 to Ring 2 until it reaches Ring 2 master node. Port 1/3/2, the secondary interface on Ring 2 changes to blocking mode since it received its own packet, then blocks the packet to prevent a loop.

FIGURE 10 Flow of RHP Packets When a Link for Shared Interfaces Breaks

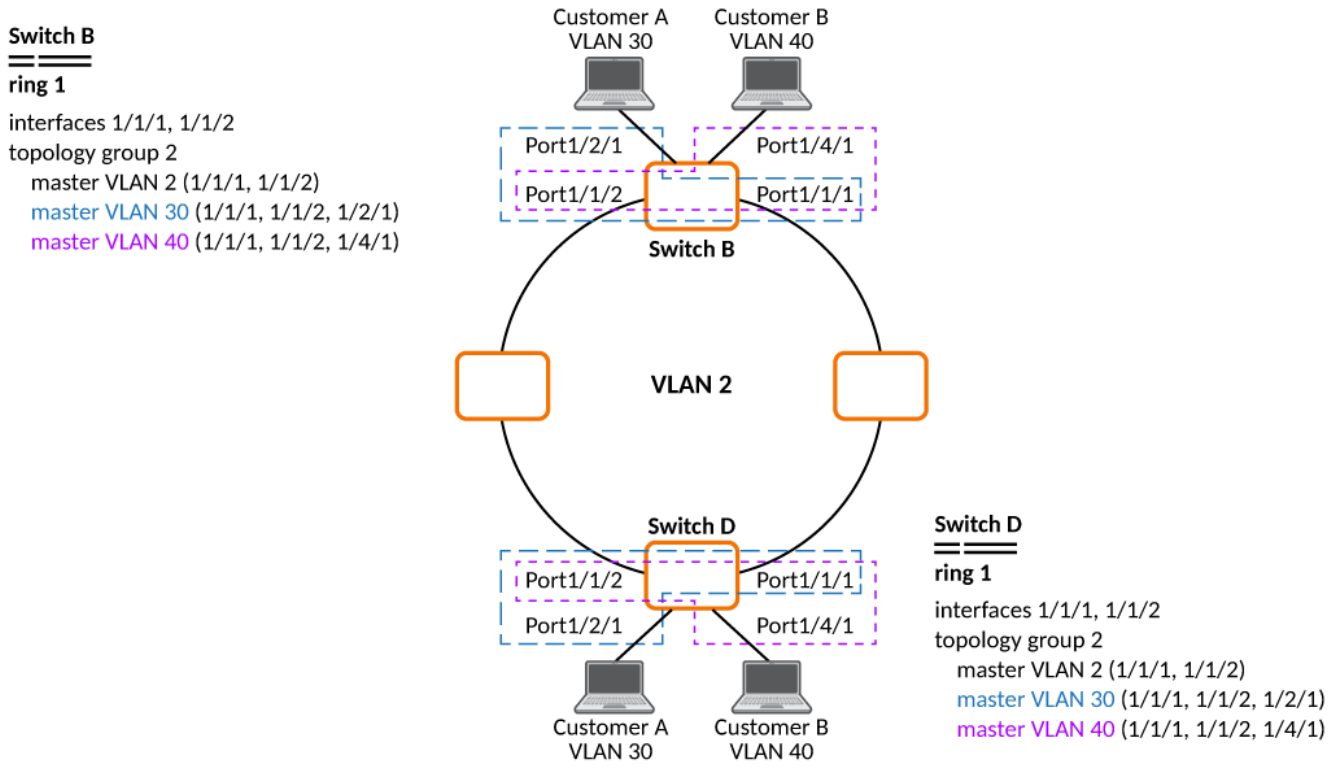


RHP packets follow this flow until the link is restored; then the RHP packet returns to it normal flow as shown in Figure 8 on page 27.

Master VLANs and Customer VLANs

All the ring ports must be in the same VLAN. Placing the ring ports in the same VLAN provides Layer 2 connectivity for a given customer across the ring. The following figure shows an example.

FIGURE 11 Metro Ring - Ring VLAN and Customer VLANs



Notice that each customer has their own VLAN. Customer A has VLAN 30 and Customer B has VLAN 40. Customer A host attached to Switch D can reach the Customer A host attached to Switch B at Layer 2 through the ring. Since Customer A and Customer B are on different VLANs, they will not receive each other traffic.

You can configure MRP separately on each customer VLAN. However, this is impractical if you have many customers. To simplify configuration when you have a lot of customers (and therefore a lot of VLANs), you can use a topology group.

A topology group enables you to control forwarding in multiple VLANs using a single instance of a Layer 2 protocol such as MRP. A topology group contains a master VLAN and member VLANs. The master VLAN contains all the configuration parameters for the Layer 2 protocol (STP, MRP, or VSRP). The member VLANs use the Layer 2 configuration of the master VLAN.

In Figure 11, VLAN 2 is the master VLAN and contains the MRP configuration parameters for ring 1. VLAN 30 and VLAN 40, the customer VLANs, are member VLANs in the topology group. Since a topology group is used, a single instance of MRP provides redundancy and loop prevention for both the customer VLANs.

If you use a topology group:

- The master VLAN must contain the ring interfaces. The ports must be tagged, since they will be shared by multiple VLANs.
- The member VLAN for a customer must contain the two ring interfaces and the interfaces for the customer. Since these interfaces are shared with the master VLAN, they must be tagged. Do not add another customer interfaces to the VLAN.

For more information about topology groups, refer to Topology Groups on page 289.

Refer to MRP CLI Example on page 39 for the configuration commands required to implement the MRP configuration shown in Figure 11.

Metro Ring Protocol Diagnostics

The Metro Ring Protocol (MRP) diagnostics feature calculates how long it takes for RHP packets to travel through the ring. When you enable MRP diagnostics, the software tracks RHP packets according to their sequence numbers and calculates how long it takes an RHP packet to travel one time through the entire ring. When you display the diagnostics, the CLI shows the average round-trip time for the RHP packets sent since you enabled diagnostics. The calculated results have a granularity of 1 microsecond.

Metro Ring Protocol Configuration

To configure Metro Ring Protocol (MRP), perform the following tasks. You need to perform the first task on only one of the nodes. Perform the remaining tasks on all the nodes.

1. Disable one of the ring interfaces. This prevents a Layer 2 loop from occurring while you are configuring the devices for MRP.
2. Add an MRP ring to a port-based VLAN. When you add a ring, the CLI changes to the configuration level for the ring, where you can perform the following tasks.
   - Optionally, specify a name for the ring.
   - Enable the device to be the master for the ring. Each ring can have only one master node.
   - Specify the MRP interfaces. Each device has two interfaces to an MRP ring.

     On the master node, the primary interface is the one that originates RHPs. Ring control traffic and Layer 2 data traffic will flow in the outward direction from this interface by default. On member nodes, the direction of traffic flow depends on the traffic direction selected by the master node. Therefore, on a member node, the order in which you enter the interfaces does not matter.
   - Optionally, change the hello time and the preforwarding time. These parameters control how quickly failover occurs following a change in the state of a link in the ring.
   - Enable the ring.
3. Optionally, add the ring VLAN to a topology group to add more VLANs to the ring. If you use a topology group, make sure you configure MRP on the group master VLAN. Refer to Topology Groups on page 289.
4. Re-enable the interface you disabled to prevent a Layer 2 loop. Once MRP is enabled, MRP will prevent the Layer 2 loop.

NOTE
To take advantage of every interface in a Metro network, you can configure another MRP ring and either configure a different Master node for the ring or reverse the configuration of the primary and secondary interfaces on the Master node. Configuring multiple rings enables you to use all the ports in the ring. The same port can forward traffic one ring while blocking traffic for another ring.

Configuring Metro Ring Protocol

To configure MRP ring between two devices device 1 (mrp-master) and device 2 (mrp-member), complete the following steps.

1. Enter the global configuration mode.

```
DUT1# configure terminal
```

2. Configure VLAN on the device.

```
DUT1(config)#vlan 2
```

3. Add ports to the VLAN.

```
DUT1(config-vlan-2)#tagged ethernet 2/2/2 ethernet 2/2/3
```

This step adds tagged port(s) ethernet 2/2/2 to 2/2/3 to port-vlan 2.

Ideally, a VLAN that runs MRP should have only two uplink/p2p links because MRP configuration takes only two interfaces.

4. Define MRP ring ID to enable MRP.

```
DUT1(config-vlan-2)#metro-ring 2
```

NOTE
The ring ID ranges from 1 through 1023. 256 is reserved for VSRP.

The prompt changes to MRP mode to define MRP parameters.

5. In the MRP mode, configure other MRP parameters such as the following:

a) Define a name for the MRP ring.

```
DUT1(config-vlan-2-mrp-2)#name mrp-node1
```

b) Configure MRP ring interfaces.

```
DUT1(config-vlan-2-mrp-2)#ring-interfaces ethernet 2/2/2 ethernet 2/2/3
```

c) Define the device as MRP ring Master.

```
DUT1(config-vlan-2-mrp-2)#master
```

There can be only one MRP Master node in the MRP Ring. Master node will have a Blocking port after convergence.

d) (Optional) Enable MRP ring diagnostics.

```
DUT1(config-vlan-2-mrp-2)#diagnostics
```

MRP ring diagnostics can only be enabled on a Master node, otherwise the following error is displayed - *"Error - Metro Ring diagnostics can only be enabled on ring Master"*.

e) (Optional) Change MRP ring default timers.

- Change the preforwarding timer.

```
DUT1(config-vlan-2-mrp-2)#preforwarding-time 600
```

The preforwarding time must be at least twice the value of the hello-time and must be a multiple of the hello-time. If UDLD is also enabled on the device, Ruckus recommends that you set the MRP preforwarding time slightly higher than the default of 300 ms; for example, to 400 or 500 ms.

- Change the hello-time in accordance with the pre-forwarding time.

```
DUT1(config-vlan-2-mrp-2)#hello-time 300
```

6. Enable MRP ring.

```
DUT1(config-vlan-2-mrp-2)#enable
```

7. Similarly configure MRP on other devices in the same ring which will be acting as ring member(s).

```
DUT2(config)#vlan 2
DUT2(config-vlan-2)#tagged ethernet 1/2/1 ethernet 1/2/2
Added tagged port(s) ethe 1/2/1 to 1/2/2 to port-vlan 2.
DUT2(config-vlan-2)#metro-ring 2
DUT2(config-vlan-2-mrp-2)#ring-interfaces ethernet 1/2/1 ethernet 1/2/2
DUT2(config-vlan-2-mrp-2)#enable
```

8. Verify MRP configuration on device 1.

```
DUT1(config)#show running vlan 2
vlan 2 by port
 tagged ethe 2/2/2 to 2/2/3
 metro-ring 2
  master
  ring-interfaces ethernet 2/2/2 ethernet 2/2/3
  enable
  preforwarding-time 600
  hello-time 300
  name mrp-node1
  diagnostics
!
!
DUT1(config)#
```

Displaying MRP Information

The following output displays the MRP details on device 1.

```
DUT1(config-vlan-2)# show metro-ring
Total MRP entries configured = 1

Metro Ring 2 - mrp-node1
=============
Ring       State      Ring       Master     Topo       Hello      Prefwing
id                    role       vlan       group      time(ms)   time(ms)
2          enabled    master     2          not conf   300        600

Ring interfaces Interface role Forwarding state Active interface interface type
ethernet 2/2/2   primary        forwarding       ethernet 2/2/2   regular
ethernet 2/2/3   secondary      blocking         ethernet 2/2/3   regular

RHPs sent            RHPs rcvd            TC RBPDUs rcvd       State changes
1535                 2310                 0                    5
```

The following output displays the MRP details based on the specified ring ID.

```
DUT1(config-vlan-2)# show metro-ring 2

Metro Ring 2 - mrp-node1
=============
Ring       State      Ring       Master     Topo       Hello      Prefwing
id                    role       vlan       group      time(ms)   time(ms)
2          enabled    master     2          not conf   300        600    <<<<<<<<<< master

Ring interfaces Interface role Forwarding state Active interface interface type
ethernet 2/2/2   primary        forwarding       ethernet 2/2/2   regular
ethernet 2/2/3   secondary      blocking         ethernet 2/2/3   regular

RHPs sent            RHPs rcvd            TC RBPDUs rcvd       State changes
1548                 2323                 0                    5
```

The following output displays the MRP diagnostics result on the master node.

```
DUT1(config-vlan-2)# show metro-ring 2 diagnostics
```

```
Metro Ring 2 - mrp-node1
=============
diagnostics results

Ring       Diag       RHP average     Recommended     Recommended
id         state      time(microsec)  hello time(ms)  Prefwing time(ms)
2          enabled    < 1628          100             300

Diag frame sent      Diag frame lost
1555                 1
DUT1(config-vlan-2)#
```

The following output displays the MRP details on device 2:

```
DUT2(config-vlan-2)# show metro-ring
Total MRP entries configured = 1

Metro Ring 2
=============
Ring       State      Ring       Master     Topo       Hello      Prefwing
id                    role       vlan       group      time(ms)   time(ms)
2          enabled    member     2          not conf   100        300                          <<<<<<<<<<<
member

Ring interfaces Interface role Forwarding state Active interface interface type
ethernet 1/2/2   primary        forwarding       ethernet 1/2/2   regular
ethernet 1/2/1   secondary      forwarding       ethernet 1/2/1   regular

RHPs sent            RHPs rcvd            TC RBPDUs rcvd       State changes
0                    31                   0                    4
```

The following output displays the MRP details based on the specified ring ID on device 2.

```
DUT2(config-vlan-2)# show metro-ring 2

Metro Ring 2
=============
Ring       State      Ring       Master     Topo       Hello      Prefwing
id                    role       vlan       group      time(ms)   time(ms)
2          enabled    member     2          not conf   100        300

Ring interfaces Interface role Forwarding state Active interface interface type
ethernet 1/2/2   primary        forwarding       ethernet 1/2/2   regular
ethernet 1/2/1   secondary      forwarding       ethernet 1/2/1   regular

RHPs sent            RHPs rcvd            TC RBPDUs rcvd       State changes
0                    40                   0                    4
```

Scenario - 2: Configuring MRP

To configure MRP ring on device 1 (mrp-master) and device 2 (mrp-member) and MRP ring3 in the same VLAN between device 2 and device 3 (mrp-master), complete the following steps.

1. Complete the MRP configuration (ring 2) between device 1 and device 2 as specified in Configuring Metro Ring Protocol on page 31.
2. Add uplinks/p2p ports between device 2 and device 3 to the same VLAN (VLAN 2).

   ```
   device(config-vlan-2)# tagged ethernet 1/2/3 ethernet 1/2/4
   Added tagged port(s) ethe 1/2/3 to 1/2/4 to port-vlan 2.
   ```

3. Configure MRP ring ID.

   ```
   device(config-vlan-2)# metro-ring 3
   ```

   The prompt changes to MRP mode to define MRP parameters.

4. Configure MRP ring interfaces.

```
device(config-vlan-2-mrp-3)# ring-interfaces ethernet 1/2/3 ethernet 1/2/4
```

5. Enable MRP ring.

```
device(config-vlan-2-mrp-3)# enable
```

6. Verify MRP configuration on device 2.

```
device(config-vlan-2)# show run vlan 2
vlan 2 by port
 tagged ethe 1/2/1 to 1/2/4
 metro-ring 2
  ring-interfaces ethernet 1/2/2 ethernet 1/2/1
  enable
 metro-ring 3
  ring-interfaces ethernet 1/2/4 ethernet 1/2/3
  enable
!
!
device(config-vlan-2)#
```

7. Configure VLAN on device 3.

```
device(config)# vlan 2
```

8. Add ports to the VLAN.

```
device(config-vlan-2)# tagged ethernet 1/2/3 ethernet 1/2/4
Added tagged port(s) ethe 1/2/3 to 1/2/4 to port-vlan 2.
```

9. Configure MRP ring ID.

```
device(config-vlan-2)# metro-ring 3
```

The prompt changes to MRP mode to define MRP parameters.

10. Configure MRP ring interfaces.

```
device(config-vlan-2-mrp-3)# ring-interfaces e 1/2/3 e 1/2/4
```

11. Define the device as MRP ring Master.

```
device(config-vlan-2-mrp-3)# master
```

12. Enable MRP ring.

```
device(config-vlan-2-mrp-3)# enable
```

13. Enable MRP ring diagnostics.

```
device(config-vlan-2-mrp-2)# diagnostics
```

MRP ring diagnostics can only be enabled on a Master node, otherwise the following error is displayed "Error - Metro Ring diagnostics can only be enabled on ring Master."

14. Verify MRP configuration on device 3.

```
device(config-vlan-2)# show run vlan 2
vlan 2 by port
 tagged ethe 1/2/3 to 1/2/4
 metro-ring 3
  master
  ring-interfaces ethernet 1/2/3 ethernet 1/2/4
  enable
  diagnostics
!
!
device(config-vlan-2)#
```

Displaying MRP Information (Scenario 2 Configuration)

The following output displays the MRP details on device 2.

```
DUT2(config-vlan-2)# show metro-ring
Total MRP entries configured = 2

Metro Ring 2
=============
Ring       State      Ring       Master     Topo       Hello      Prefwing
id                    role       vlan       group      time(ms)   time(ms)
2          enabled    member     2          not conf   100        300

Ring interfaces Interface role Forwarding state Active interface interface type
ethernet 1/2/2   primary        forwarding       ethernet 1/2/2   regular
ethernet 1/2/1   secondary      forwarding       ethernet 1/2/1   regular

RHPs sent            RHPs rcvd            TC RBPDUs rcvd       State changes
0                    627                  0                    4


Metro Ring 3
=============
Ring       State      Ring       Master     Topo       Hello      Prefwing
id                    role       vlan       group      time(ms)   time(ms)
3          enabled    member     2          not conf   100        300

Ring interfaces Interface role Forwarding state Active interface interface type
ethernet 1/2/4   primary        forwarding       ethernet 1/2/4   regular
ethernet 1/2/3   secondary      forwarding       ethernet 1/2/3   regular

RHPs sent            RHPs rcvd            TC RBPDUs rcvd       State changes
0                    62                   0                    4
```

The following output displays the MRP details based on the specified ring ID (ring 3).

```
DUT2(config-vlan-2)# show metro-ring 3

Metro Ring 3
=============
Ring       State      Ring       Master     Topo       Hello      Prefwing
id                    role       vlan       group      time(ms)   time(ms)
3          enabled    member     2          not conf   100        300

Ring interfaces Interface role Forwarding state Active interface interface type
ethernet 1/2/4   primary        forwarding       ethernet 1/2/4   regular
ethernet 1/2/3   secondary      forwarding       ethernet 1/2/3   regular

RHPs sent            RHPs rcvd            TC RBPDUs rcvd       State changes
0                    116                  0                    4
```

The following output displays the MRP details based on the specified ring ID (ring 2).

```
DUT2(config-vlan-2)# show metro-ring 2

Metro Ring 2
=============
Ring       State      Ring       Master     Topo       Hello      Prefwing
id                    role       vlan       group      time(ms)   time(ms)
2          enabled    member     2          not conf   100        300

Ring interfaces Interface role Forwarding state Active interface interface type
ethernet 1/2/2   primary        forwarding       ethernet 1/2/2   regular
ethernet 1/2/1   secondary      forwarding       ethernet 1/2/1   regular

RHPs sent            RHPs rcvd            TC RBPDUs rcvd       State changes
0                    657                  0                    4
```

The following output displays the MRP details on device 2:

```
DUT3(config-vlan-2)# show metro-ring
Total MRP entries configured = 1

Metro Ring 3
=============
Ring       State      Ring       Master     Topo       Hello      Prefwing
id                    role       vlan       group      time(ms)   time(ms)
3          enabled    master     2          not conf   100        300

Ring interfaces Interface role Forwarding state Active interface interface type
ethernet 1/2/3   primary        forwarding       ethernet 1/2/3   regular
ethernet 1/2/4   secondary      blocking         ethernet 1/2/4   regular

RHPs sent            RHPs rcvd            TC RBPDUs rcvd       State changes
76                   73                   0                    2
```

The following output displays the MRP details based on the specified ring ID (ring 3).

```
DUT3(config-vlan-2)# show metro-ring 3

Metro Ring 3
=============
Ring       State      Ring       Master     Topo       Hello      Prefwing
id                    role       vlan       group      time(ms)   time(ms)
3          enabled    master     2          not conf   100        300

Ring interfaces Interface role Forwarding state Active interface interface type
ethernet 1/2/3   primary        forwarding       ethernet 1/2/3   regular
ethernet 1/2/4   secondary      blocking         ethernet 1/2/4   regular

RHPs sent            RHPs rcvd            TC RBPDUs rcvd       State changes
106                  103                  0                    2
```

The following output displays the MRP diagnostics result on the master node.

```
DUT3(config-vlan-2)# show metro-ring 3 diagnostics

Metro Ring 3
=============
diagnostics results

Ring       Diag       RHP average     Recommended     Recommended
id         state      time(microsec)  hello time(ms)  Prefwing time(ms)
3          enabled    < 1157          100             300

Diag frame sent      Diag frame lost
139                  0
```

Scenario - 3: Configuring MRP for Topology Group

To configure MRP ring between device 1 (mrp-master) and device 2 (mrp-member) for topology group, complete the following steps .

1. Complete the MRP configuration (ring 2) between device 1 and device 2 as specified in Configuring Metro Ring Protocol on page 31.
2. Configure multiple VLANs as per the requirement and add the same ring interfaces to the VLAN.

```
DUT1(config)# vlan 11 to 20
DUT1(config-mvlan-11-20)# tagged ethernet 2/2/2 ethernet 2/2/3
```

3. Configure a topology-group with MRP-configured VLAN as master VLAN and member VLAN as other VLANs where you want to run MRP.

```
DUT1(config)# topology-group 1
DUT1(config-topo-group-1)# master-vlan 2
DUT1(config-topo-group-1)# member-vlan 11 to 20
```

4. Perform the similar configurations on device 2.

```
DUT2(config)# vlan 2
DUT2(config-vlan-2)# tagged ethernet 1/2/1 ethernet 1/2/2
Added tagged port(s) ethe 1/2/1 to 1/2/2 to port-vlan 2.
DUT2(config-vlan-2)# metro-ring 2
DUT1(config-vlan-2-mrp-2)# name mrp-node2
DUT2(config-vlan-2-mrp-2)# ring-interfaces ethernet 1/2/1 ethernet 1/2/2
DUT2(config-vlan-2-mrp-2)# enable
DUT2(config-vlan-2-mrp-2)# exit
DUT2(config-vlan-2)# vlan 11 to 20
DUT2(config-mvlan-11-20)# tagged ethernet 1/2/1 ethernet 1/2/2
DUT2(config-mvlan-11-20)# exit
DUT2(config)#topology-group 1
DUT2(config-topo-group-1)# master-vlan 2
DUT2(config-topo-group-1)# member-vlan 11 to 20
```

Displaying MRP information for Topology Group

The following output displays the MRP details on device 1.

```
DUT1(config)# show metro-ring
Total MRP entries configured = 1

Metro Ring 2 - mrp-node1
=============
Ring       State      Ring       Master     Topo       Hello      Prefwing
id                    role       vlan       group      time(ms)   time(ms)
2          enabled    master     2          1          100        300

Ring interfaces Interface role Forwarding state Active interface interface type
ethernet 2/2/2   primary        forwarding       ethernet 2/2/2   regular
ethernet 2/2/3   secondary      blocking         ethernet 2/2/3   regular

RHPs sent            RHPs rcvd            TC RBPDUs rcvd       State changes
5488                 6205                 0                    18
```

The following output displays the topology group information.

```
DUT1(config)# show topology-group
Topology Group 1
=================
 master-vlan 2
 member-vlan 11 to 20

 Number of Member VLANs         10
 Common control ports           L2 protocol
 ethe 2/2/2                     MRP
```

```
 ethe 2/2/3                    MRP
 Per vlan free ports
```

The following output displays the MRP details based on device 2.

```
DUT2(config)# show metro-ring
Total MRP entries configured = 1

Metro Ring 2 - mrp-node2
=============
Ring       State      Ring       Master     Topo       Hello      Prefwing
id                    role       vlan       group      time(ms)   time(ms)
2          enabled    member     2          1          100        300

Ring interfaces Interface role Forwarding state Active interface interface type
ethernet 1/2/2   primary        forwarding       ethernet 1/2/2   regular
ethernet 1/2/1   secondary      forwarding       ethernet 1/2/1   regular

RHPs sent            RHPs rcvd            TC RBPDUs rcvd       State changes
0                    3499                 0                    4
```

The following output displays the topology group information on device 2:

```
DUT2(config)# show topology-group 1
Topology Group 1
================
 master-vlan 2
 member-vlan 11 to 20

 Number of Member VLANs         10
 Common control ports           L2 protocol
 ethernet 1/2/1                 MRP
 ethernet 1/2/2                 MRP
 Per vlan free ports
```

MRP CLI Example

The following examples show the CLI commands required to implement the MRP configuration shown in Figure 11 on page 30.

NOTE
For simplicity, the figure shows the VLANs on only two switches. The CLI examples implement the ring on all four switches.

MRP Commands on Switch A (Master Node)

The following commands configure a VLAN for the ring. The ring VLAN must contain both of the node interfaces with the ring. Add these interfaces as tagged interfaces, since the interfaces also must be in each of the customer VLANs configured on the node.

```
device(config)# vlan 2
device(config-vlan-2)# tag ethernet 1/1/1 to 1/1/2
device(config-vlan-2)# metro-ring 1
device(config-vlan-2-mrp-1)# name "Metro A"
device(config-vlan-2-mrp-1)# master
device(config-vlan-2-mrp-1)# ring-interface ethernet 1/1/1 ethernet 1/1/2
device(config-vlan-2-mrp-1)# enable
device(config-vlan-2-mrp-1)# exit
device(config-vlan-2)# exit
```

The following commands configure the customer VLANs. The customer VLANs must contain both the ring interfaces as well as the customer interfaces.

```
device(config)# vlan 30
device(config-vlan-30)# tag ethernet 1/1/1 to 1/1/2
device(config-vlan-30)# tag ethernet 1/2/1
```

```
device(config-vlan-30)# exit
device(config)# vlan 40
device(config-vlan-40)# tag ethernet 1/1/1 to 1/1/2
device(config-vlan-40)# tag ethernet 1/4/1
device(config-vlan-40)# exit
```

The following commands configure topology group 1 on VLAN 2. The master VLAN is the one that contains the MRP configuration. The member VLANs use the MRP parameters of the master VLAN. The control interfaces (the ones shared by the master VLAN and member VLAN) also share MRP state.

```
device(config)# topology-group 1
device(config-topo-group-1)# master-vlan 2
device(config-topo-group-1)# member-vlan 30
device(config-topo-group-1)# member-vlan 40
```

MRP Commands on Switch B

The commands for configuring Switches B, C, and D are similar to the commands for configuring Switch A, with two differences: the nodes are not configured to be the ring master. Omitting the **master** command is required for non-master nodes.

```
device(config)# vlan 2
device(config-vlan-2)# tag ethernet 1/1/1 to 1/1/2
device(config-vlan-2)# metro-ring 1
device(config-vlan-2-mrp-1)# name "Metro A"
device(config-vlan-2-mrp-1)# ring-interface ethernet 1/1/1 ethernet 1/1/2
device(config-vlan-2-mrp-1)# enable
device(config-vlan-2)# exit
device(config)# vlan 30
device(config-vlan-30)# tag ethernet 1/1/1 to 1/1/2
device(config-vlan-30)# tag ethernet 1/2/1
device(config-vlan-30)# exit
device(config)# vlan 40
device(config-vlan-40)# tag ethernet 1/1/1 to 1/1/2
device(config-vlan-40)# tag ethernet 1/4/1
device(config-vlan-40)# exit
device(config)# topology-group 1
device(config-topo-group-1)# master-vlan 2
device(config-topo-group-1)# member-vlan 30
device(config-topo-group-1)# member-vlan 40
```

MRP Commands on Switch C

```
device(config)# vlan 2
device(config-vlan-2)# tag ethernet 1/1/1 to 1/2
device(config-vlan-2)# metro-ring 1
device(config-vlan-2-mrp-1)# name "Metro A"
device(config-vlan-2-mrp-1)# ring-interface ethernet 1/1/1 ethernet 1/1/2
device(config-vlan-2-mrp-1)# enable
device(config-vlan-2)# exit
device(config)# vlan 30
device(config-vlan-30)# tag ethernet 1/1/1 to 1/1/2
device(config-vlan-30)# tag ethernet 1/2/1
device(config-vlan-30)# exit
device(config)# vlan 40
device(config-vlan-40)# tag ethernet 1/1/1 to 1/1/2
device(config-vlan-40)# tag ethernet 1/4/1
device(config-vlan-40)# exit
device(config)# topology-group 1
device(config-topo-group-1)# master-vlan 2
device(config-topo-group-1)# member-vlan 30
device(config-topo-group-1)# member-vlan 40
```

MRP Commands on Switch D

```
device(config)# vlan 2
device(config-vlan-2)# tag ethernet 1/1/1 to 1/1/2
device(config-vlan-2)# metro-ring 1
device(config-vlan-2-mrp-1)# name "Metro A"
device(config-vlan-2-mrp-1)# ring-interface ethernet 1/1/1 ethernet 1/1/2
device(config-vlan-2-mrp-1)# enable
device(config-vlan-2)# exit
device(config)# vlan 30
device(config-vlan-30)# tag ethernet 1/1/1 to 1/1/2
device(config-vlan-30)# tag ethernet 1/2/1
device(config-vlan-30)# exit
device(config)# vlan 40
device(config-vlan-40)# tag ethernet 1/1/1 to 1/1/2
device(config-vlan-40)# tag ethernet 1/4/1
device(config-vlan-40)# exit
device(config)# topology-group 1
device(config-topo-group-1)# master-vlan 2
device(config-topo-group-1)# member-vlan 30
device(config-topo-group-1)# member-vlan 40
```

Virtual Switch Redundancy Protocol (VSRP)

VSRP Overview

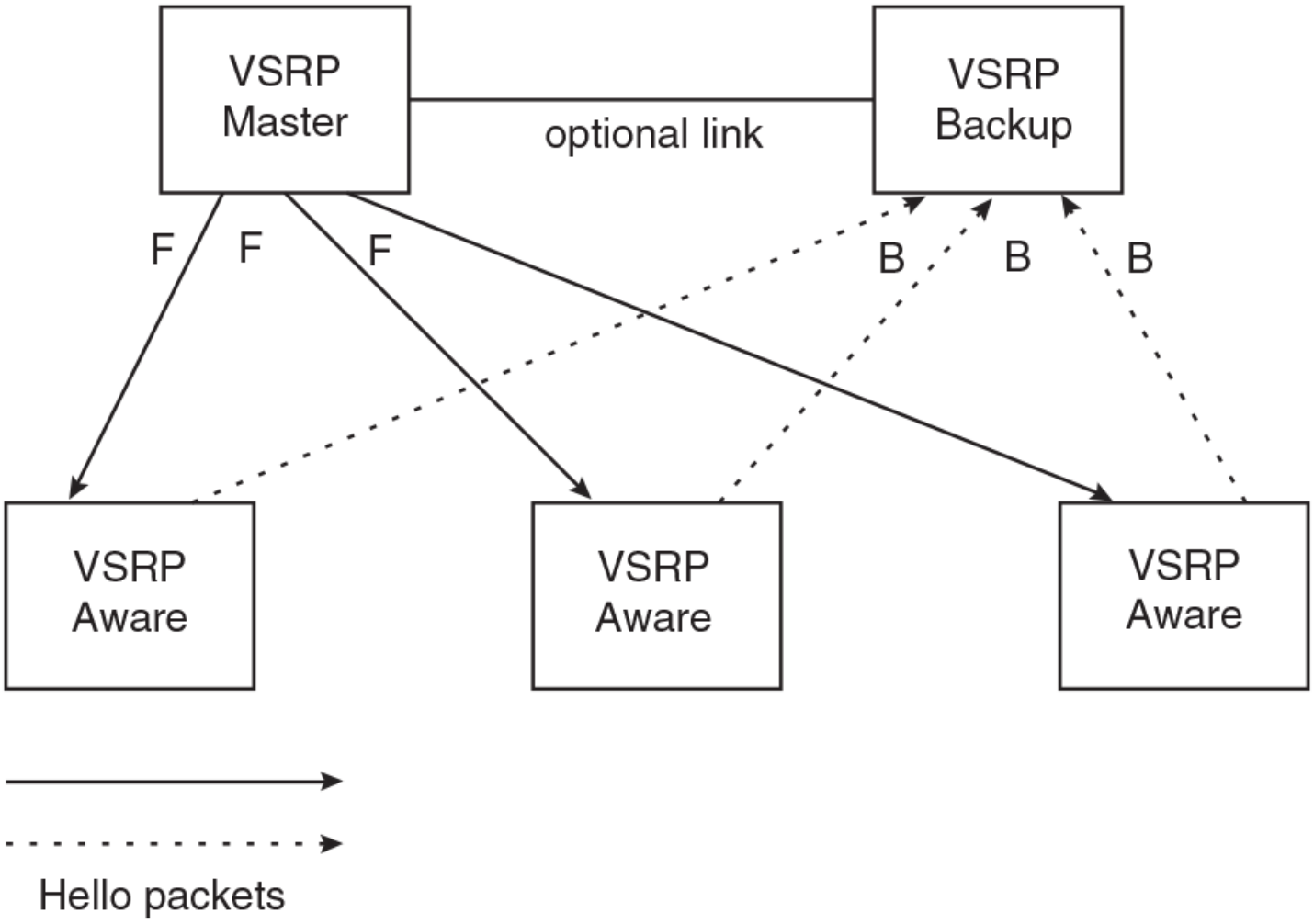
Virtual Switch Redundancy Protocol (VSRP) is a RUCKUS proprietary protocol that provides redundancy and sub-second failover in Layer 2 and Layer 3 mesh topologies. Based on the RUCKUS Virtual Router Redundancy Protocol Extended (VRRP-E), VSRP provides one or more backups for a device. If the active device becomes unavailable, one of the backups takes over as the active device and continues forwarding traffic for the network.

RUCKUS switches support full VSRP as well as VSRP-awareness . A RUCKUS device that is not itself configured for VSRP but is connected to a RUCKUS device that is configured for VSRP, is considered to be VSRP aware.

You can use VSRP for Layer 2, Layer 3, or for both layers. On Layer 3 devices, Layer 2 and Layer 3 share the same VSRP configuration information.

The following example shows an example of a VSRP configuration.

FIGURE 12 VSRP Mesh - Redundant Paths for the Traffic



In this example, two RUCKUS devices are configured as redundant paths for VRID 1. On each of the devices, a Virtual Router ID (VRID) is configured on a port-based VLAN. Since VSRP is primarily a Layer 2 redundancy protocol, the VRID applies to the entire VLAN. However, you can selectively remove individual ports from the VRID if needed.

Following Master election (described below), one of the RUCKUS devices becomes the Master for the VRID and sets the state of all the VLAN ports to Forwarding. The other device is a Backup and sets all the ports in its VRID VLAN to Blocking.

If a failover occurs, the Backup becomes the new Master and changes all its VRID ports to the Forwarding state.

> **NOTE**
> The link between VSRP Master and VSRP Backup is "optional" in the above diagram. However, if the VSRP-aware device is a RUCKUS ICX device, this link is required and recommended. This is due to the need for interoperability between devices of these two platforms having different default timers. The link between the VSRP Master and Backup guarantees that the VSRP Hello message is flowing between the VSRP Master and the VSRP Standby directly to cause VSRP transition instead of relying on VSRP-Aware devices to forward and risk missing the VSRP Hello message.

Other RUCKUS devices can use the redundant paths provided by the VSRP devices. In this example, three RUCKUS devices use the redundant paths. A RUCKUS device that is not itself configured for VSRP but is connected to a RUCKUS device that is configured for VSRP, is VSRP aware . In this

example, the three RUCKUS devices connected to the VSRP devices are VSRP aware. A RUCKUS device that is VSRP aware can failover its link to the new Master in sub-second time, by changing the MAC address associated with the redundant path.

When you configure VSRP, make sure each of the non-VSRP RUCKUS devices connected to the VSRP devices has a separate link to each of the VSRP devices.

VSRP Configuration Notes and Feature Limitations

- VSRP and IEEE 802.1Q Q-n-Q tagging are not supported together on the same device.
- VSRP and Super Aggregated VLANs are not supported together on the same device.
- The VLAN supports IGMP snooping version 2 and version 3 when VSRP or VSRP-aware is configured on a VLAN.

VSRP Redundancy

You can configure VSRP to provide redundancy for Layer 2 and Layer 3:

- Layer 2 only - The Layer 2 links are backed up but specific IP addresses are not backed up.
- Layer 2 and Layer 3 - The Layer 2 links are backed up and a specific IP address is also backed up. Layer 3 VSRP is the same as VRRP-E. However, using VSRP provides redundancy at both layers at the same time.

The RUCKUS ICX device supports Layer 2 and Layer 3 redundancy. You can configure a device for either Layer 2 only or Layer 2 and Layer 3. To configure for Layer 3, specify the IP address you are backing up.

> **NOTE**
> If you want to provide Layer 3 redundancy only, disable VSRP and use VRRP-E.

Master Election and Failover

Each VSRP device advertises its VSRP priority in Hello messages. During Master election, the VSRP device with the highest priority for a given VRID becomes the Master for that VRID. After Master election, the Master sends Hello messages at regular intervals to inform the Backups that the Master is healthy.

If there is a tie for highest VSRP priority, the tie is resolved as follows:

- Layer 2 devices - The Layer 2 device for which the lowest four octets in the MAC address is highest will become the master. VSRP compares the base MAC addresses of the devices.
- Layer 3 devices - The Layer 3 device for which the virtual routing interface has a higher IP address becomes the master.

VSRP Failover

Each Backup listens for Hello messages from the Master. The Hello messages indicate that the Master is still available. If the Backups stop receiving Hello messages from the Master, the election process occurs again and the Backup with the highest priority becomes the new Master.

Each Backup waits for a specific period of time, the Dead Interval, to receive a new Hello message from the Master. If the Backup does not receive a Hello message from the Master by the time the Dead Interval expires, the Backup sends a Hello message of its own, which includes the Backup's VSRP priority, to advertise the Backup's intent to become the Master. If there are multiple Backups for the VRID, each Backup sends a Hello message.

When a Backup sends a Hello message announcing its intent to become the Master, the Backup also starts a hold-down timer. During the hold-down time, the Backup listens for a Hello message with a higher priority than its own.

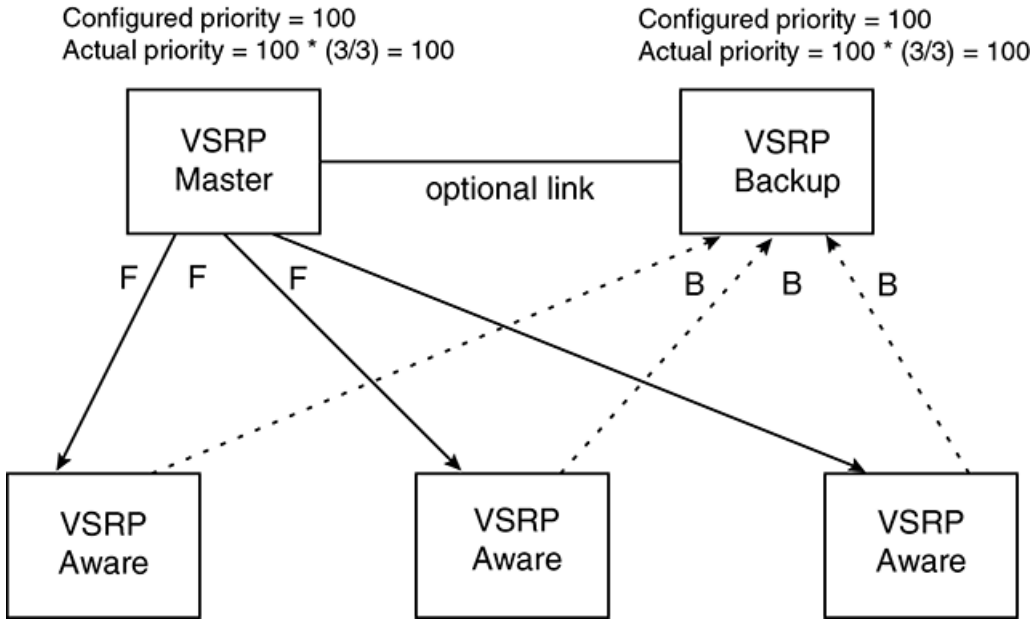
- If the Backup receives a Hello message with a higher priority than its own, the Backup resets its Dead Interval and returns to normal Backup status.
- If the Backup does not receive a Hello message with a higher priority than its own by the time the hold-down timer expires, the Backup becomes the new Master and starts forwarding Layer 2 traffic on all ports.

If you increase the timer scale value, each timer value is divided by the scale value. To achieve sub-second failover times, you can change the scale to a value up to 10. This shortens all the VSRP timers to 10 percent of their configured values.

VSRP Priority Calculation

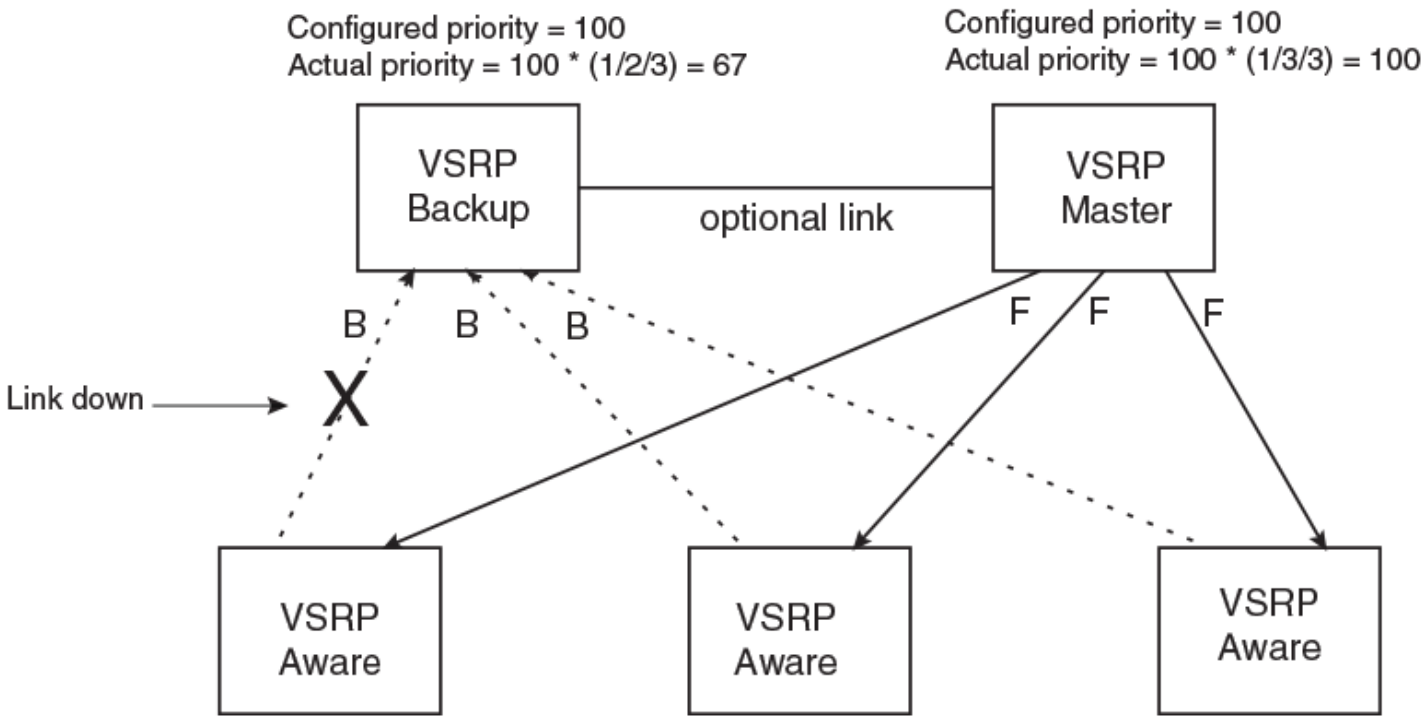
Each VSRP device has a VSRP priority for each VRID and its VLAN. The VRID is used during Master election for the VRID. By default, a device VSRP priority is the value configured on the device (which is 100 by default). However, to ensure that a Backup with a high number of up ports for a given VRID is elected, the device reduces the priority if a port in the VRID VLAN goes down. For example, if two Backups each have a configured priority of 100, and have three ports in VRID 1 in VLAN 10, each Backup begins with an equal priority, 100. This is shown in the following figure.

FIGURE 13 VSRP Priority



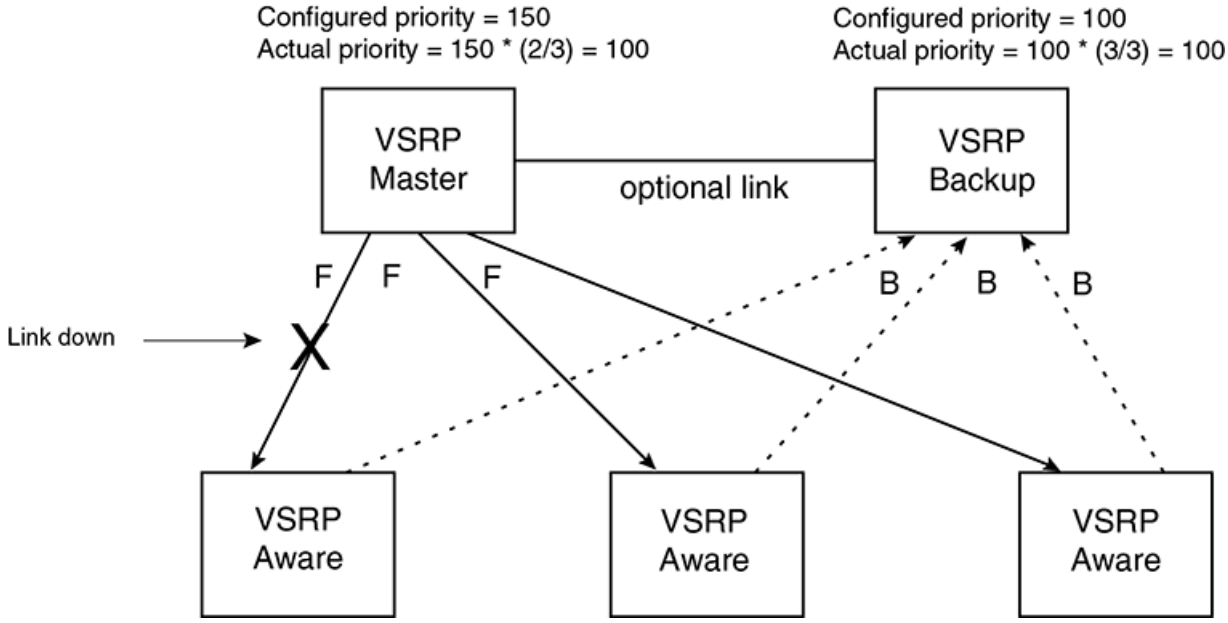
However, if one of the VRID ports goes down on one of the Backups, that Backup priority is reduced. If the Master priority is reduced enough to make the priority lower than a Backup priority, the VRID fails over to the Backup. The following figure shows an example.

FIGURE 14 VSRP Priority Recalculation



You can reduce the sensitivity of a VSRP device to failover by increasing its configured VSRP priority. For example, you can increase the configured priority of the VSRP device on the left in Figure 14 to 150. In this case, failure of a single link does not cause failover. The link failure caused the priority to be reduced to 100, which is still equal to the priority of the other device. This is shown in the following figure.

FIGURE 15 VSRP Priority Bias

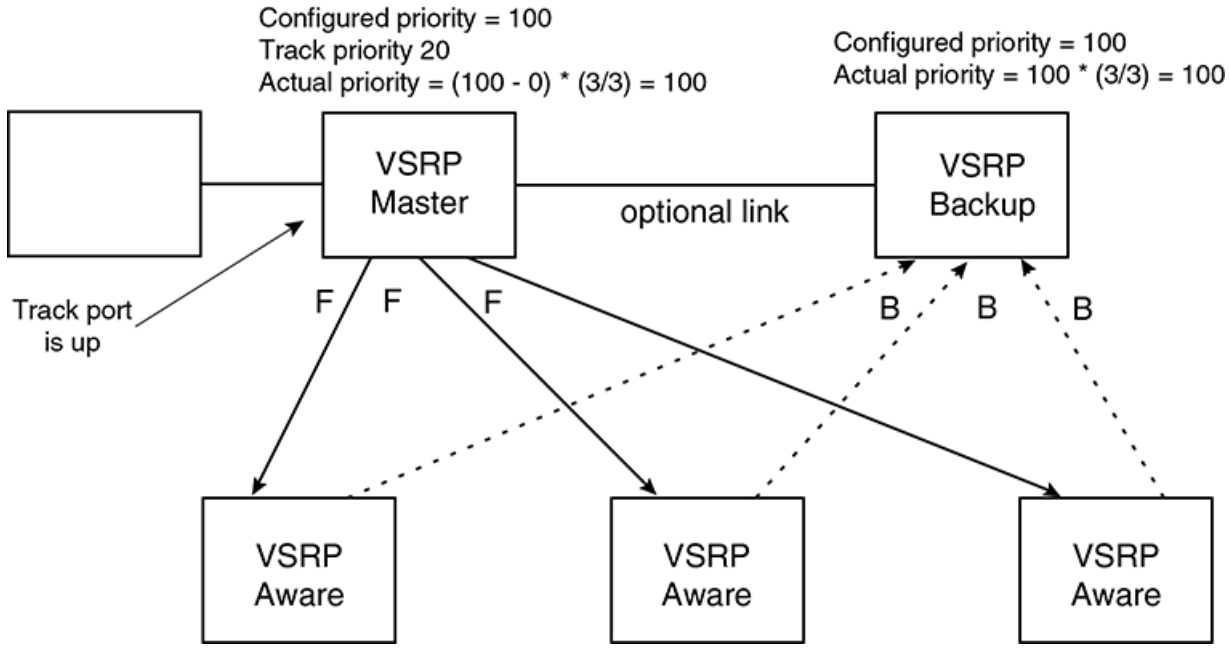


Track Ports

Optionally, you can configure track ports to be included during VSRP priority calculation. In VSRP, a track port is a port that is not a member of the VRID VLAN, but the state of which is nonetheless considered when the priority is calculated. Typically, a track port represents the exit side of traffic received on the VRID ports. By default, no track ports are configured.

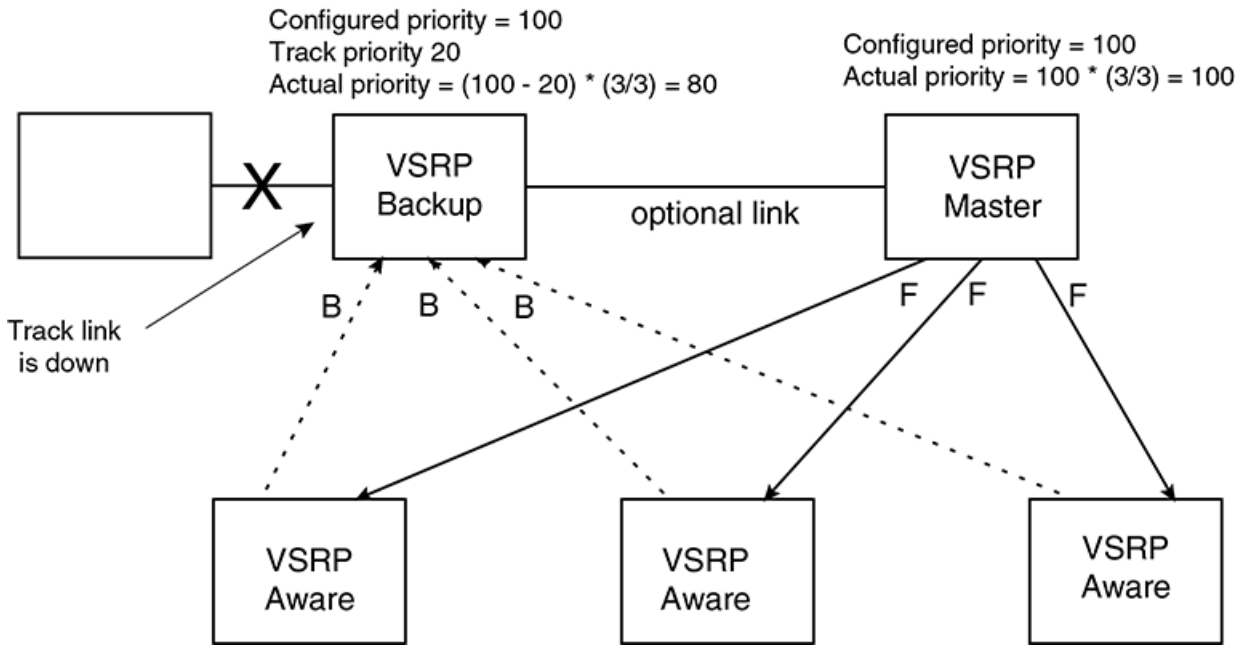
When you configure a track port, you assign a priority value to the port. If the port goes down, VSRP subtracts the track port priority value from the configured VSRP priority. For example, if you configure a track port with priority 20 and the configured VSRP priority is 100, the software subtracts 20 from 100 if the track port goes down, resulting in a VSRP priority of 80. The new priority value is used when calculating the VSRP priority. The following figure shows an example.

FIGURE 16 Track Port Priority



In Figure 16, the track port is up. Since the port is up, the track priority does not affect the VSRP priority calculation. If the track port goes down, the track priority does affect VSRP priority calculation, as shown in the following figure.

FIGURE 17 Track Port Priority Subtracted During Priority Calculation



MAC Address Failover on VSRP-Aware Devices

VSRP-aware devices maintain a record of each VRID and its VLAN. When the device has received a Hello message for a VRID in a given VLAN, the device creates a record for that VRID and VLAN and includes the port number in the record. Each subsequent time the device receives a Hello message for the same VRID and VLAN, the device checks the port number:

- If the port number is the same as the port that previously received a Hello message, the VSRP-aware device assumes that the message came from the same VSRP master that sent the previous message.
- If the port number does not match, the VSRP-aware device assumes that a VSRP failover has occurred to a new master, and moves the MAC addresses learned on the previous port to the new port.

The VRID records age out if unused. This can occur if the VSRP-aware device becomes disconnected from the master. The VSRP-aware device will wait for a Hello message for the period of time equal to the following.

VRID Age = Dead Interval + Hold-down Interval + (3 x Hello Interval)

The values for these timers are determined by the VSRP device sending the Hello messages. If the master uses the default timer values, the age time for VRID records on the VSRP-aware devices is as follows.

3 + 3 + (3 x 1) = 9 seconds

In this case, if the VSRP-aware device does not receive a new Hello message for a VRID in a given VLAN, on any port, the device assumes the connection to the Master is unavailable and removes the VRID record.

VSRP Interval Timers

The VSRP Hello interval, Dead interval, Backup Hello interval, and Hold-down interval timers are individually configurable. You also can easily change all the timers at the same time while preserving the ratios among their values. To do so, change the timer scale. The *timer scale* is a value used by the software to calculate the timers. The software divides a timer value by the timer scale value. By default, the scale is 1. This means the VSRP timer values are the same as the values in the configuration.

Configuring Device Redundancy Using VSRP

Virtual Switch Redundancy Protocol (VSRP) provides device redundancy for specific ports in a port-based VLAN. Configuring VSRP device redundancy in your network leads to faster failover times if an interface goes offline.

VSRP is enabled after assigning a Virtual Routing ID (VRID) on specific ports in a port-based VLAN and setting a backup priority for the device. Repeat this task on each device selected for VSRP redundancy.

NOTE
VSRP is enabled by default on RUCKUS devices, but may be disabled if Virtual Router Redundancy Protocol (VRRP) or VRRP Extended (VRRP-E) is currently enabled.

1. On any device on which you want to configure VSRP service, from privileged EXEC mode, enter global configuration mode by issuing the **configure terminal** command.

   ```
   device# configure terminal
   ```

2. Optionally, globally enable the VSRP protocol.

   This is required only if VSRP was disabled earlier and you want to re-enable it.

   ```
   device(config)# router vsrp
   ```

3. Configure a VLAN by assigning an ID to the VLAN.

   ```
   device(config)# vlan 200
   ```

4. Configure the interfaces on which VSRP service is to be enabled by adding ports to the VLAN.

   ```
   device(config-vlan-200)# tagged ethernet 1/1/1 to 1/1/8
   ```

   In this example, a range of tagged Ethernet interfaces is configured.

5. Assign a VSRP VRID to the VLAN.

   ```
   device(config-vlan-200)# vsrp vrid 1
   ```

6. (Optional) Add additional ports to the VSRP instance.

   ```
   device(config-vlan-200-vrid-1)# include-port ethernet 1/1/10
   ```

7. (Optional) Configure VRID IP address if you are configuring Layer 3 redundancy.

   ```
   device(config-vlan-200-vrid-1)# ip-address 10.10.10.1
   ```

   VSRP does not require you to specify an IP address. If you do not specify an address, VSRP provides Layer 2 redundancy. If you specify an IP address, VSRP provides Layer 2 and Layer 3 redundancy.

8. Designate this device as a backup VSRP device with a priority higher than the default priority.

```
device(config-vlan-200-vrid-1)# backup priority 110
```

   The priority is used to determine the initial VSRP master device. If a VSRP master device goes offline, the backup device with the highest priority will assume the role of master device.

9. Enable a backup router to send hello messages to the master VSRP device.

```
device(config-vlan-200-vrid-1)# advertise backup
```

   By default, backup VSRP devices do not send hello messages to advertise themselves to the master.

10. Enable the VRRP session.

    You can also use the **enable** command to enable the VRRP session.

```
device(config-vlan-200-vrid-1)# activate
```

11. Return to privileged EXEC mode.

```
device(config-vlan-200-vrid-1)# end
```

12. Display VSRP information about the VRID to verify the configuration steps in this task.

```
device# show vsrp vrid 1

Total number of VSRP routers defined: 2
VLAN 200
auth-type no authentication
VRID 1
State    Administrative-status  Advertise-backup  Preempt-mode  save-current
standby  enabled                disabled          true          false

Parameter      Configured   Current   Unit
priority       110          80        (100-0)*(4.0/5.0)
hello-interval 10           1         sec/1
dead-interval  10           3         sec/1
hold-interval  3            3         sec/1
initial-ttl    5            5         hops
next hello sent in 00:00:00.8
Member ports: ethe 1/1/1 to 1/1/8
Operational ports: ethe 1/1/1 to 1/1/6
Forwarding ports:  ethe 1/1/1 to 1/1/6
```

    This is an optional step. Before entering the **show vsrp vrid** command, you may need to activate several VSRP backup devices.

The following example configures VSRP service for VRID 1 on Ethernet interfaces 1/1/1 to 1/1/8 of VLAN 200.

```
device# configure terminal
device(config)# router vsrp
device(config)# vlan 200
device(config-vlan-200)# tagged ethernet 1/1/1 to 1/1/8
device(config-vlan-200)# vsrp vrid 1
device(config-vlan-200-vrid-1)# backup priority 110
device(config-vlan-200-vrid-1)# advertise backup
device(config-vlan-200-vrid-1)# activate
device(config-vlan-200-vrid-1)# end
device# show vsrp vrid 1
```

Configuring Optional VSRP Parameters

You can configure several optional VSRP parameters.

VSRP is configured and enabled.

VSRP is enabled after assigning a Virtual Routing ID (VRID) on specific ports in a port-based VLAN and setting a backup priority for the device. You can configure a number of optional parameters once VSRP is enabled.

NOTE
VSRP is enabled by default on RUCKUS devices, but may be disabled if Virtual Router Redundancy Protocol (VRRP) or VRRP Extended (VRRP-E) is currently enabled.

NOTE
All the steps in this section are optional.

1. On any device on which you want to configure, from privileged EXEC mode, enter global configuration mode by issuing the **configure terminal** command.

   ```
   device# configure terminal
   ```

2. Configure a VLAN by assigning an ID to the VLAN.

   ```
   device(config)# vlan 200
   ```

3. Assign a VSRP VRID to the VLAN.

   ```
   device(config-vlan-200)# vsrp vrid 1
   ```

4. Configure a Backup to save the VSRP timer values received from the master instead of the timer values configured on the backup.

   ```
   device(config-vlan-200-vrid-1)# save-current-values
   ```

5. Configure how many hops the packet can traverse before being dropped.

   ```
   device(config-vlan-200-vrid-1)# initial-ttl 5
   ```

6. Configure the number of seconds between hello messages from the master to the backups for a given VRID.

   ```
   device(config-vlan-200-vrid-1)# hello-interval 10
   ```

7. Configure the number of seconds a backup waits for a Hello message from the master before determining that the master is offline.

   ```
   device(config-vlan-200-vrid-1)# dead-interval 15
   ```

8. Configure the interval for the backup to send hello messages to the master when the advertisement is enabled.

   ```
   device(config-vlan-200-vrid-1)# backup-hello-interval 180
   ```

9. Change the hold-down time interval.

   The hold-down interval prevents Layer 2 loops from occurring during failover, by delaying the new master from forwarding traffic long enough to ensure that the failed master is really unavailable.

   ```
   device(config-vlan-200-vrid-1)# hold-down-interval 4
   ```

Configuring Authentication on VSRP Interfaces

If the interfaces on which you configure the VRID use authentication, the VSRP packets on those interfaces must also use the same authentication.

A VSRP session must be configured and running.

If you configure your device interfaces to use a simple password to authenticate traffic, VSRP interfaces can be configured with the same simple password, and VSRP packets that do not contain the password are dropped. If your interfaces do not use authentication, neither does VSRP. Repeat this task on all interfaces on all devices that support the VRID.

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Configure the VLAN on which a VSRP VRID is assigned.

```
device(config)# vlan 100
```

3. Enter the simple text password configuration.

```
device(config-vlan-100)# vsrp auth-type simple-text-auth ourpword
```

4. Verify the password.

```
device(config-vlan-200)# show vsrp
VLAN 200
 auth-type simple text password
 VRID 1
 ========
  State      Administrative-status Advertise-backup Preempt-mode save-current
  initialize enabled               enabled          true         false

  Parameter        Configured Current    Unit/Formula
  priority         100        0          (100-0)*(0.0/1.0)
  hello-interval   1          1          sec/1
  dead-interval    3          3          sec/1
  hold-interval    3          3          sec/1
  initial-ttl      2          2          hops

  Member ports:      ethe 1/1/1
  Operational ports: None
  Forwarding ports:  None
  Restart ports:     None
```

Tracking Ports and Setting the VSRP Priority

Configuring port tracking on an exit path interface and setting a priority on a VSRP device enables VSRP to monitor the interface. If the interface goes down, the VRID's VSRP priority is reduced by the amount of the track port priority you specify.

This capability is useful for tracking the state of the exit interface for the path for which the VRID is providing redundancy.

1. Enter the global configuration mode.

```
device# configure terminal
```

2. Optionally, globally enable VSRP.

```
device(config)# router vsrp
```

3. Configure a VLAN by assigning an ID to the VLAN.

```
device(config)# vlan 200
```

4. Configure the interfaces on which VSRP service is to be enabled by adding ports to the VLAN.

```
device(config-vlan-200)# tagged ethernet 1/1/1 to 1/1/8
```

5. Assign a VSRP VRID to the VLAN.

```
device(config-vlan-200)# vsrp vrid 1
```

6. Configure the track port and priority.

```
device(config-vlan-200-vrid-1)# track-port ethernet 1/2/4 priority 4
```

   The priority value is used when a tracked port goes down and the new priority is set to this value. Ensure that the priority value is lower than the priority set for any existing master or backup device to force a renegotiation for the master device.

Disabling Backup Pre-Emption Setting

By default, a backup that has a higher priority than another backup that has become the master can preempt the master, and take over the role of master. If you want to prevent this behavior, disable preemption.

Preemption applies only to backups and takes effect only when the master has failed and a backup has assumed ownership of the VRID. The feature prevents a backup with a higher priority from taking over as master from another backup that has a lower priority but has already become the master of the VRID.

Preemption is especially useful for preventing flapping in situations where there are multiple backups and a backup with a lower priority than another backup has assumed ownership, because the backup with the higher priority was unavailable when ownership changed.

If you enable the non-preempt mode (thus disabling the preemption feature) on all the backups, the backup that becomes the master following the disappearance of the master continues to be the master. The new master is not preempted.

Disabling VSRP Backup Preemption

VRRP backup preemption prevents a backup with a higher priority from taking over as master from another backup that has a lower priority but has already become the master of the VRID.

A VSRP session must be globally enabled using the **router vsrp** command in global configuration mode.

1. Configure a VLAN by assigning an ID to the VLAN.

```
device(config)# vlan 200
```

2. Configure the interfaces on which VSRP service is to be enabled by adding ports to the VLAN.

```
device(config-vlan-200)# tagged ethernet 1/1/1 to 1/1/8
```

3. Assign a VSRP VRID to the VLAN.

```
device(config-vlan-200)# vsrp vrid 1
```

4. Disable preemption on a backup.

```
device(config-vlan-200-vrid-1)# non-preempt-mode
```

VSRP-Aware Security Features

This feature protects against unauthorized VSRP hello packets by enabling you to configure VSRP-aware security parameters. Without VSRP-aware security, a VSRP-aware device passively learns the authentication method conveyed by the received VSRP hello packet. The VSRP-aware device then stores the authentication method until it ages out with the aware entry.

The VSRP-aware security feature enables you to perform the following:

- Define the specific authentication parameters that a VSRP-aware device will use on a VSRP backup switch. The authentication parameters that you define will not age out.
- Define a list of ports that have authentic VSRP backup switch connections. For ports included in the list, the VSRP-aware switch will process VSRP hello packets using the VSRP-aware security configuration. Conversely, for ports not included in the list, the VSRP-aware switch will not use the VSRP-aware security processing.

Of the hello packets do not meet the acceptance criteria, the VSRP-aware device forwards the packets normally, without any VSRP-aware security processing.

Configuring Security Parameters on a VSRP-Aware Device

VSRP-aware security parameters protects against unauthorized VSRP hello packets.

VSRP is configured on the device.

Without VSRP-aware security, a VSRP-aware device passively learns the authentication method conveyed by the received VSRP hello packet. The VSRP-aware device then stores the authentication method until it ages out with the aware entry.

1. From global configuration mode, configure a VLAN by assigning an ID to the VLAN

   ```
   device(config)# vlan 200
   ```

2. Specify an authentication string for VSRP hello packets.

   ```
   device(config-vlan-200)# vsrp-aware vrid 3 simple-text-auth pri-key
   ```

3. Configure the device to flush MAC addresses at the VLAN level instead at the port level. MAC address will be flushed for every topology change received on the VSRP-aware ports.

   This configuration should be used in network in which the RUCKUS switch operates as the VSRP-aware device connecting to another device configured as a VSRP Master. MAC address

   ```
   device(config-vlan-200)# VSRP-aware vrid 3 tc-vlan-flush
   ```

4. Verify the configuration using the **show vsrp-aware vlan** command.

   ```
   device(config-vlan-200)# vsrp-aware vrid 1 tc-vlan-flush
   device(config-vlan-200)# show vsrp aware vlan 200
   Aware Port Listing
      VLAN ID VRID Last  Port Auth  Type        Mac-Flush  Age
       200     1   N/A   no-auth    Configured  Enabled    00:00:00.0
   ```

5. Optionally, display active VRID interfaces.

   ```
   device# show vsrp aware
   Aware port listing
   VLAN ID  VRID  Last Port
   100      1     1/3/2
   200      2     1/4/1
   ```

VSRP Fast Start

VSRP fast start allows non-RUCKUS or non-VSRP aware devices that are connected to a RUCKUS device that is the VSRP master to quickly switchover to the new master when a VSRP failover occurs.

This feature causes the port on a VSRP master to restart when a VSRP failover occurs. When the port shuts down at the start of the restart, ports on the non-VSRP aware devices that are connected to the VSRP master flush the MAC address they have learned for the VSRP master. After a specified time, the port on the previous VSRP master (which now becomes the backup) returns back online. Ports on the non-VSRP aware devices switch over to the new master and learn its MAC address.

Special Considerations when Configuring VSRP Fast Start

Consider the following when configuring VSRP fast start:

- VSRP is sensitive to port status. When a port goes down, the VSRP instance lowers its priority based on the port up fraction. Since the VSRP fast start feature toggles port status by bringing ports down and up it can affect VSRP instances because their priorities get reduced when a port goes down. To avoid this, the VSRP fast start implementation keeps track of ports that it brings down and suppresses port down events for these ports (as concerns VSRP).
- Once a VSRP restart port is brought up by a VSRP instance, other VSRP instances (in master state) that have this port as a member do not go to forwarding immediately. This is a safety measure that is required to prevent transitory loops. This could happen if a peer VSRP node gets completely cut off from this node and assumed master state. In this case, where there are 2 VSRP instances that are in master state and forwarding, the port comes up and starts forwarding immediately. This would cause a forwarding loop. To avoid this, the VSRP instance delays forwarding.

Recommendations for Configuring VSRP Fast Start

The following recommendations apply to configurations where multiple VSRP instances are running between peer devices sharing the same set of ports:

- Multiple VSRP instances configured on the same ports can cause VSRP instances to be completely cut off from peer VSRP instances. This can cause VSRP instances to toggle back and forth between master and backup mode. For this reason, we recommend that you configure VSRP fast start on a per port basis rather than for the entire VLAN.
- We recommend that VSRP peers have a directly connected port without VSRP fast start enabled on it. This allows protocol control packets to be received and sent even if other ports between the master and standby are down.
- The VSRP restart time should be configured based on the type of connecting device since some devices can take a long time to bring a port up or down (as long as several seconds). In order to ensure that the port restart is registered by neighboring device, the restart time may need to be changed to a value higher than the default value of 1 second.

Configuring VSRP Fast Start Globally

VSRP fast start enables non-RUCKUS ICX or non-VSRP aware devices that are connected to a RUCKUS ICX device which is the VSRP master to quickly switch over to the new master when VSRP failover occurs.

VSRP is enabled.

VSRP fast start can be enabled on a VSRP-configured device, either on a VALN to which the VRID of the VSRP-configured device belongs (globally) or on a port that belongs to the VRID.

1. On any device on which you want to configure, from privileged EXEC mode, enter global configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

2. Configure a VLAN by assigning an ID to the VLAN.

```
device(config)# vlan 100
```

3. Assign a VSRP VRID to the VLAN.

```
device(config-vlan-100)# vsrp vrid 100
```

4. Enable VSRP fast start. Globally configure a VSRP-configured device to shut down its ports when a failover occurs, and restart after a specified time. This will shut down all the ports, with the specified VRID, that belong to the VLAN when failover occurs.

```
device(config-vlan-100-vrid-100)# restart-ports 5
```

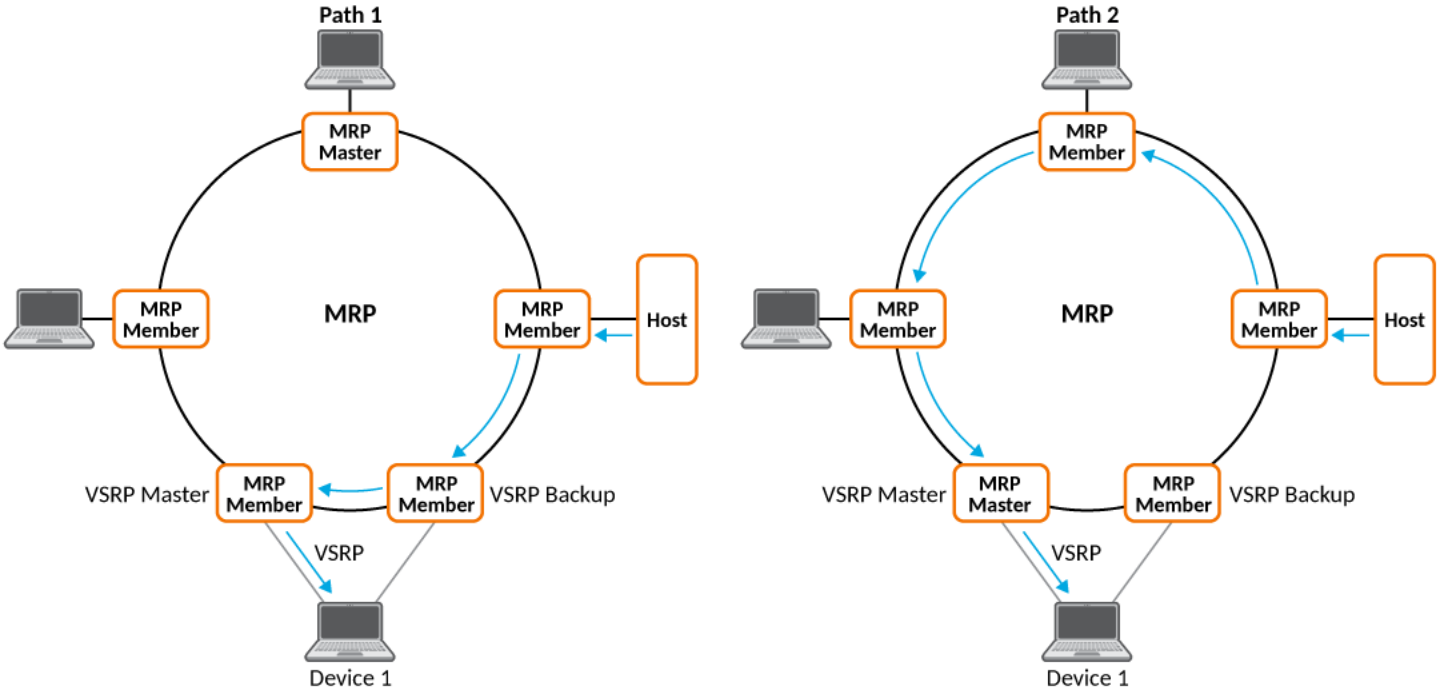
5. Verify the configuration using **show vsrp vrid** command.

```
device# show vsrp vrid 100
VLAN 100
  auth-type no authentication
  VRID 100
  ========
   State      Administrative-status Advertise-backup Preempt-mode save-current
   master     enabled               disabled         true         false
   Parameter        Configured Current    Unit/Formula
   priority         100        50         (100-0)*(2.0/4.0)
   hello-interval   1          1          sec/1
   dead-interval    3          3          sec/1
   hold-interval    3          3          sec/1
   initial-ttl      2          2          hops
   next hello sent in 00:00:00.3
   Member ports:      ethernet 1/2/5 to 1/2/8
   Operational ports: ethernet 1/2/5 ethernet 1/2/8
   Forwarding ports:  ethernet 1/2/5 ethernet 1/2/8
   Restart ports:     1/2/5(1) 1/2/6(1) 1/2/7(1) 1/2/8(1)
```

VSRP and MRP Signaling

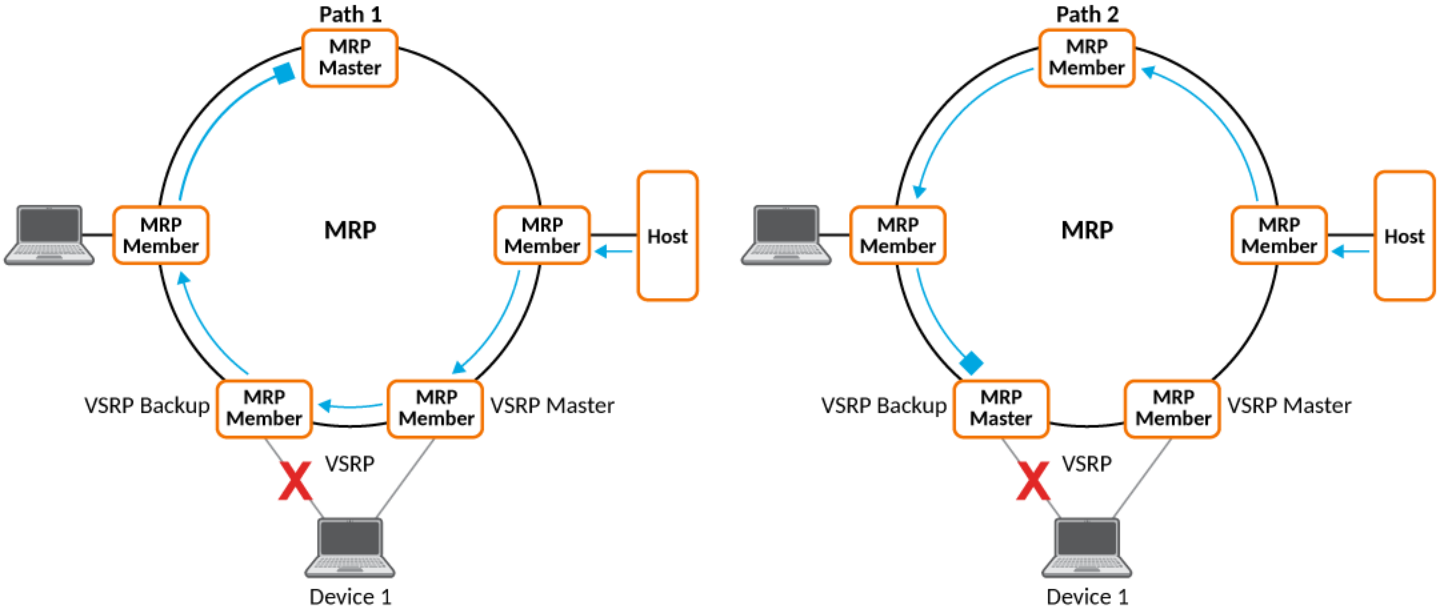
A device may connect to an MRP ring through VSRP to provide a redundant path between the device and the MRP ring. VSRP and MRP signaling ensures rapid failover by flushing MAC addresses appropriately. The host on the MRP ring learns the MAC addresses of all devices on the MRP ring and VSRP link. From these MAC addresses, the host creates a MAC database (table), which is used to establish a data path from the host to a VSRP-linked device. The following figure below shows two possible data paths from the host to Device 1.

FIGURE 18 Two Data Paths from Host on an MRP Ring to a VSRP-Linked Device



If a VSRP failover from master to backup occurs, VSRP needs to inform MRP of the topology change; otherwise, data from the host continues along the obsolete learned path and never reach the VSRP-linked device, as shown in the following figure.

FIGURE 19 VSRP on MRP Rings that Failed Over

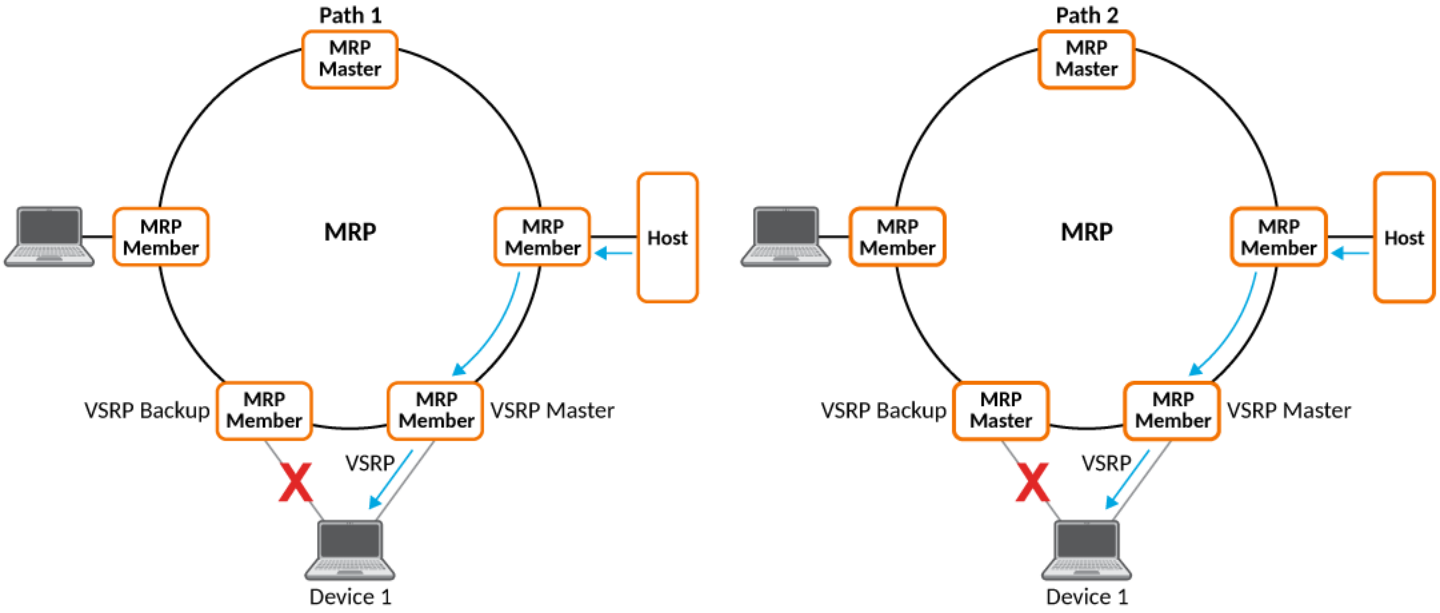


A signaling process for the interaction between VSRP and MRP ensures that MRP is informed of the topology change and achieves convergence rapidly. When a VSRP node fails, a new VSRP master is selected. The new VSRP master finds all MRP instances impacted by the failover. Then each MRP instance does the following:

- The MRP node sends out an MRP PDU with the mac-flush flag set three times on the MRP ring.
- The MRP node that receives this MRP PDU empties all the MAC entries from its interfaces that participate on the MRP ring.
- The MRP node then forwards the MRP PDU with the mac-flush flag set to the next MRP node that is in forwarding state.

The process continues until the master MRP node secondary (blocking) interface blocks the packet. Once the MAC address entries have been flushed, the MAC table can be rebuilt for the new path from the host to the VSRP-linked device as shown in the following figure.

FIGURE 20 New Path Established



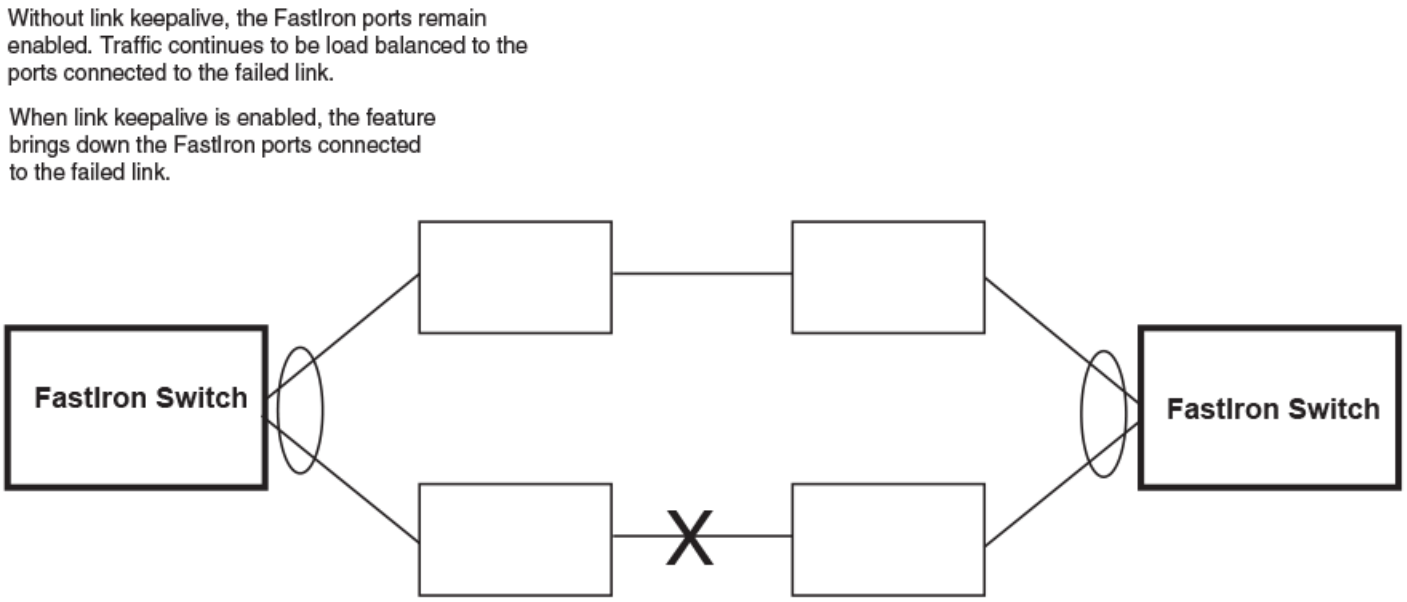
There are no CLI commands used to configure this process.

UDLD

UDLD Overview

Uni-Directional Link Detection (UDLD) monitors a link between two RUCKUS devices and brings the ports on both ends of the link down if the link goes down at any point between the two devices. UDLD can also help to detect wiring mistakes when receive and transmit fibers are not connected to the same port on the remote side. This feature is useful for links that are individual ports and for trunk links. The following figure shows an example.

FIGURE 21 UDLD Example



Normally, a RUCKUS device load balances traffic across the ports in a trunk group. In this example, each RUCKUS device load balances traffic across two ports. Without the UDLD feature, a link failure on a link that is not directly attached to one of the RUCKUS devices is undetected by the RUCKUS devices. As a result, the RUCKUS devices continue to send traffic on the ports connected to the failed link.

When UDLD is enabled on the trunk ports on each RUCKUS device, the devices detect the failed link, disable the ports connected to the failed link, and use the remaining ports in the trunk group to forward the traffic.

Ports enabled for UDLD exchange proprietary health-check packets once every second (the keepalive interval). If a port does not receive a health-check packet from the port at the other end of the link within the keepalive interval, the port waits for two more intervals. If the port still does not receive a health-check packet after waiting for three intervals, the port concludes that the link has failed and takes the port down.

UDLD for Tagged Ports

The default implementation of UDLD sends the packets untagged, even across tagged ports. If the untagged UDLD packet is received by a third-party switch, that switch may reject the packet. As a result, UDLD may be limited only to RUCKUS devices, since UDLD may not function on third-party switches.

To solve this issue, you can configure ports to send out UDLD control packets that are tagged with a specific VLAN ID. This feature also enables third-party switches to receive the control packets that are tagged with the specified VLAN. For tagged operation, all of the following conditions must be met:

- A VLAN is specified when UDLD is configured
- The port belongs to the configured VLAN as tagged member
- All the devices across the UDLD link are in the same VLAN

Configuration Notes and Feature Limitations for UDLD

- UDLD is supported only on Ethernet ports.
- UDLD can be enabled on only one VLAN for tagged port.
- The link-keepalive protocol is not supported on Isolated or Community VLAN ports.
- To configure UDLD on a LAG, you must enable and configure the feature on each port of the LAG individually.
- Low UDLD **link-keepalive** interval and retry options are not recommended as they are more sensitive and prone to flaps.
- When UDLD is enabled on a LAG port, LAG threshold is not supported.
- Dynamic trunking is not supported. If you want to configure a LAG that contains ports on which UDLD is enabled, you must remove the UDLD configuration from the ports. After you create the LAG, you can re-add the UDLD configuration.
- If MRP is also enabled on the device, RUCKUS recommends that you set the MRP preforwarding time slightly higher than the default of 300 ms; for example, to 400 or 500 ms.

Configuring UDLD

Uni-Directional Link Detection (UDLD) monitors a link between two RUCKUS devices and brings the ports on both ends of the link down if the link goes down at any point between the two devices.

1. Enable UDLD:
   - On a port for untagged control packets:

     ```
     device(config)# link-keepalive ethernet 1/1/1
     ```

   - On a trunk group:

     ```
     device(config)# link-keepalive ethernet 1/1/1 ethernet 1/1/2
     ```

   - To receive and send UDLD control packets tagged with a specific VLAN ID:

     ```
     device(config)# link-keepalive ethernet 1/1/18 vlan 22
     ```

     This command enables UDLD on port 1/1/18 and allows UDLD control packet tagged with VLAN 22 to be received and sent on port 1/1/18.

     The command does not support LAG virtual interfaces. Attempting to configure them will have no effect.

     NOTE
     You must configure the same VLANs that will be used for UDLD on all devices across the network; otherwise, the UDLD link cannot be maintained.

2. (Optional) Change the link health-check packet send interval.

   ```
   device(config)# link-keepalive interval 4
   ```

3. (Optional) Set how many retries a port to makes when sent health-checks receive no reply.

   ```
   device(config)# link-keepalive retries 10
   ```

Displaying UDLD Information

UDLD information is displayed using several **show** commands.

1. To display UDLD information for all ports, use the **show link-keepalive** command with no options.

   ```
   device# show link-keepalive

   Total link-keepalive enabled ports: 4
   Keepalive Retries: 3    Keepalive Interval: 1 Sec.
   Port    Physical Link   Logical Link    State           Link-vlan
   1/1/1   up              up              FORWARDING      3
   1/1/2   up              up              FORWARDING
   1/1/3   down            down            DISABLED
   1/1/4   up              down            DISABLED
   ```

2. If a port is disabled by UDLD, the change also is indicated in the output of the **show interfaces brief** command.

```
device# show interfaces brief

Port    Link State      Dupl Speed Trunk Tag Priori MAC            Name
1/1/1   Up   LK-DISABLE None None  None  No  level0 0000.00a9.bb00
1/1/2   Down None       None None  None  No  level0 0000.00a9.bb01
1/1/3   Down None       None None  None  No  level0 0000.00a9.bb02
1/1/4   Down None       None None  None  No  level0 0000.00a9.bb03
```

   If the port was already down before you enabled UDLD for the port, the port state is listed as None.

3. To display detailed UDLD information for a specific port, use the **show link-keepalive** command with a port number.

```
device# show link-keepalive ethernet 4/1/1

Current State   : up            Remote MAC Addr  : 0000.00d2.5100
Local Port      : 4/1/1         Remote Port      : 2/1/1
Local System ID : e0927400      Remote System ID : e0d25100
Packets sent    : 254           Packets received : 255
Transitions     : 1             Link-vlan        : 100
```

4. The **show interface ethernet** command also displays the UDLD state for an individual port. In addition, the line protocol state listed in the first line will say "down" if UDLD has brought the port down.

```
device# show interface ethernet 1/1/1

FastEthernet1/1/1 is down, line protocol is up, link keepalive is enabled
  Hardware is FastEthernet, address is 0000.00a9.bbca (bia 0000.00a9.bbca)
  Configured speed auto, actual unknown, configured duplex fdx, actual unknown
  Member of L2 VLAN ID 1, port is untagged, port state is DISABLED
  STP configured to ON, priority is level0, flow control enabled
  mirror disabled, monitor disabled
  Not member of any active trunks
  Not member of any configured trunks
  No port name
  300 second input rate: 0 bits/sec, 0 packets/sec, 0.00% utilization
  300 second output rate: 0 bits/sec, 0 packets/sec, 0.00% utilization
  0 packets input, 0 bytes, 0 no buffer
  Received 0 broadcasts, 0 multicasts, 0 unicasts
  0 input errors, 0 CRC, 0 frame, 0 ignored
  0 runts, 0 giants, DMA received 0 packets
  19 packets output, 1216 bytes, 0 underruns
  Transmitted 0 broadcasts, 19 multicasts, 0 unicasts
  0 output errors, 0 collisions, DMA transmitted 19 packets
```

   In this example, the port has been brought down by UDLD. Notice that in addition to the information in the first line, the port state on the fourth line of the display is listed as DISABLED.

Clearing UDLD Statistics

You can clear UDLD statistics before changing the UDLD configuration.

The **clear link-keepalive statistics** command clears the Packets sent, Packets received, and Transitions counters in the **show link-keepalive ethernet** command output.

To clear UDLD statistics, enter the following **clear link-keepalive statistics** command.

```
device# clear link-keepalive statistics
```

Link Aggregation Group

Overview of Link Aggregation

Link aggregation allows you to bundle multiple physical Ethernet links to form a single logical trunk providing enhanced performance and redundancy. The aggregated trunk is referred to as a Link Aggregation Group (LAG). The LAG is viewed as a single logical link by connected devices, the Spanning Tree Protocol, IEEE 802.1Q VLANs, and so on. When one physical link in the LAG fails, the other links stay up. A small drop in traffic is experienced when the link carrying the traffic fails.

To configure links to form a LAG, the physical links must be of the same speed. Link aggregation can be done by statically configuring the LAG, or by dynamically configuring the LAG using the IEEE 802.1AX Link Aggregation Control Protocol (LACP).

When queuing traffic from multiple input sources to the same output port, all input sources are given the same weight, regardless of whether the input source is a single physical link or a trunk with multiple member links.

Link aggregation maintains the following benefits:

- Increased bandwidth (The logical bandwidth can be dynamically changed as the demand changes.)
- Increased availability
- Load sharing
- Rapid configuration and reconfiguration

You can use a single interface to configure any of the following LAG types:

- Static LAGs: These LAGs are manually-configured aggregate links containing multiple ports. In static link aggregation, links are added into a LAG without exchanging any control packets between the partner systems. The distribution and collection of frames on static links is determined by the operational status and administrative state of the link.
- Dynamic LAGs: A dynamic LAG type uses the Link Aggregation Control Protocol (LACP), to maintain aggregate links over multiple ports. Typically, two partner systems sharing multiple physical Ethernet links can aggregate a number of those physical links using LACP. LACP creates a LAG on both partner systems and identifies the LAG by the LAG ID. All links with the same administrative key, and all links that are connected to the same partner switch become members of the LAG. LACP PDUs are exchanged between ports on each device to determine if the connection is active. The LAG shuts down ports if their connections are no longer active.

  NOTE
  In earlier releases, the LAG functionality was referred to as Trunk Groups.

- Keep-alive LAGs: In a keep-alive LAG, single connection between a single port on two RUCKUS devices is established. In a keep-alive LAG, LACP PDUs are exchanged between the two ports to determine if the connection between the devices is still active. If it is determined that the connection is no longer active, the ports are blocked.

  NOTE
  In earlier releases, the keep-alive LAG functionality was referred to as Single Link LACP.

New LAG configuration procedures supersede any prior configurations procedures for LAGs and Dynamic Link Aggregation. When a RUCKUS device is upgraded to FastIron 08.0.61 or later, any configurations for LAGs or Dynamic Link Aggregation defined in releases prior to FastIron 08.0.61 are converted to a FastIron 08.0.61-compatible (or later) LAG configuration.

LAG Virtual Interface

The LAG virtual interface is a logical interface that represents a bundle of physical interfaces that form a LAG. The primary port that anchored the LAG no longer exists beginning with FastIron 08.0.61. Instead, all features running over a LAG will be anchored on the LAG virtual interface. All member ports of the LAG are treated as secondary ports and therefore, any physical port can be added or removed from the LAG without tearing down the LAG. The LAG virtual interface is created when a LAG is configured.

The LAG can be added, modified, or deleted using the respective commands in the LAG virtual interface mode. The LAG virtual interface and the properties it supports are similar to an Ethernet interface. The configuration of the LAGs using the LAG virtual interface is described in Creating a Link Aggregation Group (LAG) on page 74. Like Ethernet interfaces, the LAG virtual interface also supports multi-interface mode.

Any new configuration or changes made to the LAG virtual interface are propagated to all ports in the LAG, ensuring a consistent configuration of the member ports. Interface-level configurations can be applied only on the LAG virtual interface and not on any member ports.

NOTE
LAG virtual interface is applicable only for static LAGs and dynamic LAGs and is not supported for keep-alive LAGs.

LAG virtual interface is displayed among the list of interfaces in various **show** commands.

To view the LAG virtual interface details, refer to Displaying LAG Information on page 84.

LAG MAC Address

The Layer 2 protocols (xSTP and MRP) running over the LAG use the LAG MAC address as the source MAC address. The first physical port that is being added to the LAG becomes the MAC provider for the LAG virtual interface. When the first physical port (MAC provider) moves out, the next port is chosen as the MAC provider. However, a constant MAC address can be assigned to the LAG virtual interface using the **lag-mac** command.

NOTE
If VE or Layer 3 is configured on the LAG, the **show interface brief** command output displays the stack-mac as the LAG virtual interface MAC address.

LAG Formation Rules

- A port can be a member of only a single LAG, which can be a static, dynamic, or keep-alive LAG.
- RUCKUS ICX devices cannot form a LAG between two stacks using a RUCKUS optical breakout cable because the cable is not supported on a stack.
- In a linear stack, all ports of the same LAG must be connected to the same unit. Otherwise, you may lose connectivity when a stack port is disconnected from the Active stack unit.
- All ports configured in a LAG must be of the same configured speed, (for example, all **auto** or **1g** or **10g**, and so on).
- All ports configured in a LAG must be of the same operational speed.
- LAG formation rules are checked when a static or dynamic LAG is operational.
- All ports configured in a LAG must be configured in the same VLAN.
- All ports configured in a LAG must be configured with the same port attributes. The following lists contain Layer 2, Layer 3, and ACL requirements:
- Layer 2 requirements:

  The LAG is rejected for the following reasons:

  - The LAG ports do not have the same untagged VLAN component.
  - The LAG ports do not share the same VLAN membership or do not share the same uplink VLAN membership.
  - The LAG ports are configured as member ports of another LAG.

- Layer 3 requirements:

  The LAG is rejected if any of the member ports has any Layer 3 configurations, such as IPv4 or IPv6 address, OSPF, RIP, RIPng, IS-IS, route-only, and so on.

- ACL requirements:

  All LAG ports must have the same ACL configurations; otherwise, the LAG is rejected.

- Properties of the member ports that are being added to the LAG must be identical with respect to the following parameters:
  - Port tag type (untagged or tagged port)
  - Configured port speed and duplex
  - Operational port speed and duplex
  - LAG is formed with different speed capability ports, but the speed change is supported only to the least common speed port.
  - TOS-based configuration: On deletion of the LAG, each port inherits the same TOS-based QoS configuration.

  To change port parameters, you must change them on the LAG virtual interface. The software automatically applies the changes to the other ports in the LAG.

- The device on the other end of the LAG link must support the same number of ports in the link.
- A LAG is supported on 1-Gbps, 2.5-Gbps, 5-Gbps, 10-Gbps, 25-Gbps, 40-Gbps, and 100-Gbps ports.
- A LAG is formed with ports that have different speed capabilities. A speed change is supported to the least common speed possible among the physical ports.
- Port assignment on a module need not be consecutive. The port range can contain gaps. For example, you can configure ports 1, 3, and 4 (excluding 2).
- All the ports must be connected to the same physical or logical device at the other end.
- The sFlow configuration can be enabled on an individual port within a LAG. sFlow cannot be configured on LAG virtual interface. sFlow must be configured in LAG mode for each of the member ports separately.
- LAG virtual interface or any member ports of the LAG cannot be configured as sFlow source.
- Error-disabled interface cannot be added to LAG. To add the interface to LAG, recover the interface from error-disabled state.

Error Disable

LAG ports with operational speeds that do not match the LAG virtual interface are error-disabled with the "lag-oper-speed-mismatch" reason. Error-disabled ports can be recovered if the operational speed is adjusted to match the LAG virtual interface speed.

Error Disable Recovery

The error disable auto-recovery mechanism is used to bring the ports out of error-disabled state. You can use the **speed-duplex** command to change the configured speed of the member ports, which will result in an operational speed change implicitly. Depending on the peer speed duplex setting, either the port will come up in a negotiated operational speed or the port will remain remains down if the speed does not match the peer speed configuration. Using the **speed-duplex** command on VLAG interfaces flaps the LAG member ports and all member ports that came up will have some operational speed subsequently.

As different port types are bundled, their port speed capabilities, such as maximum speed and range of speeds supported, are not the same. Therefore, you can use the **speed-duplex** command only if all member ports can support the given speed value. If the configured speed cannot be supported on at least one member port, the **speed-duplex** command is rejected with an appropriate error message.

An interface error disabled for LAG operational speed mismatch can be recovered in the following ways:

- Configure auto recovery timer for lag operational speed mismatch error disable reason.

- Apply a speed change using the **speed-duplex** command at the LAG level to configure the speed and auto-recover the error-disabled interfaces in the LAG.
- Disabling and enabling the interface LAG.
- Disabling and enabling the errdisabled interface/member-port under lag-level.
- Remove the error-disabled interface from the LAG to reset the error-disabled state and keep the port disabled.

LAG Virtual Anchor Speed

The LAG interface retains the reference operational speed as long as the member port is up. When the last port goes down, the operational speed is reset. Depending on the subsequent sequence of ports coming up, the LAG interface reference speed is used.

Since we rely on the first UP port to choose anchor operational speed, the following behaviors are not avoidable:

- Error-disabled ports may change over a system reload because the anchor operational speed can differ depending on the order in which the links are brought up.
- A similar situation can happen when all LAG member ports go down and the ports are subsequently brought up in a different order.

As above behaviors can affect traffic forwarding over the LAG, make use of **speed-duplex** CLI available under vlag interface level to force a desired speed value for a LAG. As **speed-duplex** will be persistent over the reload, the LAG ports are error-disabled due to the operational speed mismatch before and after the reload.

Use Cases

Some ports become error-disabled if their operational speeds differ. Use the **speed-duplex** command to align speeds and select appropriate ports for aggregation to achieve optimal bandwidth.

Some ports gets error disabled if the operational speed of the ports are different. For example, if there are two ports, 1-Gbps and 10-Gbps, the 10-Gbps port will be error disabled because the 1-Gbps port comes up first. If you use the **speed-duplex** command, the operational speed of the 10-Gbps port becomes 1-Gbps to match the lowest common speed. You must select the appropriate combination of ports to be aggregated to obtain the best effective bandwidth.

Configuration Notes for FastIron Devices in a Traditional Stack

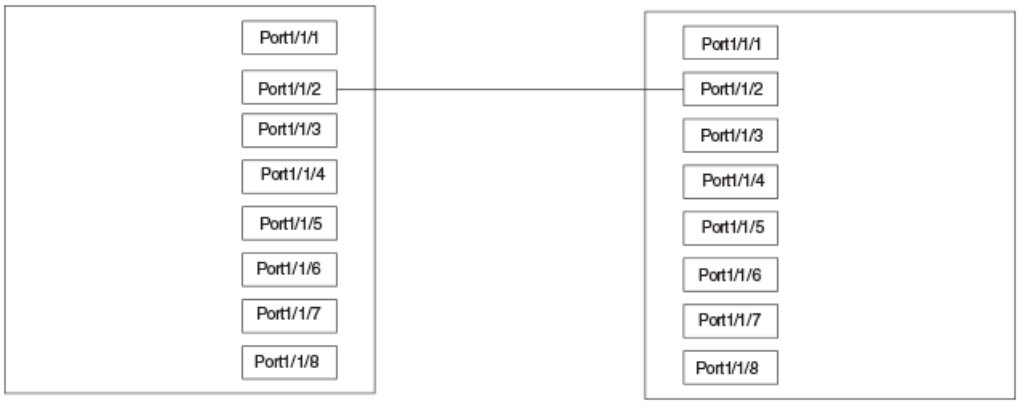
In a RUCKUS traditional stack system, a LAG may have port members distributed across multiple stack units. Both static and dynamic LAGs are supported.

The following notes apply to FastIron stackable devices that are part of a traditional stack:

- If a stack unit fails or is removed from the stack, its static LAG configuration becomes a reserved configuration on the Active Controller. Any remaining ports of the static LAG in the traditional stack continue to function.
- When a new stack unit is added to a traditional stack, the new unit receives the running configuration and LAG information, including a list of ports that are up and are members of a LAG, from the Active Controller.
- Before merging two traditional stack devices, make sure that there are no static LAGs configured between them, which may result in self-looped ports.
- When a traditional stack device with a static LAG partitions into multiple traditional stacks, loops and forwarding errors may occur. In these cases, user intervention is required to remove the loops.
- 10-Gbps links support up to 16 ports in a LAG for stackable units.

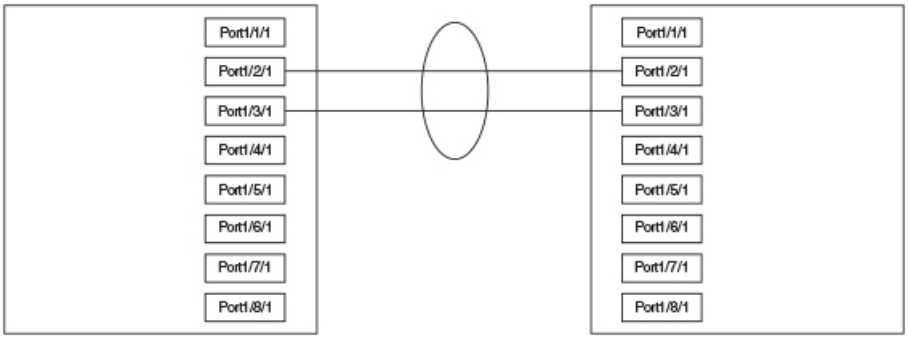
The following figure displays an example of a valid, keep-alive LAG link between two devices. This configuration does not aggregate ports, but uses the LACP PDUs to maintain the connection status between the two ports.

FIGURE 22 Example of a 1-port Keep-Alive LAG



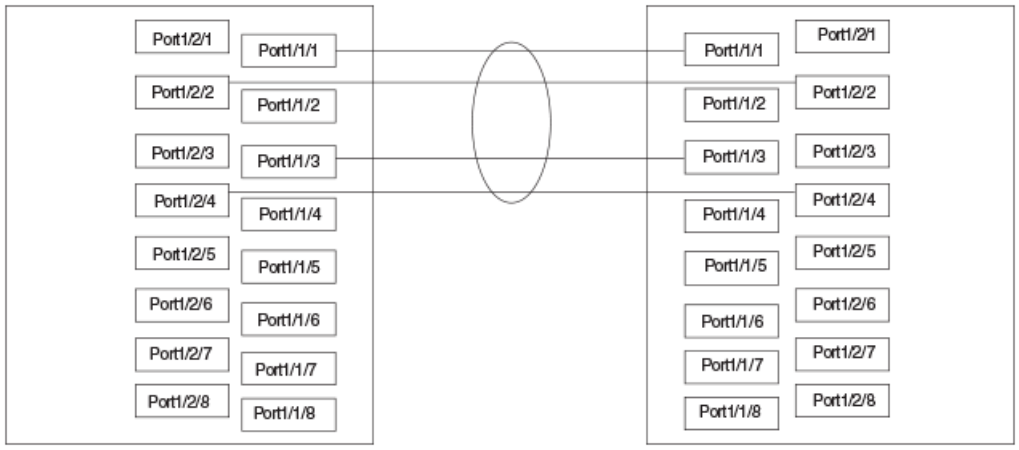
The following figure shows an example of a valid 2-port LAG link between devices where the ports on each end are on the same interface module. Ports in a valid 2-port LAG on one device are connected to two ports in a valid 2-port LAG on another device.

FIGURE 23 Example of a 2-port LAG



The following figure shows an example of two devices connected over a 4-port LAG where the ports on each end of the LAG are on different interface modules.

FIGURE 24 Examples of a Multislot, Multiport LAG



Maximum Number of LAGs

The following table lists the maximum number of LAGs you can configure on a RUCKUS ICX device and the valid number of ports in a LAG. The table applies to static and LACP ports.

TABLE 4 Maximum Number of LAGs

Model	Maximum Number of LAGs		Valid Number of Ports in a Group
	Static	LACP	
ICX 7850 ICX 7650 ICX 7550	256	256	1 to 16 **NOTE** The RUCKUS ICX device can scale up to a maximum of 2048 LAG ports only.
ICX 8200	128	128	1 to 8 **NOTE** The RUCKUS ICX 8200 can scale up to 1024 LAG ports and is also limited to the number of ports on the device.

Upgrade and Downgrade Notes

When a RUCKUS ICX device is upgraded to FastIron 08.0.61, any configurations for LAGs or Dynamic Link Aggregation defined in releases prior to FastIron 08.0.61 will be converted to a FastIron 08.0.61-compatible (or later) LAG configuration.

Auto downgrade is not supported. You must save the old configuration in the previous configuration format before the upgrade followed by the downgrade.

LAG Load Sharing

RUCKUS ICX devices load share across the ports in the LAG group. The method used for the load sharing depends on the device type and traffic type (Layer 2 or Layer 3).

Support for IPv6 When Sharing Traffic Across a LAG Group

Devices that support IPv6 take the IPv6 address for a packet into account when sharing traffic across a LAG group. The load sharing is performed in the same way it is for IPv4 addresses; that is, LAG types with a traffic load that is shared based on IPv4 address information can use IPv6 addresses to determine load sharing.

Load Balancing for Broadcast, Unknown Unicast, and Multicast Traffic

Known unicast traffic is always load balanced across all the ports of a LAG group based on the traffic's Layer 2 and Layer 3 source and destination parameters of the traffic.

Broadcast, unknown unicast, multicast traffic is load balanced based either on source and destination IP addresses and protocol field, or, in some cases, on source and destination IP addresses, protocol field, source MAC address, and destination MAC address.

The load balancing method for bridged traffic varies depending on the traffic type. Load balancing for routed traffic is always based on the source and destination IP addresses and protocol field.

TABLE 5 LAG Load Sharing on RUCKUS ICX Devices

Traffic Type	Load Balancing Method
Layer 2 bridged non-IP	Source and destination MAC addresses
Layer 2 bridged IP non-TCP/UDP	Source and destination MAC addresses, and source and destination IP addresses
Layer 2 bridged IPv4 TCP/UDP	Source and destination IP addresses, and source and destination TCP/UDP ports
Layer 2 bridged IPv4 non-TCP/UDP	Source and destination IP addresses
Layer 2 bridged IPv6 TCP/UDP	Source and destination IP addresses, source and destination TCP and UDP ports, and flow label
Layer 2 bridged IPv6 non-TCP/UDP	Source and destination TCP and UDP ports, and flow label
Layer 3 routed traffic	Source and destination IP addresses and protocol field
Layer 3 multicast	Source and destination IP addresses and protocol field, and, for TCP/UDP packets, Layer 4 source and destination ports

LAG Hashing on Stacking Products

LAG hashing on stacking products is required when multicast routing is configured on a tunnel interface and the IP multicast packets terminate in the tunnel (for example, when the **ip pim**, **ip pim-sparse**, or **ip igmp proxy** multicast routing commands are configured on a tunnel interface).

Stacking trunk hashing for FastIron devices is dynamic. Based on load, traffic is distributed across individual links in the trunk.

Symmetric Load Balancing

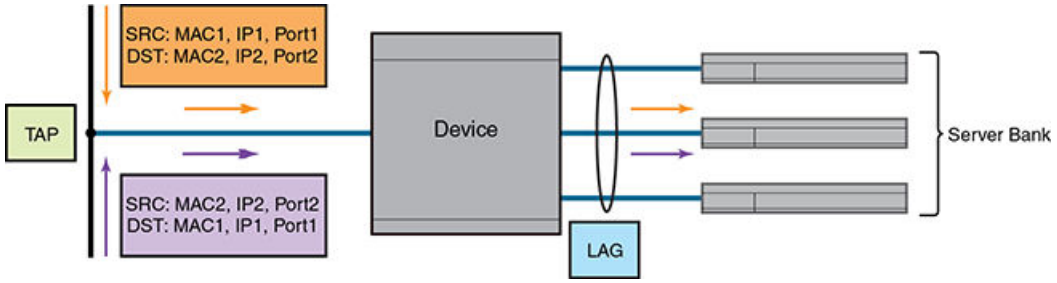
Symmetric load balancing is a mechanism of interchanging the source and destination addresses to ensure that bidirectional traffic specific to a particular source and destination address pair flows out of the same member of a trunk group.

NOTE
Symmetric load balancing is not supported on non-IP data traffic.

For many monitoring and security applications, bidirectional conversations flowing through the system must be carried on the same port of a LAG. For network telemetry applications, network traffic is tapped and sent to a RUCKUS ICX device, which can send hash-selected traffic to the application servers' downstream. Each server analyzes the bidirectional conversations. Therefore, the devices must enable symmetric load balancing to accomplish bidirectional conversations. In addition, the firewall between the RUCKUS ICX devices can be configured to allow the bidirectional conversations per link of the LAG. These network telemetry applications also require symmetric load balancing on the LAGs between the devices.

NOTE
Symmetric load balancing is supported on RUCKUS ICX 7550, and RUCKUS ICX 7850.

FIGURE 25 Symmetric Load Balancing



NOTE
Symmetric load balancing can also be used in case of Equal-cost multi-path routing (ECMP) where the same next hop is selected for bidirectional conversation.

You can enable symmetric load balancing for IPv4 and IPv6 data traffic on RUCKUS ICX devices using the **load-balance symmetric** command.

To confirm whether symmetric load balancing is enabled, use the **show running-config** command.

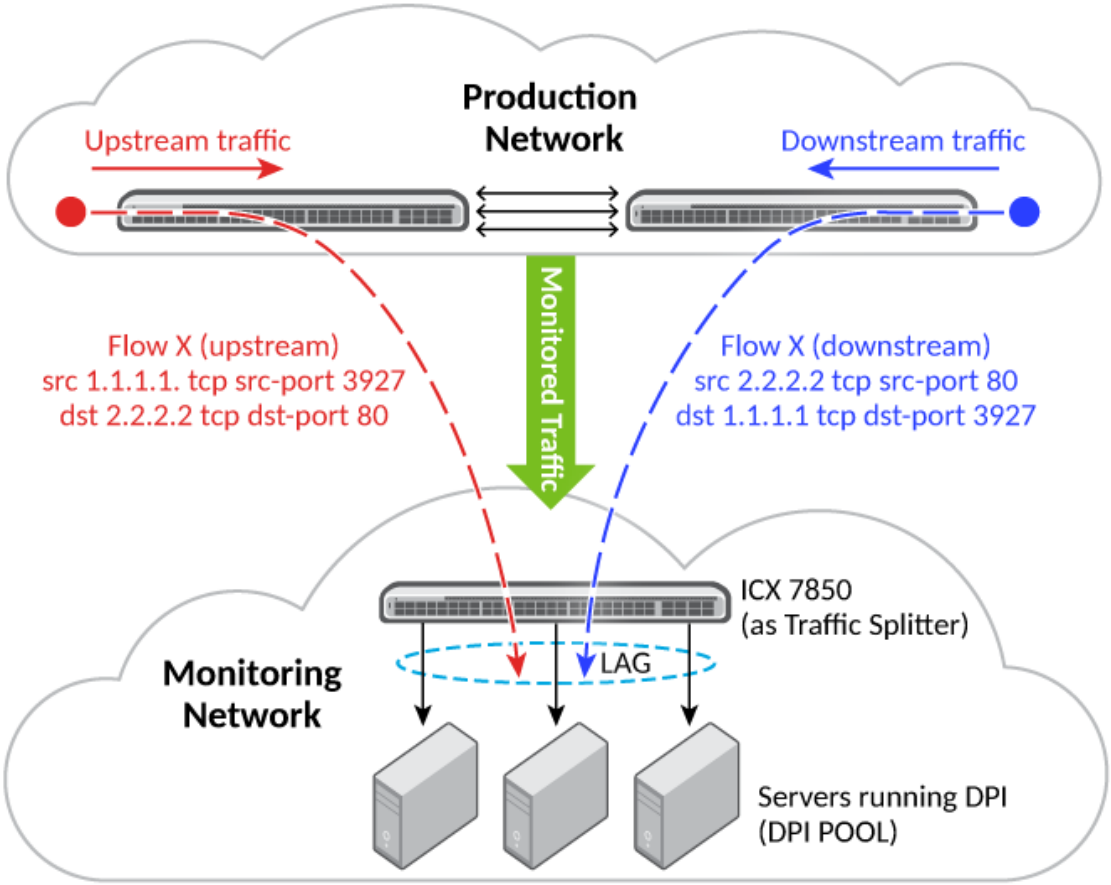
NOTE
Symmetric load balancing is a system-level configuration and may affect load sharing among LAG members as compared to non-symmetric load balancing and the ECMP next hop load sharing by not fairly utilizing all the LAG links. It might also affect load sharing within a stack trunk in case of broadcast, unknown unicast, and multicast (BUM) traffic where the user may not see all the stack trunk member links getting fairly utilized.

TABLE 6 Fields Used for Hash Calculation Based on Packet Types

Packet Type	Hashing Field	Is Symmetric Load Balancing Supported on RUCKUS ICX Platforms?
Non-IP packets	Source MAC address and destination MAC address	No
IPv4/IPv6 packets	SIP, DIP, protocol type, and Layer 4 source or destination ports (only if non-fragmented packet)	Yes
TCP/UDP packets	SIP, DIP, protocol type, and Layer 4 source or destination ports (only if non-fragmented packet)	Yes
IP-in-IP tunnel/GRE packets	Layer 4 source or destination ports (only if non-fragmented packet), SIP, DIP, and protocol type from the inner IP payload	Yes

Use Case: Deploying RUCKUS ICX 7850 as a Traffic Splitter in a DPI Solution

FIGURE 26 Symmetric Load Balancing in RUCKUS ICX 7850



Production network: Traffic flowing in the production network is mirrored onto a few ports that connect to the monitoring network.

Monitoring network: In the monitoring network, the RUCKUS ICX 7850 is deployed as a traffic splitter. There are multiple servers hosting the DPI application and connected to RUCKUS ICX 7850. All monitored traffic is transparently flooded onto the VLAN and is load-balanced among the outgoing ports connected to the DPI pool.

NOTE
The use case assumes that the bidirectional traffic pertaining to the same SIP-DIP pair and the same Layer 4 source-destination pair goes to the same DPI (connected to one of the LAG ports).

After enabling symmetric load balancing, Flow X upstream traffic (with SIP as 1.1.1.1, DIP as 2.2.2.2, Layer 4 source port as 3927, and Layer 4 destination port as 80) and Flow X downstream traffic (with SIP as 2.2.2.2, DIP as 1.1.1.1, Layer 4 source port as 80, and Layer 4 destination port as 3927) will hash to the same member link of the LAG, resulting in the bidirectional conversation going to the same DPI pool.

Configuring a LAG

The following configuration procedures are used to configure a LAG. Depending upon whether you are configuring a static, dynamic or keep-alive LAG, the configuration procedures may or may not apply as described:

- Creating a Link Aggregation Group
- Adding Ports to a LAG
- Configuring the properties of the LAG on the LAG virtual interface.
- Configuring the Load Sharing Type (Optional)
- Specifying the LAG Threshold for a Static LAG Group (Optional)
- Configuring an LACP Timeout (Optional). The LACP Timeout is Long by default.
- Configuring LACP operation mode as active or passive (Optional). The LACP operation mode is Active by default.

Creating a Link Aggregation Group (LAG)

Complete the following steps to create a LAG.

1. Enter the global configuration mode.

   ```
   device# configure terminal
   ```

2. Create a LAG. You can configure any of the following LAG types: static LAG, dynamic LAG, or keep-alive LAG.
   - Configure a static LAG.

     ```
     device(config)# lag blue static id 11
     ```

   - Configure a dynamic LAG.

     ```
     device(config)# lag dblue dynamic id auto
     ```

   - Configure a keep-alive LAG.

     ```
     device(config)# lag kblue keep-alive
     ```

   The LAG ID can be automatically generated and assigned to a LAG using the **auto** option.

   LAG IDs are unique for each LAG in the system. A LAG ID cannot be assigned to more than one LAG. If a LAG ID is already used, the CLI will reject the new LAG configuration and display an error message that suggests the next available LAG ID that can be used.

   NOTE
   The LAG ID parameter is applicable for static and dynamic LAGs only. No explicit configuration of a LAG ID is allowed on keep-alive LAGs.

   The keep-alive LAG option allows you to configure a LAG for use in keep-alive applications similar to the UDLD feature.

3. Add ports to the LAG. The following example shows configuration of a static LAG with two ports.

```
device(config-lag-blue)# ports ethernet 1/3/1 ethernet 1/3/2
```

Upon the addition of the first physical port to a LAG, a LAG virtual interface is created and is available for user configuration.

If there is only one port in the LAG, a single port LAG is formed. For a dynamic LAG, LACP is started for each LAG port. For a keep-alive LAG, no LAG is formed and LACP is started on the LAG port.

A static or dynamic LAG can consist of 1 to 8 or 1 to 16 ports (depending on the device you are using) of the same type and speed that are on any interface module within the RUCKUS chassis. A keep-alive LAG consists of only one port.

The ports can be added to the LAG sequentially as shown in the following example:

```
device(config-lag-blue)# ports ethernet 1/3/1 ethernet 1/2/2 ethernet 1/4/3 ethernet 1/3/4
```

A range of ports from a single interface module can be specified. In the following example, Ethernet ports 1, 2, 3 and 4 on the interface module in slot 3 are configured in a single LAG:

```
device(config-lag-blue)# ports ethernet 1/3/1 to 1/3/4
```

Additionally, you can mix a range of ports from one interface module with individual ports from other interface modules to form a LAG as shown in the following:

```
device(config-lag-blue)# ports ethernet 1/3/1 to 1/3/4 ethernet 1/2/2
```

Configuring LAG Virtual Interface

A LAG virtual interface allows you to enter the LAG virtual interface mode or multi-LAG virtual interface mode. To configure a LAG virtual interface, enter the following commands. All commands applicable on the physical port is available in the LAG virtual interface mode and the same applies for multi-LAG virtual interface commands as well.

1. Enter the global configuration mode.

```
device# configure terminal
```

2. Create a LAG. You can configure any of the following LAG types: static LAG or dynamic LAG.

```
device(config)# lag blue dynamic id 11
```

3. Add ports to the LAG.

```
device(config-lag-blue)# ports ethernet 1/3/1 ethernet 1/3/2
```

Upon the addition of the first physical port to a LAG, a LAG virtual interface is created and is available for user configuration.

4. Configure a LAG virtual interface which allows you to enter the LAG virtual interface mode or multi-LAG virtual interface mode.

```
device(config)# interface lag 11
device(config-lag-if-lg11)#
```

The following example shows multi-LAG virtual interface configuration.

```
device(config)# interface lag 1 lag 5 lag 2
device(config-lag-mif-lg1-lg2,lg5)#
```

You can configure the properties of the LAG on the LAG virtual interface and the changes made to the LAG virtual interface is propagated to all ports in the LAG.

The following options are supported under LAG virtual interface.

```
device(config-lag-if-lg11)# logging enable
device(config-lag-if-lg11)# acl-mirror-port ethernet 3/1/1
device(config-lag-if-lg11)# authentication
device(config-lag-if-lg11)# arp inspection trust
device(config-lag-if-lg11)# bandwidth 10000
device(config-lag-if-lg11)# broadcast limit 1000 kbps
device(config-lag-if-lg11)# dhcp snooping trust
device(config-lag-if-lg11)# dhcp6 snooping trust
device(config-lag-if-lg11)# dot1x port-control auto
device(config-lag-if-lg11)# ethernet loopback
device(config-lag-if-lg11)# ipv6-neighbor inspection trust
device(config-lag-if-lg11)# loop-detection
device(config-lag-if-lg11)# mac access-group 1
device(config-lag-if-lg11)# mac-learn-disable
device(config-lag-if-lg11)# mac-authentication auth-filter 1
device(config-lag-if-lg11)# max-vlan 16
device(config-lag-if-lg11)# monitor both
device(config-lag-if-lg11)# multicast limit 1000 kbps log
device(config-lag-if-lg11)# openflow enable
device(config-lag-if-lg11)# packet-inerror-detect 1000
device(config-lag-if-lg11)# port-name abc
device(config-lag-if-lg11)# protected-port
device(config-lag-if-lg11)# pvst-mode
device(config-lag-if-lg11)# pvstplus-protect
device(config-lag-if-lg11)# restart-vsrp-port 10
device(config-lag-if-lg11)# route-only
device(config-lag-if-lg11)# source-guard enable
device(config-lag-if-lg11)# spanning-tree
device(config-lag-if-lg11)# spanning-tree 802-1w admin-pt2pt-mac
device(config-lag-if-lg11)# spanning-tree 802-1w admin-edge-port
device(config-lag-if-lg11)# spanning-tree root-protect
device(config-lag-if-lg11)# spanning-tree designated-protect
device(config-lag-if-lg11)# speed-duplex
device(config-lag-if-lg11)# stp-bpdu-guard
device(config-lag-if-lg11)# stp-protect
device(config-lag-if-lg11)# tag-profile enable
device(config-lag-if-lg11)# trust dscp
device(config-lag-if-lg11)# unknown-unicast limit 1000 kbps
device(config-lag-if-lg11)# vlan-config add all-tagged
```

Configuring a Layer 3 Link Aggregation Group (LAG)

FastIron devices with Layer 3 images support Layer 3 LAGs, which are used for routing and not switching.

Perform the following steps to enable routing on a LAG:

1. Enter the global configuration mode.

```
device# configure terminal
```

2. Create a LAG.

```
device(config)# lag blue dynamic id 11
```

3. Add ports to the LAG.

```
device(config-lag-blue)# ports ethernet 1/3/1 ethernet 1/3/2
```

Upon the addition of the first physical port to a LAG, a LAG virtual interface is created and is available for user configuration.

4. Configure a LAG virtual interface which allows you to enter the LAG virtual interface mode or multi-LAG virtual interface mode.

```
device(config)# interface lag 11
```

5. Enable routing on the LAG.

```
device(config-lag-if-lg11)# route-only
```

6. Assign an IP address for the LAG.

```
device(config-lag-if-lg11)# ip address 25.0.0.2/24
```

The following example shows the creation of a dynamic LAG that is used for routing on a FastIron device with Layer 3 image.

```
device(config)# lag ruckus-LAG dynamic id 55
device(config-lag- ruckus-LAG)# ports ethernet 4/1/4 ethernet 1/1/1 ethernet 2/1/3 ethernet
3/1/4
device(config-lag- ruckus-LAG)# exit
device(config)# interface lag 55
device(config-lag-if-lg55)# route-only
device(config-lag-if-lg55)# ip address 25.0.0.2/24
```

Configuring an LACP Timeout

In a dynamic or keep-alive LAG, a port timeout can be configured as short (3 seconds) or long (90 seconds). After you configure the port timeout, the port remains in that timeout mode even it is up or down.

All the ports in a LAG must have the same timeout mode. This requirement is checked when the LAG is enabled on the ports.

Complete the following steps to configure a port for a short LACP timeout.

1. Enter the global configuration mode.

```
device# configure terminal
```

2. Create a dynamic LAG.

```
device(config)# lag blue dynamic id 1
```

3. Add ports to the LAG.

```
device(config-lag-blue)# ports ethernet 1/3/1 ethernet 1/3/2
```

4. Configure the timeout mode for the port.

```
device(config-lag-blue)# lacp-timeout short
```

The following example configures a port for a short LACP timeout.

```
device(config)# lag blue dynamic id 1
device(config-lag-blue)# lacp-timeout short
LAG blue un-deployed successfully!
LAG blue deployed successfully!
device(config-lag-blue)#
```

Specifying the LAG Threshold for a LAG

You can configure the RUCKUS device to disable all of the ports in a LAG when the number of active member ports drops below a specified threshold value. When a LAG is shut down because the number of ports drops below the configured threshold, the LAG is kept intact and it is re-enabled if enough ports become active to reach the threshold. For example, if a LAG has eight ports, and the threshold for the LAG is five, the LAG is disabled if the number of active ports in the LAG drops below five. If the LAG is disabled, then traffic is forwarded over a different link or LAG.

NOTE
This configuration is only applicable for static LAGs.

Complete the following steps to establish a LAG consisting of four ports, and then establish a threshold of three ports for this LAG.

1. Enter the global configuration mode.

   ```
   device# configure terminal
   ```

2. Create a static LAG.

   ```
   device(config)# lag blue static id 1
   ```

3. Add ports to the LAG.

   ```
   device(config-lag-blue)# ports ethernet 1/3/1 to 1/3/4
   ```

4. Configure the threshold value for the number of active member ports in a LAG.

   ```
   device(config-lag-blue)# trunk-threshold 3
   ```

   In this example, if the number of active ports drops below three, then all the ports in the LAG are disabled.

   When a LAG is shut down because the number of ports drops below the configured threshold, the LAG is kept intact and it is re-enabled if enough ports become active to reach the threshold.

   NOTE
   The **trunk-threshold** command should be configured only at one end of the trunk. If it is set on both sides, link failures will result in race-conditions and they will not function properly.

The following example establishes a LAG consisting of four ports, then establish a threshold for this LAG of three ports.

```
device(config)# lag blue static id 1
device(config-lag-blue)# ports ethernet 1/3/1 to 1/3/4
device(config-lag-blue)# trunk-threshold 3
```

Enabling Ports Within a LAG

Complete the following steps to enable an individual port within a LAG.

1. Enter the global configuration mode.

   ```
   device# configure terminal
   ```

2. Create a LAG.

   You can configure any of the following LAG types: static LAG, dynamic LAG, or keep-alive LAG.

   ```
   device(config)# lag blue dynamic id 1
   ```

3. Add ports to the LAG.

   ```
   device(config-lag-blue)# ports ethernet 1/3/1 ethernet 1/3/2
   ```

4. Enable an individual port within a LAG.

   ```
   device(config-lag-blue)# enable ethernet 1/3/1
   ```

   To enable a port belonging to a keep-alive LAG, you must enable it from the interface level.

   ```
   device(config-lag-test)# interface ethernet 1/7/8
   device(config-if-e1000-1/7/8)# enable
   device(config-if-e1000-1/7/8)#
   ```

Disabling Ports Within a LAG

You can disable an individual port within a LAG using the **disable** command within the LAG configuration.

1. Enter the global configuration mode.

   ```
   device# configure terminal
   ```

2. Create a LAG.

   ```
   device(config)# lag blue dynamic id 11
   ```

3. Add ports to the LAG.

   ```
   device(config-lag-blue)# ports ethernet 1/3/1 ethernet 1/3/2
   ```

4. Disable an individual port.

   ```
   device(config-lag-blue)# disable ethernet 1/3/1
   ```

   To disable a port belonging to a keep-alive LAG, you must configure from the interface level as displayed in the following example.

   ```
   device(config-lag-test)# interface ethernet 1/7/8
   device(config-if-e1000-1/7/8)# disable
   device(config-if-e1000-1/7/8)#
   ```

Removing a Port from a Currently Operational LAG

Ports can be deleted from a currently operational LAG. However, when deleting ports from an operational LAG, you must consider the following limitations:

- If deleting a port will result in the trunk threshold value becoming greater than the number of ports in the LAG, the port deletion will be rejected.
- When you remove a port from an operational LAG, the port is disabled automatically.

Complete the following steps to remove a port from a LAG.

1. Enter the global configuration mode.

```
device# configure terminal
```

2. Create a LAG.

```
device(config)# lag blue static id 1
```

3. Add ports to the LAG.

```
device(config-lag-blue)# ports ethernet 1/3/1 ethernet 1/3/2
```

4. Remove a port from the LAG.

```
device(config-lag-blue)# no ports ethernet 1/3/1
```

   To enable a port belonging to a keep-alive LAG, you must enable it from the interface level.

```
device(config-lag-test)# interface ethernet 1/7/8
device(config-if-e1000-1/7/8)# enable
device(config-if-e1000-1/7/8)#
```

Monitoring LAG Virtual Interface and Individual LAG Port

Once a LAG is created, monitoring across all ports of the LAG can be configured on the LAG virtual interface. You can configure the device to monitor individual ports in a LAG, including Ethernet or named ports. You can monitor each of the member ports individually. If a new port is added to a LAG and the entire LAG is monitored, the new port will also be mirrored by the same port monitoring traffic across the entire LAG.

NOTE
You can use only one mirror port for each monitored LAG port.

Complete the following steps to monitor LAG virtual interface.

1. Enter the global configuration mode.

```
device# configure terminal
```

2. Configure port mirroring.

```
device(config)# mirror-port ethernet 3/1/1
```

3. Create a LAG.

```
device(config)# lag blue dynamic id 1
```

4. Add ports to the LAG.

```
device(config-lag-blue)# ports ethernet 1/1/1 to 1/1/2 ethernet 2/1/3 to 2/1/4
```

5. Configure a LAG virtual interface which allows you to enter the LAG virtual interface mode or multi-LAG virtual interface mode.

```
device(config)# interface lag 1
```

6. Enter the **monitor both** command.

```
device(config-lag-if-lg1)# monitor both
```

7. Enter the **exit** command to exit from the LAG virtual interface mode.

```
device(config-lag-if-lg1)# exit
```

8. To view the running configurations, enter the **show mirror** command.

```
device(config)# show mirror
```

The following example monitors the LAG virtual interface:

```
device(config)# mirror-port ethernet 3/1/1
device(config)# lag blue dynamic id 1
device(config-lag-blue)# ports ethernet 1/1/1 to 1/1/2 ethernet 2/1/3 to 2/1/4
device(config)# interface lag 1
device(config-lag-if-lg1)# monitor both
device(config-lag-if-lg1)# exit
device(config)# show mirror
Mirror port 3/1/1
Input monitoring : (U1/M1) 1 2
Input monitoring : (U2/M1) 3 4
Input monitoring : (LAG) 1
Output monitoring : (U1/M1) 1 2
Output monitoring : (U2/M1) 3
```

Example of Monitoring an Individual LAG Port

To monitor traffic on an individual ports in a LAG, use the following commands.

```
device(config)# lag blue static id 1
device(config-lag-blue)# ports ethernet 1/1/1 ethernet 1/1/47
device(config-lag-blue)# monitor ethe-port-monitored 1/1/47 ethernet 1/1/15 output
```

Assigning a Name to a Port Within a LAG

Complete the following steps to assign a name to an individual port within a LAG.

1. Enter the global configuration mode.

```
device# configure terminal
```

2. Create a LAG.

```
device(config)# lag blue dynamic id 1
```

3. Add ports to the LAG.

```
device(config-lag-blue)# ports ethernet 1/1/1 ethernet 1/1/3
```

4. Assign a name to a port in the LAG.

```
device(config-lag-blue)# port-name "RUCKUS lag" ethernet 1/1/1
```

   A port name with spaces must be enclosed within double quotation marks.

The following example assigns a name to a port in a LAG.

```
device# configure terminal
device(config)# lag blue dynamic id 1
device(config-lag-blue)# ports ethernet 1/1/1 ethernet 1/1/3
device(config-lag-blue)# port-name "RUCKUS lag" ethernet 1/1/1
```

Allowable Characters for LAG Names

When creating a LAG name, you can use spaces in a file or subdirectory name if you enclose the name in double quotes. For example, to specify a subdirectory name that contains spaces, enter a string such as the following: "a long subdirectory name". The maximum length for a string is 64 characters.

The following characters are valid in file names:

- All upper and lowercase letters
- All digits

Any of the following special characters are valid:

- $
- %
- '
- -
- _
- .
- @
- ~
- `
- !
- (
-)
- {
- }
- ^
- #
- &

Enabling sFlow Forwarding on a Port in a LAG

Complete the following steps to enable sFlow forwarding on an individual port within a LAG.

1. Enter the global configuration mode.

```
device# configure terminal
```

2. Create a LAG.

```
device(config)# lag blue static id 1
```

3. Add ports to the LAG.

```
device(config-lag-blue)# ports ethernet 1/1/1 ethernet 1/1/3
```

4. Enable sFlow forwarding on an individual port within a LAG.

```
device(config-lag-blue)# sflow forwarding ethernet 1/1/1
```

The following example enables sFlow forwarding on an individual port within a LAG.

```
device# configure terminal
device(config)# lag blue static id 1
device(config-lag-blue)# ports ethernet 1/1/1 ethernet 1/1/3
device(config-lag-blue)# sflow forwarding ethernet 1/1/1
```

For a keep-alive LAG, sFlow can be enabled only at the interface level and not at a lag context. To configure sFlow for an interface belonging to the keep-alive LAG, configure directly under the interface.

```
device(config-lag-test)# interface ethernet 1/7/8
device(config-if-e1000-1/7/8)# sflow forwarding
device(config-if-e1000-1/7/8)#
```

Setting the sFlow Sampling Rate for a LAG

Complete the following steps to set the sFlow sampling rate for a LAG.

1. Enter the global configuration mode.

```
device# configure terminal
```

2. Create a LAG.

```
device(config)# lag blue static id 1
```

3. Add ports to the LAG.

```
device(config-lag-blue)# ports ethernet 1/3/1
```

4. Set the sFlow sampling rate.

```
device(config-lag-blue)# sflow sample 512
```

The following example set the sFlow sampling rate for a LAG.

```
device# configure terminal
device(config)# lag blue static id 1
device#(config-lag-blue)# ports ethernet 1/3/1
device#(config-lag-blue)# sflow sample 512
```

For a keep-alive LAG, you need to configure sFlow sampling at the interface level and not within the LAG configuration.

```
device#(config-lag-test)# interface ethernet 1/7/8
device#(config-if-e1000-1/7/8)# sflow sample 512
device#(config-if-e1000-1/7/8)#
```

IP Assignment Within a LAG

Layer 3 static or dynamic LAG support IP assignment. All the configurations has to be done on the LAG virtual interface.

The following is a sample configuration:

```
lag lag_dist_a_1 dynamic id 15
 ports ethe 1/1/1 to 1/1/12
 !
router vrrp
```

```
!
interface lag 15
ip address 192.168.10.1 255.255.255.0
ip vrrp vrid 1
 backup priority 50 track-priority 10
 ip-address 192.168.1.10
 activate
```

Renaming an Existing LAG

You can change the name of an existing LAG without causing any impact on the functionality of the LAG.

Complete the following steps to rename a LAG.

1. Enter the global configuration mode.

   ```
   device# configure terminal
   ```

2. Create a LAG.

   ```
   device(config)# lag blue static id 1
   ```

3. Add ports to the LAG.

   ```
   device(config-lag-blue)# ports ethernet 1/1/1 ethernet 1/1/3
   ```

4. Change the name of the LAG.

   ```
   device(config-lag-blue)# update-lag-name blue1
   ```

   The new name provided must be unique and unused. The LAG configuration mode will exit after successful name update.

Displaying LAG Information

You can display LAG information for a RUCKUS ICX device in either a **full** or **brief** mode.

The following example displays the **brief** option of the **show lag** command.

```
device# show lag brief
Total number of LAGs:          2
Total number of deployed LAGs: 2
Total number of trunks created:2 (126 available)
LACP System Priority / ID:     1 / 609c.9fbc.bf14
LACP Long timeout:             90, default: 90
LACP Short timeout:            3, default: 3
LAG            Type    Deploy Trunk Intf     Port List
tosp12         dynamic  Y     1     lg1      e 1/1/5 e 1/1/7
tosp16         static   Y     2     lg2      e 1/1/6 e 1/1/8
```

The following example displays the full option of the **show lag** command.

```
device# show lag
Total number of LAGs:          2
Total number of deployed LAGs: 2
Total number of trunks created:2 (126 available)
LACP System Priority / ID:     1 / 609c.9fbc.bf14
LACP Long timeout:             90, default: 90
LACP Short timeout:            3, default: 3
=== LAG "tosp12" ID 1 (dynamic Deployed) ===
LAG Configuration:
   Ports:         e 1/1/5 e 1/1/7
   Port Count:    2
   Lag Interface: lg1
   Trunk Type:    hash-based
```

```
   LACP Key:      20001
Deployment: HW Trunk ID 1
Port       Link    State   Dupl Speed Trunk Tag Pvid Pri MAC             Name
1/1/5      Down    None    None None  1     No  1    0   609c.9fbc.bf14

1/1/7      Disable None    None None  1     No  1    0   609c.9fbc.bf14

Port       [Sys P] [Port P] [ Key ] [Act][Tio][Agg][Syn][Col][Dis][Def][Exp][Ope
]
1/1/5           1        1   20001   Yes   S   Agg  Syn  No   No   Def  No   Dwn
1/1/7           1        1   20001   Yes   S   Agg  Syn  No   No   Def  No   Dwn
 Partner Info and PDU Statistics
Port          Partner         Partner     LACP      LACP
             System ID         Key     Rx Count  Tx Count
1/1/5    1-0000.0000.0000        4        0         0
1/1/7    1-0000.0000.0000        6        0         0
=== LAG "tosp16" ID 2 (static Deployed) ===
LAG Configuration:
   Ports:         e 1/1/6 e 1/1/8
   Port Count:    2
   Lag Interface: lg2
   Trunk Type:    hash-based
Deployment: HW Trunk ID 2
Port       Link    State   Dupl Speed Trunk Tag Pvid Pri MAC             Name
1/1/6      Down    None    None None  2     No  1    0   609c.9fbc.bf14

1/1/8      Down    None    None None  2     No  1    0   609c.9fbc.bf14
```

The following example displays the LAG virtual interface properties.

```
device# show interfaces lag 1
Lag lg1 is down, line protocol is down
Configured speed Auto, actual None, configured duplex fdx, actual none
Member of L2 VLAN ID 1, port is untagged, port state is None
BPDU guard is Disabled, ROOT protect is Disabled, Designated protect is Disabled
STP configured to ON, priority is level0, mac-learning is enabled
Openflow is Disabled, OpenflowHybrid mode is Disabled
Mirror disabled, Monitor disabled
Mac-notification is disabled
Member of active trunk ports 1/1/10,lg1, Lag Interface is lg1
Member of configured trunk ports 1/1/10,lg1, Lag Interface is lg1
No port name
300 second input rate: 0 bits/sec, 0 packets/sec, 0.00% utilization
300 second output rate: 0 bits/sec, 0 packets/sec, 0.00% utilization
0 packets input, 0 bytes, 0 no buffer
Received 0 broadcasts, 0 multicasts, 0 unicasts
0 input errors, 0 CRC, 0 frame, 0 ignored
0 runts, 0 giants
0 packets output, 0 bytes, 0 underruns
Transmitted 0 broadcasts, 0 multicasts, 0 unicasts
0 output errors, 0 collisions
Relay Agent Information option: Disabled
```

The following example displays the LAG virtual interface properties in brief.

```
device# show interfaces brief lag 11
Port Link State DuplSpeed Trunk Tag PvidPriMAC Name
lg11 Up Forward Full 3G 11 No 1 5 cc4e.24b4.2208
```

The following example displays the LAG statistics in brief.

```
device# show statistics brief lag 11 ethernet 1/1/1 ethernet 1/1/2 ethernet 1/1/3
Port In Packets Out Packets In Errors Out Errors
1/1/1 579796975 579766803 0 0
1/1/2 3091857541 3091894422 0 0
1/1/3 648306647 648679359 0 0
lg11 4319961163 4320340584 0 0
TOTAL 8639922326 8640681168 0 0
```

The following example displays the LAG statistics.

```
device# show statistics lag 11
LAG Counters:
InOctets737501513146 OutOctets737434067904
InPkts1969927245 OutPkts1970065542
InBroadcastPkts1189300210 OutBroadcastPkts1189300758
InMulticastPkts604719045 OutMulticastPkts780764786
InUnicastPkts0 OutUnicastPkts0
InBadPkts0
InFragments0
InDiscards0 OutErrors0
CRC 0 Collisions 0
InErrors0 LateCollisions0
InGiantPkts0
InShortPkts0
InJabber0
InFlowCtrlPkts0 OutFlowCtrlPkts0
InBitsPerSec2833412080 OutBitsPerSec2833412080
InPktsPerSec1207587 OutPktsPerSec1207717
InUtilization75.00% OutUtilization75.00%
```

The following example displays the running config of LAG virtual interface.

```
device# show running-config interface lag 11
interface lag 11
loop-detection shutdown-disable
route-only
ipaddress 1.1.1.1 255.255.255.0
priority 5
trust dscp
!
```

Enabling LAG Hardware Failover

LAG hardware failover reduces the time of packet loss if a LAG member is down, with minimal software intervention, using loopback on the down port. LAG hardware failover is disabled by default and is supported only on ICX 7850 device.

Enter the **failover all** command in the LAG configuration mode to enable LAG hardware failover. **failover next** command enables failover on the next port in LAG.

Complete the following steps to enable LAG hardware failover on all ports.

1. Enter the global configuration mode.

   ```
   device# configure terminal
   ```

2. Create a LAG.

   ```
   device(config)# lag blue dynamic id 11
   ```

3. Add ports to the LAG.

   ```
   device(config-lag-blue)# ports ethernet 1/3/1 ethernet 1/3/2
   ```

4. Enable LAG hardware failover on all ports.

   ```
   device(config-lag-blue)# failover all
   ```

The following example enables LAG hardware failover on all ports.

```
device# configure terminal
device(config)# lag blue dynamic id 11
device(config-lag-blue)# ports ethernet 1/3/1 ethernet 1/3/2
device(config-lag-one)# failover all
```

Preboot eXecution Environment Boot Support

The Preboot eXecution Environment (PXE), also known as Pre-Execution Environment, is an environment to boot devices using a network interface independent of data storage devices (such as hard disks) or installed operating systems. Consider an environment in which a PXE-capable host forms a dynamic LAG with a FastIron device. After the host successfully boots and runs an operating system, the LACP initiates negotiation to form the dynamic LAG for network access. To boot from the network, the host must be able to connect with the FastIron device initially without a dynamic LAG. To enable this, you can configure PXE boot support on one of the member ports of a dynamic LAG. This ensures that the port is logically operational as soon as you connect this port to the host, even when the dynamic LAG is not operating. At this stage, the port is in "force-up" mode and the **show lag** command shows the operational status "Ope" of this port as "Frc". Once the host successfully boots from the network using this port, the dynamic LAG can form to connect the host to the network with the LAG link. Even if the dynamic LAG fails later, this port is brought back to "force-up" mode and remains logically operational.

Enabling PXE Boot Support on a Port

You can configure the member port of a dynamic LAG to be logically operational even when the dynamic LAG is not operating. This enables PXE boot support on this port.

NOTE
The port should be an edge port on which you have not configured protocols such as STP, MRP, and UDLD.

1. Enter the global configuration mode.

```
device# configure terminal
```

2. Create a dynamic LAG.

```
device(config)# lag blue dynamic id 11
```

3. Add ports to the LAG.

```
device(config-lag-blue)# ports ethernet 1/3/1 ethernet 1/3/2
```

4. Enable PXE boot support on the member port.

```
device(config-lag-blue)# force-up ethernet 1/3/1
```

NOTE
You can enable PXE boot support on only one member port of a dynamic LAG.

The following example shows PXE boot support enabled on member port 1/3/1 a dynamic LAG.

```
device# configure terminal
device(config)# lag blue dynamic id 11
device(config-lag-blue)# ports ethernet 1/3/1 ethernet 1/3/2
device(config-lag-blue)# force-up ethernet 1/3/1
```

User-Configured Peer Information per LACP

RUCKUS ICX devices allow users to define their desired peers under the dynamic LAG configuration if they do not want the default first LACP trunk port to be defined as the LAG's peer information record.

In certain cases, when ports of one dynamic LAG are connected to two different LACP peers (different system IDs, or same system ID with different key values), the device forms one LACP trunk per dynamic LAG and the other port is moved to the error disabled state. In a dynamic LAG, each member port stores a record of its peer's LACP information (system priority, system ID, and system key) from the latest LACPDU it received. This information is known as the port's peer information record. Because all member ports of an LACP trunk share the same local and peer information, the dynamic LAG's peer information record can be any one of its unique LACP trunk port's peer information record (system priority, system ID, or system key). If a dynamic LAG has no associated LACP trunk, its peer information record is stored as NULL.

The **peer-info** command is used to configure the peer system ID and system key for a single dynamic LAG.

The following example specifies the desired LACP peer under the dynamic LAG configuration to be defined as the LAG's peer information record.

```
device(config-lag-DUT1)# peer-info sys-mac 609c.609c.609c sys-pri 10 key 29999
Config Peer info (system_priority=10,system_id=609c.609c.609c,system_key=29999) will replace existing
dynamic Peer info (system_priority=1,system_id=609c.9f26.8d20,system_key=20001) for lag 1
```

NOTE
When there is no user configuration, the system makes sure there is only one LACP trunk within one dynamic LAG. It allows the first LACP trunk port's LACP peer information record to be defined as the LAG's peer information record.

NOTE
Run the **show lag** command to view information about the LACP peer's partner system ID (priority and MAC address) and partner system key.

Dynamic LACP Syslog Messages

The syslog messages in the following table are generated when dynamic LACP is configured in the system.

TABLE 7 Dynamic LACP Syslog Messages

Syslog Message	Definition
`<14>1d12h07m57s:System: dynamic lag interface 2/1/12's peer info (priority=1,id=0024.3821.5600,key=10000) mis-matches with lag's peer info (priority=1,id=0024.3821.5600,key=480), set to mismatch Error`	The port 2/1/12 is set to the mismatch error state.
`System: dynamic lag 100, has new peer info (priority=1, id=0024.3821.5600,key=480) (LACPduRcvd)\n`	The system creates a new peer information record for dynamic LAG 100.

Resilient Hashing

Resilient hashing is a load balancing method to minimize the destination path remapping in case of LAG link failure. Resilient hashing works in conjunction with static hashing algorithm.

Static hashing is a conventional method of distributing the traffic within a LAG uniformly so that the volume of traffic sent to every physical link in a LAG is approximately the same. A LAG's member link is selected by calculating a hash-based on packet headers and a subsequent modulo operation based on the number of physical links in the trunk group. If one of the LAG member links fail, due to module number change, the static hashing algorithm might choose a new member link even for those flows which were not hashed to the failed link. The change in the mapping of the destination path may cause traffic disruption in terms of packet loss or packets wrongly delivered even for the flows that were not hashed to the failed link of the LAG.

Resilient hashing addresses the limitation of static hashing (where destination path remapping for traffic flows going out of non-affected member links of a LAG) by using a flow table for selecting an outgoing port of a LAG for a particular flow.

> **NOTE**
> Resilient hashing is supported on RUCKUS ICX 7650 device only.

Resilient hashing provides the following benefits:

- When a member link of a LAG goes down, it does not affect the flows bound to the remaining working member links of the LAG.
- When a new member link is added to a LAG, the destination path remapping is minimized by redistributing some of the existing flows to the new member link.
- Resilient hashing can be used in data center deployments where it is critical for the network to deliver packets in order during LAG link failures.

The following table explains the destination path results for static and resilient hashing.

TABLE 8 Destination Path Outcome for Static Hashing and Resilient Hashing

Size of the Trunk Group	Static Hashing	Resilient Hashing	Remarks
4	10 % 4 = 2 Flow is going out of the LAG member link at index 2.	Flow is assigned to a LAG member based on the flow set table (outgoing LAG member port at index 10 (10 & 127) is selected for this flow.	The size of the original trunk group is 4.
3	10 % 3 = 1 The same flow will go out of LAG member link at index 1.	Flow is still assigned to the same member link of the LAG (assuming that the LAG member port that went down was not carrying this flow)	One member is deleted.
5	10 % 5 = 0 The same flow will go out of LAG member at index 0.	There is a minimal distribution of flows from existing member LAG links to the newly added member link.	One more member is added to the link. The trunk group size is 5 now.

Resilient Hashing Limitations

Resilient hashing has the following limitations:

- Resilient hashing supports uniform traffic distribution only until one of the link fails and another link takes up the traffic. It does not guarantee uniform distribution of traffic across all members of a LAG since it depends on the flow set table in the hardware and the traffic pattern.
- It is applicable to unicast traffic only.
- In a stacking system, symmetric hashing does not work on resilient hashing LAGs. This is a hardware limitation since the resilient hashing flow set table is programmed differently on each of the stack units.

Configuring Resilient Hashing

To configure resilient hashing, perform the following tasks:

1. Enter the global configuration mode.

```
device# configure terminal
```

2. Create a static or dynamic LAG.

```
device(config)# lag test static
```

3. Add ports to the LAG.

```
device(config-lag-test)# ports ethernet 1/1/1 to 1/1/3
```

4. Enable resilient hashing on the LAG.

```
device(config-lag-test)# trunk-type resilient-hash
```

The following example enables resilient hashing on the "test" LAG.

```
device(config)# lag test static id 11
device(config-lag-test)# ports ethernet 1/1/1 to 1/1/3
device(config-lag-test)# trunk-type resilient-hash
```

The following warning messages are displayed when symmetric hashing and resilient hashing exist together in the system.

- User configures symmetric load balancing on a device that also has the resilient hashing enabled.

```
Warning: system has resilient-hash lags, symmetric hashing may not work for RH lags.
```

- User deploys resilient hashing LAG on a device that already has symmetric hashing enabled.

```
Warning: system has symmetric hashing enabled, symmetric hashing may not work on resilient-
hash lag <LAG_NAME>
```

Multi-Chassis Trunking

Multi-Chassis Trunking Overview

Multi-Chassis Trunking (MCT) is an alternative to spanning tree protocols. Spanning tree is a technology that protects the network against loops by blocking necessary ports, and having the network span to relearn topologies when one link fails in a network. MCT is a technology that allows two MCT-supporting switches to cluster together and appear as a single logical device. Trunking is a technology that allows multiple links of a device to appear as one logical link. The combination of MCT and trunking allows for creating a resilient network topology that utilizes all links in the network, creating an ideal network topology for latency-sensitive applications.

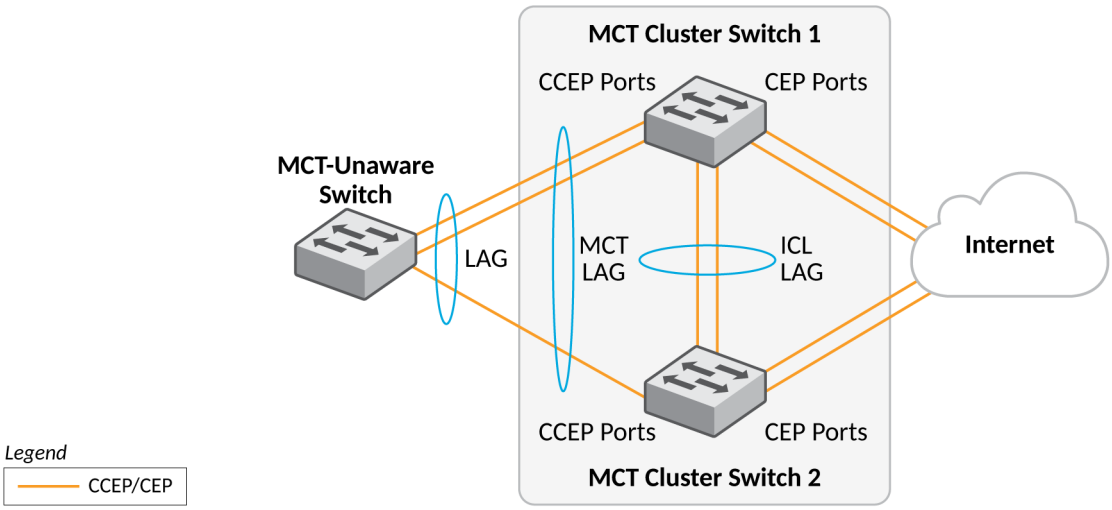
Standard static or dynamic LACP trunks provide link-level redundancy and increased capacity. However, trunks do not provide device-level redundancy. If the device to which the trunk is attached fails, the entire trunk loses network connectivity. Two devices are needed for network resiliency with trunked links to both devices. With spanning tree, one of these trunks would be blocked from use until the failure of the other trunk is detected, taking from 1 to 30 seconds and potentially adding latency and jitter, not only on the affected devices locally, but throughout the span topology. With MCT, member links of the trunk are split and connected to two clustered MCT-supporting switches. MCT has integrated loop detection, which allows all links to be active. If a failure is detected, traffic is dynamically allocated across the remaining links. The failure detection and allocation of traffic occur in sub-second time, without impact on the rest of the network.

MCT inherits all of the benefits of a trunk group and allows multiple physical links to act as a single logical link. The resulting available bandwidth is an aggregate of all the links in the group. Traffic is shared across the links in the group using dynamic flow-based load balancing, and traffic is moved to a remaining link group in sub-seconds if a failure occurs on one of the links. Without the overhead associated with spanning tree, MCT eliminates the single point of failure that exists at the device level when all links of a trunk terminate on the same device. MCT diverts a subset of the links to a second device to provide redundancy and sub-second fault detection at the device level.

How MCT Works

The following figure shows a basic MCT configuration. The MCT originates at a single MCT-unaware server or switch and terminates at two MCT-aware devices.

FIGURE 27 How MCT Works



The MCT process involves the following processes:

- Sub-second failover occurs if a link, module, control plane, or device fails.
- Sub-second failover operates at the physical level.
- Layer 2 and Layer 3 forwarding (when using fast path forwarding) is done at the first hop regardless of the VRRP-E state.
- Load balancing is flow based (it does not involve VLANs sharing across network links).
- Resiliency is supported regardless of the traffic type (Layer 3, Layer 2, or non-IP legacy protocols).
- Interaction with Metro Ring Protocol (MRP) builds larger resilient Layer 2 domains.
- Device-level redundancy is provided in addition to link and modular redundancy.
- Traffic received from an ICL(Inter-Chassis Link) port is not forwarded to the Cluster Client Edge Ports (CCEPs) if the MCT peer device has the ability to reach the same cluster client.
- Traffic received from non-ICL ports is forwarded the same way as traffic from non-MCT devices.
- Known unicast traffic received on Cluster Edge Ports (CEPs) or ICL ports is forwarded to the destination port.
- For broadcast, unknown unicast, and multicast (BUM) traffic received on ICL ports, the forwarding behavior depends on the peer MCT device's ability to reach the same client.
- Broadcast, unknown unicast, and multicast (BUM) traffic received from a CCEP is forwarded as usual, by default, flooding the entire VLAN.
- The cluster ID must be unique when there are multiple clusters interconnected in a topology. For example, in a cascaded Stage 2 MCT cluster, the cluster ID on a stage 1 pair of switches must be different from the cluster ID on a stage 2 pair of switches.

MCT Terminology

Term	Description
Cluster Client Edge Port (CCEP)	A physical port or trunk group interface on an MCT cluster device that is connected to client devices.
Cluster Edge Port (CEP)	A port on an MCT cluster device that belongs to the MCT VLAN and connects to an upstream core switch or router but is neither a CCEP nor an ICL.
Cluster Communication Protocol (CCP)	A proprietary protocol that provides reliable, point-to-point transport to synchronize information between MCT cluster devices. It provides the default MCT control path between the two peer devices. CCP comprises two main components: CCP peer management and CCP client management.

Term	Description
Inter-Chassis Link (ICL)	A single-port or multiport 1-GbE, 10-GbE, or 40-GbE LAG between the two MCT cluster devices. It provides the control path for CCP for the cluster and also serves as the data path between the two devices.
MCT cluster	A pair of devices (standalone switches or stacks) that are clustered together using MCT to appear as a single logical device. The devices are connected as peers through an Inter-Chassis Link (ICL).
MCT cluster client	A device that connects with MCT cluster devices through static or dynamic trunks. It can be a switch or an endpoint server host in the single-level MCT topology or another pair of MCT devices in a multi-tier MCT topology.
MCT cluster device	One of the two devices (or stacks) in an MCT cluster.
MCT peer device	From the perspective of an MCT cluster device, the other device (or stack) in the MCT cluster.
MCT VLANs	VLANs on which MCT cluster clients are operating. Any VLAN that has an ICL port is an MCT VLAN, even if it does not have any clients.
MCT keep-alive VLAN	The VLAN that provides a backup control path if the ICL goes down.
MCT session VLANs	The VLAN used by the MCT cluster for control operations. CCP runs over this VLAN. The interface can be a single link or a LAG. In a LAG, it must be configured on the LAG virtual interface. The MCT session VLAN subnet is not distributed in routing protocols using redistribute commands.
RBridge ID	An RBridge ID is a value assigned to MCT cluster devices and clients that uniquely identifies them and helps associate the source MAC address with an MCT device.

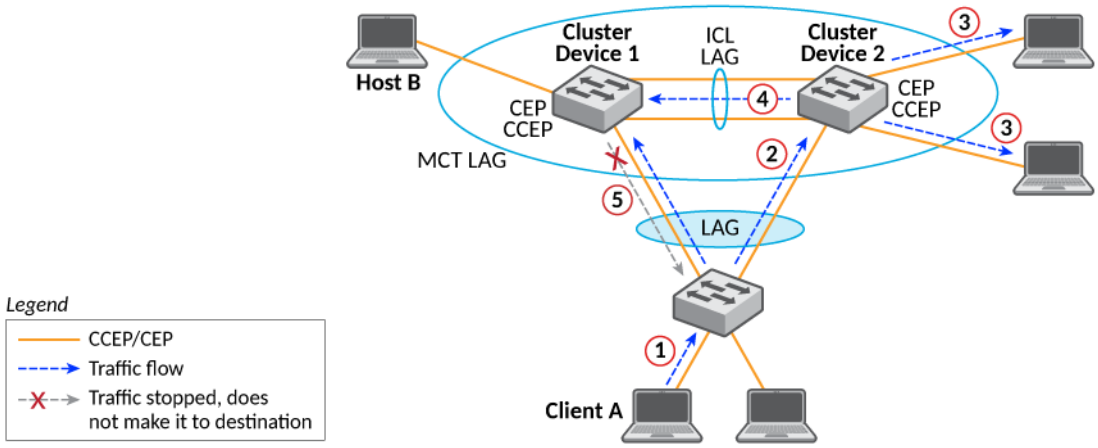
MCT Data Flow

MCT can be deployed in a single-level configuration that includes two MCT cluster devices or in a cascading configuration, where a pair of MCT cluster devices operate as switches, and another pair of cluster devices operates as routers. Refer to Single-Level MCT Configuration Example on page 150 for a single-level illustration and configuration example, and Two-Level MCT Configuration Example on page 153 for a two-level or cascading configuration example.

Broadcast, Unknown Unicast, and Multicast Traffic from a Client Through a CCEP

1. Traffic originates at the client.
2. Because the link between the client switch and the MCT cluster is a trunk, the traffic travels over one physical link. In the example shown in the following figure, the traffic travels over the link toward cluster device 2. The traffic enters the MCT cluster through the CCEP of cluster device 2.
3. The traffic is sent to any local CEPs and CCEPs.
4. It passes to the peer cluster device over the ICL link, where it is sent to the peer device's local CEPs.
5. Traffic does not pass back down to the client through the CCEP.

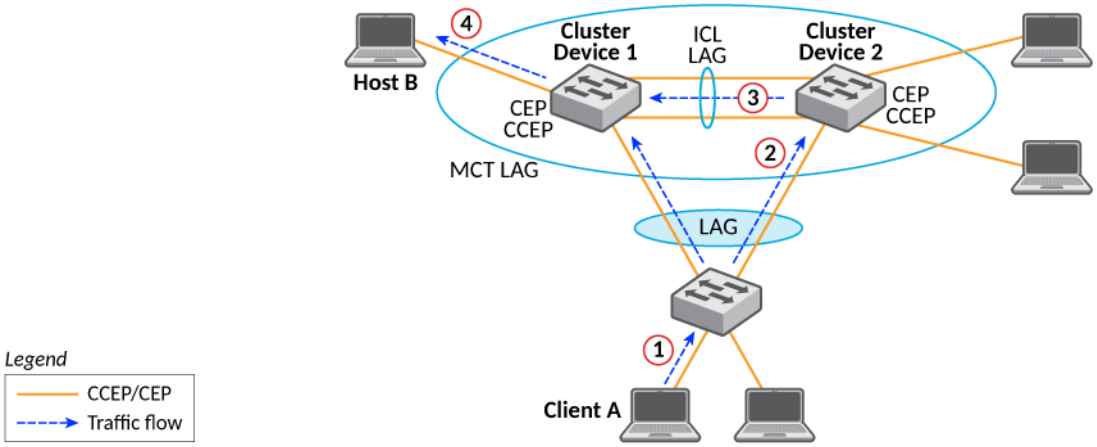
FIGURE 28 MCT Data Flow - BUM Traffic from a Client Through a CCEP



Unicast Traffic from a Client Through a CCEP to a CEP

1. Traffic originates at the client.
2. Because the link between the client switch and the MCT cluster is a trunk, the traffic travels over one physical link. As shown in the following figure, the traffic travels over the link toward cluster device 2. The traffic enters the MCT cluster through the CCEP of cluster device 2.
3. Depending on the destination, the traffic may pass over the ICL link to the other cluster device. In the following figure, the destination is on cluster device 1, so the traffic is forwarded out to the ICL port.
4. The traffic passes out to the destination.

FIGURE 29 MCT Data Flow - Unicast Traffic from a Client Through a CCEP to a CEP

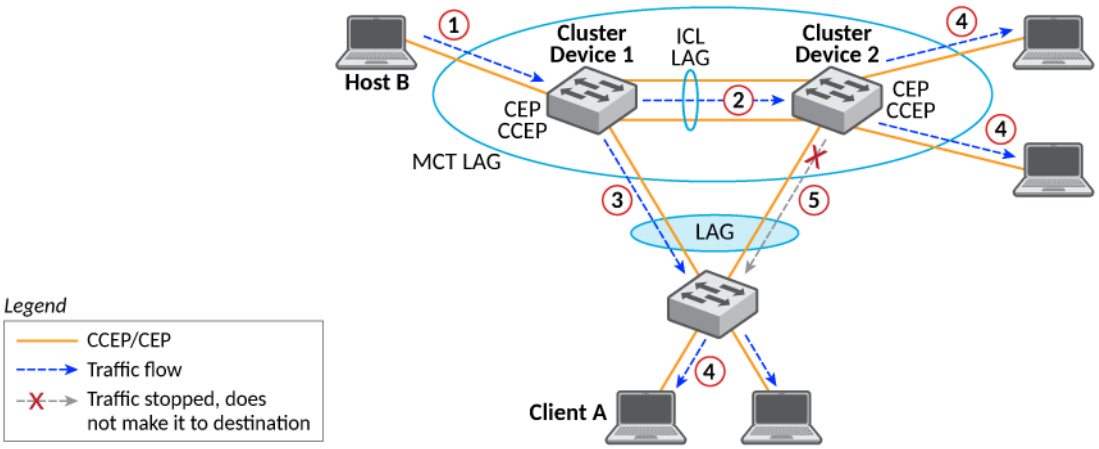


Broadcast, Unknown Unicast, and Multicast Traffic from a Client Through a CEP

1. Traffic originates at the client and enters one of the MCT cluster devices through a CEP.
2. As shown in the following figure, the traffic is sent to the peer cluster device through the ICL link.
3. The traffic is also sent to any local CCEPs and CEPs.

4. Once traffic is received on the peer cluster device, it will be sent to its local CEPs.
5. Traffic does not pass back down to the client through the CCEP.

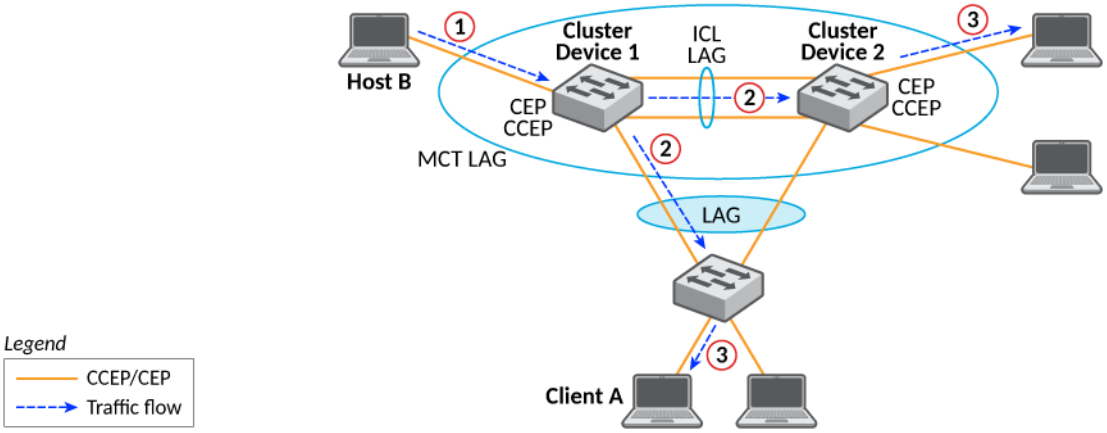
FIGURE 30 MCT Data Flow - BUM Traffic from a Client Through a CEP



Unicast Traffic from a Client Through a CEP to Another CEP or a CCEP

1. Traffic originates at the client and enters one of the cluster devices through the CEP, as shown in the following figure.
2. Depending on the destination, the traffic may pass over the ICL link to the other cluster device, or it may be sent to a local CCEP.
3. The traffic passes out to the destination.

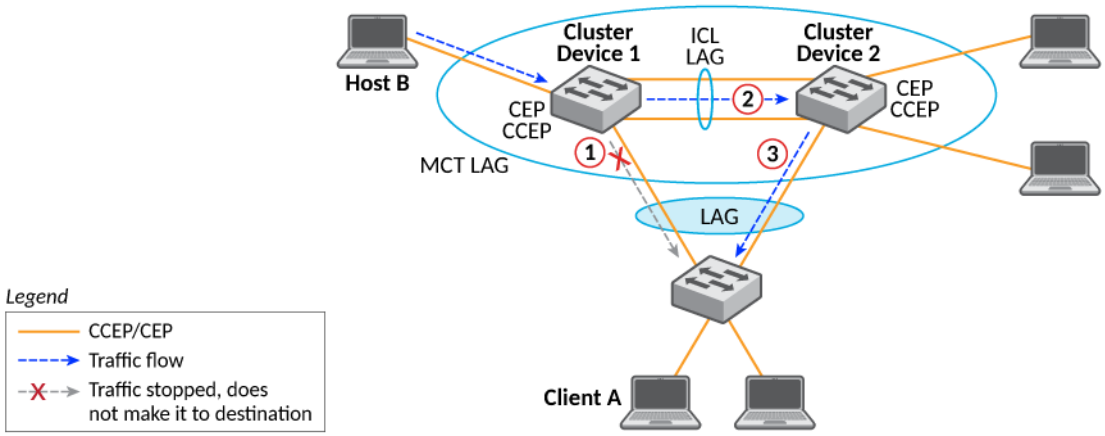
FIGURE 31 MCT Data Flow - Unicast Traffic from a Client Through a CEP to Another CEP or a CCEP



Port Failure on the Cluster Device

1. A CCEP on the cluster device that received the unicast or BUM traffic fails.
2. As shown in the following figure, the traffic is automatically redirected to the other MCT cluster device over the ICL and on to its destinations through CCEPs.

FIGURE 32 MCT Data Flow with Port Failure



MCT and VLANs

MCT relies on the following VLAN types:

- Session VLAN: Provides the control channel for CCP. RUCKUS recommends keeping only ICL ports in the session VLAN. A virtual interface must be configured on the session VLAN for the router image.
- Keep-alive VLAN: Provides a backup control path if the ICL goes down (optional, but strongly recommended).
- MCT VLAN: Serves the customer data traffic. An ICL must belong to every MCT VLAN to provide a data path between two cluster devices. When an ICL is added to a VLAN, it becomes an MCT VLAN.

MCT Feature Interaction and Supported Features

The following FastIron features are supported with MCT. All security features are locally significant and are not synchronized across an MCT cluster.

- LACP on the Cluster Client Edge Port (CCEP).
- VRRP on the CCEP.
- MRP and MRP II, with the restriction that the ICL port cannot be the secondary port of the MRP ring.
- Flooding features (such as VLAN CPU protection and multicast flooding) on MCT VLANs.
- Unidirectional Link Detection (UDLD) as independent boxes (configured independently).
- ARP as independent boxes (configured independently).
- Ingress ACLs on all MCT ports, except for ICL ports. Egress ACLs are supported only on MCT Cluster Edge Ports (CEPs).
- QoS and MAC ACLs and profiles with the same configuration on both cluster devices.
- IPv4 ACLs and rate limits. If the rules are applied on the CCEPs, the same rules must be applied to the CCEP ports on both cluster devices.
- Layer 3 routing. VE with IP address assignment is supported on CCEPs for VRRP.
- Static multiport MAC.
- Multiport authentication and 802.1X on CEPs.
- Static MAC address configuration. Static MAC addresses are programmed on both local and remote peers as static entries.
- Hitless failover. If the failover operation is performed with a cluster configuration, the TCP session is re-established. The MAC addresses from the cluster peer devices are revalidated and programmed accordingly.
- IPv6, VRRP-E (IPv6), and VRRPv3.

Features Not Supported with MCT

- SSTP, SRSTP, and MSTP are not supported on MCT VLANs on MCT clients or core switch connected to MCT peers.
- ACLs on VLAN session (ICL) ports.
- LACP on ICL.
- MSTP, VSRP, and RIP.
- MSDP.
- GRE on the ICL VE interfaces.
- DAI on the CCEPs.
- Host security features (port MAC security, multiport authentication, 802.1X, DAI, DHCP snooping) on CCEPs.
- Multiport ARP on ICL or CCEPs.
- Port MAC security is not supported on CEPs. However, the FastIron devices do not restrict the port MAC security commands to be enabled on the CEPs.
- Web authentication on MCT VLANs.

MCT Board Type Compatibility

Multi-Chassis Trunking (MCT) requires similar functionality between peer devices to ensure compatibility. A type-length-value (TLV) is introduced to differentiate the board types used in some hardware devices that can run the same software versions but have different functional capabilities.

The ICX 7550, ICX 7650, and ICX 7850 devices can run the same software versions although some of their capabilities are different. If these devices are used as MCT peers, some basic MCT functionality, such as the MAC table size, can differ between peers. To avoid any issues, the device board type information is added in new fields as a TLV in the version information message used during the handshake signal, and this identifies the hardware. The board type check is performed before the Cluster Communication Protocol (CCP) is activated to prevent any unnecessary connection operations.

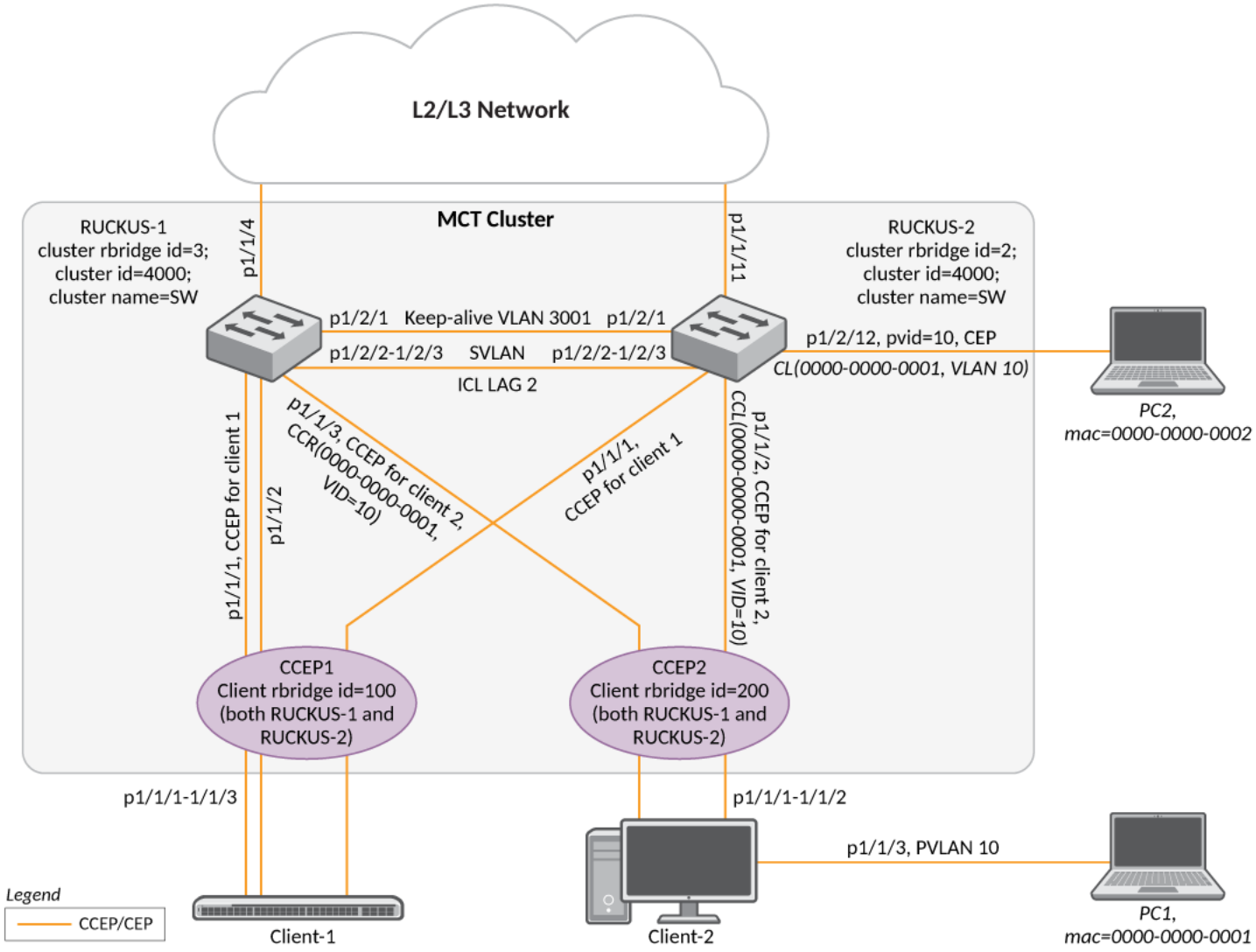
The **show cluster** command output displays a message to explain when the CCP is down because the board type does not match with the peer device as shown in the following example.

```
 Last Reason for CCP Down: Image version or board type doesn't match with peer
 Peer State: CCP Down (CCP is Down)
 Reason for CCP Down: Image version or board type doesn't match with
peer
```

Basic MCT Configuration

This section describes how to set up a basic MCT configuration. The following figure shows a basic MCT topology, which applies to Layer 2 and Layer 3. MCT can also be supported with VRRP or VRRP-E.

FIGURE 33 Basic MCT Configuration



MCT Configuration Considerations

- Configuring flow-based MAC address learning and MCT on the same device is not supported.
- For MCT, Layer 2 protocols such as STP and RSTP are not supported on cluster VLANs and session VLAN. As a result, STP must be disabled explicitly on cluster VLANs and session VLAN. STP is automatically disabled in the router image.
- One ICL can be configured per device or stack, and a device or stack can be in only one cluster.
- An ICL port must not be an untagged member of any VLAN.
- It is recommended that you set up ICL as a static LAG with at least two ports. This provides port-level redundancy and higher bandwidth for cluster communication.
- ICL ports must be part of MCT VLANs and session VLANs.
- An ICL cannot be a regular port link or an LACP trunk. It must be a single- or multiple-port static LAG.
- MDUP synchronizes all MAC entries for VLANs served by an ICL link.

- In any MCT configuration, there are two different cluster-related IDs, the Cluster ID and the Cluster RBridge ID. The Cluster ID uniquely identifies a cluster. All cluster devices in the same MCT cluster must have the same Cluster ID. The Cluster RBridge ID uniquely identifies a cluster device within the cluster. To avoid conflicts, ensure that the Cluster ID and the Cluster RBridge IDs are unique within an MCT configuration and cannot be confused with each other.
- The Cluster RBridge ID must not conflict with any Client RBridge ID or with the peer RBridge ID.
- The Client RBridge ID is unique and must be the same on cluster devices.
- RUCKUS recommends keeping only ICL ports in the session VLAN during operation.
- MCT can support up to 16 ports per LAG, depending on the device.
- An ICL interface cannot be configured as the CCEP in any client.
- BPDU guard and root guard configurations must be identical on both cluster devices.
- RUCKUS recommends that you configure a keep-alive VLAN as a separate link (not ICL). The keep-alive VLAN provides a backup control path when CCP goes down.
- From the perspective of MCT, a cluster can consist of a pair of standalones or a pair of stacks. RUCKUS does not recommend pairing a standalone and a stack in the same cluster.
- ICX 7550 and ICX 7850 stacks can be configured as a cluster. For other ICX platform families, standalone devices (meaning, not stacked) can be configured as a cluster.
- All devices in an MCT cluster must be from the same platform family (for example, all ICX 7850 devices).
- From FastIron release 09.0.10a, the maximum number of LACP LAGs that can be configured for MCT cluster increases from 256 to 300.
- Layer 2 multicast traffic may drop for a longer time (approximately 2 minutes and 27 seconds) after an active controller unit is added to an MCT peer stack.
- In an MCT configuration with stacking rings, after a standby controller goes down and comes back up, traffic loss may occur, including on ports from stack units other than the standby controller.
- In an MCT configuration with ICX 7550 or ICX 7850 peer stacks that include a static LAG connection through a copper GBIC, traffic loss of up to 12 seconds may occur when any unit rejoins the stack following a failover or unit reload.
- In a scaled MCT configuration, Layer 2 traffic spikes may occur when one of the MCT peers rejoins after a reload.
- When a core link flap or peer reload occurs and uplink traffic is being received via a cluster, a significant delay may occur before CR MAC entries are changed to CL MAC entries.

MCT Recommendations and Best Practices

The following configurations and settings are recommended for MCT.

- MCT clients must not be physically connected with each other.
- There should be only one MCT client connection to the MCT cluster from each access switch closet.
- Ensure that there is no device connectivity to MCT cluster on CEP ports.
- xSTP must not be configured on MCT VLAN.
- Enable RSTP only on MCT Cluster VLANs in deployments that have only CCEP clients.
- Do not enable RSTP on an MCT cluster in deployments that include both CEP and CCEP.
- A cluster must not be undeployed or removed using the **no deploy** command (under cluster) or **no cluster** command while traffic is running, as it may cause traffic disruption. These commands should be executed in the maintenance window.
- Configure cluster peers in loose mode with keepalive.
- Configure the **no spanning-tree** command under LAG interface on MCT clients connected to the cluster.
- Set the hello timer for VRRP-E to 20 seconds.

- If a device is connected to both cluster nodes on CEP ports making it a triangle topology, set **icl-fwd-delay** to 10 seconds under cluster.
- Configure the **ccep-up-delay** setting to 300 seconds under cluster.
- Configure the VE delay notification setting to 15 seconds.
- Layer 2 multicast traffic may drop for a longer time (approximately 2 minutes and 27 seconds) after an active controller unit is added to an MCT peer stack.
- In an MCT configuration with stacking rings, after a standby controller goes down and comes back up, traffic loss may occur, including on ports from stack units other than the standby controller.
- In an MCT configuration with ICX 7550 or ICX 7850 peer stacks that include a static LAG connection through a copper GBIC, traffic loss of up to 12 seconds may occur when any unit rejoins the stack following a failover or unit reload.
- In a scaled MCT configuration, Layer 2 traffic spikes may occur when one of the MCT peers rejoins after a reload.
- When a core link flap or peer reload occurs and uplink traffic is being received via a cluster, a significant delay may occur before CR MAC entries are changed to CL MAC entries.

Configuring MCT

This section provides basic configuration steps, which should be completed in the specified order.

Step 1: Configure ICL for Cluster Devices and LAGs for Client Devices on page 100

Step 2: Configure the MCT VLAN, MCT Session VLAN, Recommended MCT keep-alive VLAN, and ICL Forwarding Delay on page 101

Step 3: Configuring the Cluster on page 102

Step 4: Configuring Clients on page 103

After completing these steps, you can verify the configuration by running the **show cluster** command. Refer to Displaying MCT Information on page 147.

Step 1: Configure ICL for Cluster Devices and LAGs for Client Devices

You can configure static or dynamic LAGs for cluster clients. Static LAGs are manually configured aggregate links containing multiple ports. Dynamic LAGs use Link Aggregation Control Protocol (LACP) to maintain aggregate links over multiple ports. LACP PDUs are exchanged between ports on each device to determine if the connection is still active. The LAG then shuts down any port for which the connection is no longer active.

If you are configuring two stacks as cluster peer devices, the ICL LAG must include a port on the stack active controller and a port on the stack standby controller.

NOTE
For details on stack configuration, refer to the *RUCKUS FastIron Stacking Configuration Guide*.

NOTE
ICL LAGs support only static ports.

To configure an ICL static LAG for RUCKUS-1 device in the topology of Basic MCT Configuration on page 97, enter the following commands.

1. Enter the Global Configuration mode.

```
RUCKUS-1# Configure terminal
```

2. Configure a lag name and static ID.

```
RUCKUS-1(config)# lag MCT_lag static id 2
```

3. Add the ICL ports to the static lag.

```
RUCKUS-1(config-lag-MCT_lag)# ports ethernet 1/2/2 to 1/2/3
```

To configure an ICL static LAG for RUCKUS-2 device, enter the following commands.

```
RUCKUS-2(config)# lag MCT_lag static id 2
RUCKUS-2(config-lag-MCT_lag)# ports ethernet 1/2/2 to 1/2/3
```

To configure a dynamic LAG for client-1 on RUCKUS-1 device, enter the following commands.

```
RUCKUS-1(config)# lag client1_lag dynamic id 5
RUCKUS-1(config-lag-client1_lag)# ports ethernet 1/1/1 to 1/1/3
```

To configure a dynamic LAG for client-2 on RUCKUS-1 device, enter the following commands.

```
RUCKUS-1(config)# lag client2_lag dynamic id 6
RUCKUS-1(config-lag-client2_lag)# ports ethernet 1/1/3
```

To configure a dynamic LAG for client-1 on RUCKUS-2 device, enter the following commands.

```
RUCKUS-2(config)# lag client1_lag dynamic id 5
RUCKUS-2(config-lag-client1_lag)# ports ethernet 1/1/1
```

To configure a dynamic LAG for client-2 on RUCKUS-2 device, enter the following commands.

```
RUCKUS-2(config)# lag client2_lag dynamic id 6
RUCKUS-2(config-lag-client2_lag)# ports ethernet 1/1/2
```

Step 2: Configure the MCT VLAN, MCT Session VLAN, Recommended MCT keep-alive VLAN, and ICL Forwarding Delay

To create the recommended MCT keep-alive VLAN and MCT session VLAN for Ruckus-1and Ruckus-2 devices in the topology of Basic MCT Configuration on page 97, enter the following commands.

1. Create an MCT keep-alive VLAN by adding the tagged ports to the VLAN.

```
RUCKUS-1# configure terminal
RUCKUS-1(config)# vlan 3001 name MCT-keep-alive
RUCKUS-1(config-vlan-3001)# tagged ethernet 1/2/1
RUCKUS-1(config-vlan-3001)# exit

Ruckus-2# configure terminal
RUCKUS-2(config)# vlan 3001 name MCT-keep-alive
RUCKUS-2(config-vlan-3001)# tagged ethernet 1/2/1
RUCKUS-2(config-vlan-3001)# exit
```

2. Create a session VLAN by adding the ICL static lag ID to the VLAN, and disable the spanning tree protocol.

```
RUCKUS-1(config)# vlan 3000 name Session-VLAN
RUCKUS-1(config-vlan-3000)# tagged lag 2
RUCKUS-1(config-vlan-3000)# no spanning-tree

RUCKUS-2(config)# vlan 3000 name Session-VLAN
RUCKUS-2(config-vlan-3000)# tagged lag 2
RUCKUS-2(config-vlan-3000)# no spanning-tree
```

3. Create a router interface with session VLAN ID, and assign an IP address to the router interface.

```
RUCKUS-1(config)# interface ve 3000
RUCKUS-1(config-vif-3000)# ip address 10.1.1.3/24

RUCKUS-2(config)#interface ve 3000
RUCKUS-2(config-vif-3000)# ip address 10.1.1.2/24
```

4. (Optional) Configure a delay in seconds to wait after the ICL link comes up before regular traffic can be transmitted. During the delay, only BPDU traffic is forwarded. The default is 0. The valid range is 0 through 30.

```
RUCKUS-1(config)# cluster mct_stack 1000 <-creates cluster
                                         enters cluster
                                         config mode
RUCKUS-1(config-cluster-mct_stack)# icl-fwd-delay 3
RUCKUS-1(config-cluster-mct_stack)# exit
RUCKUS-1(config)#

RUCKUS-2(config)# cluster mct_stack 1000
RUCKUS-2(config-cluster-mct_stack)# icl-fwd-delay 3
RUCKUS-2(config-cluster-mct_stack)# exit
RUCKUS-2(config)#
```

5. (Optional) Configure a delay in seconds to keep dynamic CCEP LAGs on a CCP peer in blocking state while CCP comes up, for example, following a cluster-peer reload, ICL flap, or cluster re-deployment or undeployment.

```
RUCKUS-1(config)# cluster mct_stack 1000
RUCKUS-1(config-cluster-mct_stack)# ccep-up-delay 300
RUCKUS-1(config-cluster-mct_stack)# exit
RUCKUS-1(config)#

RUCKUS-2(config)# cluster mct_stack 1000
RUCKUS-2(config-cluster-mct_stack)# ccep-up-delay 300
RUCKUS-2(config-cluster-mct_stack)# exit
RUCKUS-2(config)#
```

NOTE
The delay can be set to a value from 0 through 600 seconds. The recommended setting is 300. The default setting is 0, which disables the timer.

To implicitly configure the MCT VLAN and add the ICL as a tagged member of the VLAN, enter the following commands.

```
RUCKUS-1(config)# vlan 1000 name MCT-VLAN-example
RUCKUS-1(config-vlan-1000)# tagged ethernet 1/1/1 to 1/1/3 lag 2

RUCKUS-2(config)# vlan 1000 name MCT-VLAN-example
RUCKUS-2(config-vlan-1000)# tagged ether 1/1/1 to 1/1/2 lag 2
```

Step 3: Configuring the Cluster

Configuring a local cluster requires the cluster ID and RBridge ID for the local switch or router.

Configuration of the peer device involves the peer's IP address, RBridge ID, and ICL specification.

The following task steps configure device RUCKUS-1 for the cluster in the topology of Basic MCT Configuration on page 97. The peer device, RUCKUS-2 configuration is shown in the Example section below the task.

1. Enter global configuration mode.

```
RUCKUS-1# configure terminal
```

2. Configure a cluster name and ID.

```
RUCKUS-1(config)# cluster SW 4000
```

If you do not specify a cluster name, the device will automatically generate the cluster name as CLUSTER-X.

3. Configure the RBridge ID.

```
Ruckus-1(config-cluster-SW)# rbridge-id 3
```

The RBridge ID must be different from the other cluster RBridge and any other client in the cluster.

4. Assign a VLAN to the session.

```
RUCKUS-1(config-cluster-SW)# session-vlan 3000
```

   The MCT member VLAN is defined as any VLAN of which the ICL is a member.

5. Configure the keep alive for the VLAN.

```
RUCKUS-1(config-cluster-SW)# keep-alive-vlan 3001
```

6. Configure the ICL.

```
Ruckus-1(config-cluster-SW)# icl ICL lag 2
```

7. Configure the peer ICL parameters.

```
RUCKUS-1(config-cluster-SW)# peer 10.1.1.2 rbridge-id 2 icl ICL
```

8. Deploy the configuration.

```
RUCKUS-1(config-cluster-SW)# deploy
```

The following example configures device Ruckus-2 for the cluster in the topology of Basic MCT Configuration on page 97.

```
RUCKUS-2# configure terminal
RUCKUS-2(config)# cluster SW 4000
RUCKUS-2(config-cluster-SW)# rbridge-id 2
RUCKUS-2(config-cluster-SW)# session-vlan 3000
RUCKUS-2(config-cluster-SW)# keep-alive-vlan 3001
RUCKUS-2(config-cluster-SW)# icl ICL lag 2
RUCKUS-2(config-cluster-SW)# peer 10.1.1.3 rbridge-id 3 icl ICL
RUCKUS-2(config-cluster-SW)# deploy
```

Proceed to the next task.

Step 4: Configuring Clients

Configuring cluster clients requires the client name, RBridge ID, and CCEP.

This task describes how to configure clients manually. For instructions on automatic client configuration, see Setting Up Cluster Client Automatic Configuration on page 106.

In the network shown in the Basic MCT Configuration on page 97, Client-1 has a three-port LACP trunk (1/1/1-1/1/3), while Client-2 has a two-port static trunk (1/1/1-1/1/2) towards the MCT cluster.

To configure Client-2 on Ruckus-1 in the topology of Basic MCT Configuration on page 97, enter the following steps.

1. Enter global configuration mode.

```
RUCKUS-1# configure terminal
```

2. Configure a cluster name and ID.

```
RUCKUS-1(config)# cluster SW 4000
```

3. Assign a client name.

```
RUCKUS-1(config-cluster-SW)# client client-2
```

   The client name can be different on the different cluster devices.

4. Configure the RBridge ID.

```
RUCKUS-1(config-cluster-SW-client-2)# rbridge-id 200
```

   The RBridge ID must be different from the cluster RBridge ID or the ID of any other client in the cluster.

5. Configure the dynamic lag as the client CCEP.

```
RUCKUS-1(config-cluster-SW-client-2)# client-interface lag 6
```

6. Deploy the configuration.

```
RUCKUS-1(config-cluster-SW-client-2)# deploy
```

The following example shows how to configure Client-2 on RUCKUS-2 in the topology of Basic MCT Configuration on page 97.

```
RUCKUS-2# configure terminal
RUCKUS-2(config)# cluster SW 4000
RUCKUS-2(config-cluster-SW)# client client-2
RUCKUS-2(config-cluster-SW-client-2)# rbridge-id 200
RUCKUS-2(config-cluster-SW-client-2)# client-interface lag 6
RUCKUS-2(config-cluster-SW-client-2)# deploy
```

The following example shows how to configure Client-1 on RUCKUS-1 in the topology of Basic MCT Configuration on page 97.

```
RUCKUS-1# configure terminal
RUCKUS-1(config)# cluster SW 4000
RUCKUS-1(config-cluster-SW)# client client-1
RUCKUS-1(config-cluster-SW-client-1)# rbridge-id 300
RUCKUS-1(config-cluster-SW-client-1)# client-interface lag 5
RUCKUS-1(config-cluster-SW-client-1)# deploy
```

The following example shows how to configure Client-1 on RUCKUS-2 in the topology of Basic MCT Configuration on page 97.

```
RUCKUS-2# configure terminal
RUCKUS-2(config)# cluster SW 4000
RUCKUS-2(config-cluster-SW)# client client-1
RUCKUS-2(config-cluster-SW-client-1)# rbridge-id 300
RUCKUS-2(config-cluster-SW-client-1)# client-interface lag 5
RUCKUS-2(config-cluster-SW-client-1)# deploy
```

Forcing a Port Up in a Basic MCT Configuration

In a static trunk environment, Preboot eXecution Environment (PXE) images are too small for most operating systems to leverage LACP during the boot process. As a result, during a PXE build process, traffic sent by the server is dropped, and the build process can fail.

To correct this situation, a port on the supported ICX device connected to a server that is configured as an MCT client can be set to a “force-up” state so that even if the LACPDU is not received from the server, the connected port is up and forwards packets.

NOTE
When multiple ports from the same server are connected to the ICX device, the port on the ICX device connected to the PXE-capable port on the server is the port that must be configured to the force-up state. The PXE-capable port varies from server to server.

Keep the following points in mind when configuring a port to a force-up state:

- A port can only be configured as the force-up port before the client is deployed.
- Only one port in an LACP link aggregation group (LAG) can be configured as the force-up port. If you configure multiple ports as force-up, the following error message is displayed: `Error: port portno is already configured as force-up port.`
- When a port is configured for force-up and the server boots for the first time, the port does not wait for any LACPDU but immediately begins to forward packets.
- If the port receives an LACPDU, it bundles with other ports and forms a LAG. The server is operational.

- If the force-up port goes down while in a LAG, the port continues to perform as a normal LACP trunk, and the server remains operational, with some ports down.
- If the force-up port stops receiving LACPDUs, the port ignores the time-out and remains operational.

To configure the LACP client in a force-up state, use the **client-interface link-aggregation force-up ethernet** command at the client level.

The following example shows output from the **show lag** command after the link aggregation information for a port has been configured in a force-up state.

```
Router# show lag id 163

Total number of LAGs:          11
Total number of deployed LAGs: 11
Total number of trunks created:11 (113 available)
LACP System Priority / ID:     1 / 748e.f88f.2222
LACP Long timeout:             90, default: 90
LACP Short timeout:            3, default: 3

=== LAG "CCEP-163" ID 163 (dynamic Deployed) ===
LAG Configuration:
   Ports:         e 1/1/47 to 1/1/48
   Port Count:    2
   LAG Interface: lg163
   Trunk Type:    hash-based
   LACP Key:      20163
Deployment: HW Trunk ID 3
This is a Multi Chassis Trunk: (System Id: 0180.c200.0001, Key: 30163)

Port    Link    State   Dupl Speed Trunk Tag Pvid Pri MAC             Name
1/1/47  Up      Forward Full 1G    163   Yes N/A  0   748e.f88f.2222
1/1/48  Down    None    None None  163   Yes N/A  0   748e.f88f.2222

Port   [Sys P] [Port P] [ Key ] [Act][Tio][Agg][Syn][Col][Dis][Def][Exp][Ope]
1/1/47       1        1   20163   Yes   L   Agg  Syn  Col  Dis  Def  No   Frc
1/1/48       1        1   20163   Yes   L   Agg  Syn  No   No   Def  No   Dwn


 Partner Info and PDU Statistics
Port       Partner         Partner     LACP      LACP
          System MAC         Key     Rx Count  Tx Count
1/1/47    0000.0000.0000       46     5475      5558
1/1/48    0000.0000.0000       47     5477      5487
```

Cluster Client Automatic Configuration

Client configuration includes setting the client name, client RBridge ID (unique identification for each client), client interface (CCEP), and deployment settings on both MCT cluster devices. With up to 150 clients per cluster, manual configuration can take a considerable amount of time.

Cluster client automatic configuration saves the time that would be required to complete the entire configuration manually.

The following limitations apply to cluster client automatic configuration:

- Cluster client automatic configuration is designed for generating new clients, not for updating an existing client.
- A single client span across multiple devices is not supported (cascading MCT). For example, the configuration of cascading MCT through cluster client automatic configuration is not supported.
- Multiple clients on the same device are not supported.
- When hash collisions occur (RBridge ID collisions), cluster client automatic configuration reports errors, and manual intervention is required.
- An MCT cluster will auto-configure static CCEP LAGs when any third-party switch used as an MCT client is not capable of sending LLDP TLVs.

- When the MCT client-auto-config feature is used, only one MCT client can be formed per standalone or stack.

For cluster client automatic configuration to work, the following prerequisites are required on the cluster side:

- The cluster must be configured on both MCT cluster devices.
- An MCT VLAN must be configured on both MCT cluster devices.
- The trunk group configuration must be removed from the client interfaces.
- The client interfaces must be up and operational.
- The cluster ID must be unique when there are multiple clusters interconnected in a topology. For example, in a cascaded stage 2 MCT cluster, the cluster ID on a stage 1 pair of switches must be different from the cluster ID on a stage 2 pair of switches.

The following prerequisites are required on the client side:

- VLAN and trunk group configuration must be completed.
- Link Level Discovery Protocol (LLDP) must be enabled.

Setting Up Cluster Client Automatic Configuration

Complete the following steps to configure cluster client automatic configuration.

1. Configure a session VLAN and router interfaces on session VLAN for both the devices.
2. Configure a keep-alive VLAN on both the devices.
3. Configure a lag interface and MCT VLAN for the client(switch only).
4. Configure the MCT cluster between the MCT peers.
5. Enable an LLDP on MCT cluster and client devices for auto-configuration.

```
RUCKUS-1(config-cluster-SW)# lldp run
RUCKUS-1(config-cluster-SW)# lldp enable ports ethernet 1/1/1 to 1/1/3
```

6. Enable the client auto-detect ports on both MCT devices.

```
RUCKUS-1(config-cluster-SW)# client-auto-detect ethernet 1/1/1 to 1/1/3
```

   In the port list, specify all the CCEPs for all potential clients.

7. Start the client auto-detect process on both cluster devices.

```
RUCKUS-1(config-cluster-SW)# client-auto-detect start
```

   Within one minute, the system reports information and errors (if there are mismatches such as an LACP configuration mismatch). You can fix the mismatch while the process is running. Use the **config-deploy-all** option with this command as an alternative to the **client-auto-detect config** command. The **client-auto-detect config** command also configures automatically detected clients into the running configuration and deploys all of the automatically detected clients.

8. Check and fix the automatically detected clients.

```
RUCKUS-1(config-cluster-SW)# show cluster cluster-SW client-auto-detect

cluster cluster-SW 4000
 rbridge-id 3
 session-vlan 3000
 icl ICL lag 2
peer 10.1.1.2 rbridge-id 2 icl ICL
client-auto-config ethe 1/1/1 to 1/1/3
 client-auto-config start
deploy
 client AUTO-Router002438769e00
  rbridge-id 3593
  client-interface lag 5
!
```

NOTE
At this point, the client configuration does not appear in the running configuration and cannot be modified. Static trunk and LACP configurations are not effective yet.

9. Configure automatically detected clients into the running configuration.

```
RUCKUS-1(config-cluster-ICX)# client-auto-detect start config-deploy-all
```

All automatically configured client information is now published into the running configuration, and the static trunk configuration is generated, created, and deployed. LACP is started. By default, clients are in the non-deployed state and the CCEP is put into the disabled state. Ports that are successfully programmed as CCEPs are removed from the auto-configuration-enabled port list. If the port list is empty, which means all ports are configured into clients successfully, the automatic configuration process stops. The original LLDP configuration is restored. Otherwise, the automatic configuration process continues only on the ports still left in the list.

MCT Failover Scenarios

The following scenarios describe what happens if specific elements in the MCT configuration fail:

- Client interface on one of the MCT cluster devices goes down.
  - Traffic switches to the other cluster device with minimal traffic loss.
- MCT cluster device goes down.
  - When an MCT cluster device goes down (for example, due to a power failure), the traffic fails over to the other MCT cluster device.
  - When an ICX stack is configured as a MCT cluster and a stack member goes down, traffic converges again within 10 to 30 seconds.

    NOTE
    Hitless failover within an ICX stack is enabled by default. Hitless failover and how it is configured for an MCT system is described separately in the following points.

- Hitless failover occurs.
  - The MCT CCEPs stay up during hitless switchover, failover, or upgrade. Link protocols such as UDLD and LACP on CCEPs do not flap. Traffic disruption is minimal (sub-second). The MCT CCP connection flaps once, and MAC is re-synced between the peer devices.
  - The CCP goes down and comes back up again once the hitless failover is completed.
- ICL interface or CCP goes down (keep-alive VLAN is configured).
  - If a keep-alive VLAN is used, the devices in the cluster can communicate even if the ICL goes down. If the peer device is reachable over the keep-alive VLAN, the MCT peers perform the active/standby negotiation per client. After negotiation, the standby shuts down its client ports, and the active client ports continue to forward the traffic.

  - The active/standby negotiation is performed per MCT client on the basis of RBridge ID and client local or remote accessibility. If the client is reachable from both MCT devices, the lower RBridge ID becomes the active device. If the client can be accessed only from one of the MCT devices, the cluster device on which it is reachable becomes the active device.
  - If the peer device cannot be reached over the keep-alive VLAN, then both cluster devices keep forwarding.

  NOTE
  RUCKUS recommends using keep-alive VLANs with the MCT configurations. This provides alternative access if the ICL interface goes down. However, a keep-alive VLAN must not be configured when **bpdu-flood-enable** is configured. Refer to MCT Layer 2 Protocols on page 113.

- ICL interface or CCP goes down (keep-alive VLAN is not configured).
  - When the keep-alive VLAN is not configured, both cluster devices keep forwarding. Use the **client-isolation strict** command to disable the client interface as soon as the ICL link goes down to completely isolate the client.
- Double failures occur (for example, the ICL goes down and the client interface goes down on one of the MCT cluster devices).
  - Multiple failures could cause traffic to drop, even if there is a physical path available.

Cluster Failover Mode

The following failover modes can be configured with MCT:

- Fast-failover (default): As soon as the ICL interface goes down, the CCP goes down. All the remote MAC addresses are flushed.
- Slow-failover: Even if the ICL interface goes down, the CCP waits for the hold time before taking the CCP down. Remote MAC addresses are flushed only when the CCP is down.

Configuring the Failover Mode

By default, fast-failover is enabled on the device.

NOTE
You must configure the same Failover Mode setting on both cluster devices.

To change the failover mode, enter the following commands.

1. Enter the Global Configuration mode.

```
RUCKUS-1# configure terminal
```

2. Configure a cluster name and ID.

```
RUCKUS-1(config)# cluster SW 4000
```

   If you do not specify a cluster name, the device will automatically generate the cluster name as CLUSTER-X.

3. Change the default failover mode (fast-failover).

```
RUCKUS-1(config-cluster-SX)# peer 10.1.1.3 disable-fast-failover
```

Client Isolation Mode

NOTE
You must create the same isolation mode on both cluster devices. The CLI allows modification of the client isolation mode on MCT cluster devices.

MCT cluster devices can operate in two modes. Both peer devices must be configured in the same mode.

Loose mode (default): When the CCP goes down, the peer device performs the active/standby negotiation. After negotiation, the standby shuts down its peer ports, but the active peer ports continue to forward traffic if a keep-alive VLAN is configured. If a keep-alive VLAN is not configured, both peer devices become active, and both of the client ports stay up.

Strict mode: When the CCP goes down, the interfaces on both the cluster devices are administratively shut down. In this mode, the client is completely isolated from the network if the CCP is not operational.

Configuring Client Isolation Mode

To isolate the client from the network when Cluster Communication Protocol (CCP) is not operational, enter the following commands.

1. Enter the Global Configuration mode.

```
RUCKUS-1# configure terminal
```

2. Configure a cluster name and ID.

```
RUCKUS-1(config)# cluster SW 4000
```

   If you do not specify a cluster name, the device will automatically generate the cluster name as CLUSTER-X.

3. Change the isolation mode to strict mode.

```
RUCKUS-1(config-cluster-SX)# client-isolation strict
```

   NOTE
   The CLI allows modification of the client isolation mode on MCT cluster devices. You must create the same isolation mode on both cluster devices.

Shutting Down All Client Interfaces

The client interfaces in the cluster can be shut down when performing a hitless upgrade operation. This results in failover of traffic to the peer device.

To shut down all the client interfaces in the cluster, enter the following commands.

1. Enter the Global Configuration mode.

```
RUCKUS-1# configure terminal
```

2. Configure a cluster name and ID.

```
RUCKUS-1(config)# cluster SW 4000
```

   If you do not specify a cluster name, the device will automatically generate the cluster name as CLUSTER-X.

3. Shut down all the client interfaces in the cluster.

```
RUCKUS-1(config-cluster-SX)# client-interfaces shutdown
```

Using the Keep-Alive VLAN

CCRR messages are used to exchange information between peer devices. When the CCP is up, CCRR messages are sent over the CCP. When the CCP client cannot be reached or the ICL is down, you can use the **keep-alive-vlan** command under the cluster context so CCRR messages are periodically sent over the keep-alive VLAN. Only one VLAN can be configured as a keep-alive VLAN. The keep-alive VLAN cannot be a member VLAN of the MCT, and this VLAN can be tagged or untagged.

NOTE
When a keep-alive VLAN is configured, client isolation mode cannot be configured as strict.

```
device(config-cluster-ICX))# keep-alive-vlan 10
```

When the CCP is down, the following results occur.

- If the keep-alive VLAN is configured, CCRR messages are sent every second over that VLAN.
- When CCP is down and a keep-alive VLAN is configured, active/standby selection is based on the following criteria:
  - If one device's CCEPs are up and the peer's CCEPs are down, the peer with the local CCEPs down becomes the standby.
  - Otherwise, the device with the higher RBridge ID becomes the standby.
- If no packets are received from the peer device for a period of three seconds, the peer is considered down.
- If a keep-alive VLAN is not configured and both the peer devices are up, both peers keep forwarding traffic independently.

Setting Keep-Alive Timers and Hold Time

NOTE
The keep-alive time and hold time must be the same on both cluster peers.

To specify the keep-alive timers and hold time for the peer devices, enter the following commands.

1. Enter the Global Configuration mode.

   ```
   RUCKUS-1# configure terminal
   ```

2. Configure a cluster name and ID.

   ```
   RUCKUS-1(config)# cluster SW 4000
   ```

3. Specify the keep-alive timers and hold time for the peer device.

   ```
   RUCKUS-1(config-cluster-SX)# peer 10.1.1.3 timers keep-alive 40 hold-time 120
   ```

   When using this command ensure that the peer IP address must be in the same subnet as the cluster management interface.

   NOTE
   The keep-alive VLAN and keep-alive timers are not related. The keep-alive timer is used by CCP.

MCT Hitless Upgrade

NOTE
MCT hitless upgrade is not supported for upgrade from a FastIron releases prior to 09.0.10a.

ICX devices support MCT hitless upgrade with minimal traffic disruption for MCT cluster devices. In-Service System Upgrade (ISSU) between minor releases is used for image upgrade of stacking units within the same cluster node. Stacking units within the cluster node should be upgraded one after the other to minimize traffic disruption.

Refer to the *RUCKUS FastIron Software Upgrade Guide* for more information on ISSU and upgrading a stack.

Layer 2 Behavior with MCT

Layer 2 behavior when MCT is configured includes MAC operations, dynamic trunks, port loop detection, and multicast snooping over MCT.

MAC Database Update

Each MAC address is advertised with a cost. Low-cost MAC addresses are given preference over high-cost addresses. MAC addresses that are learned locally are given the highest priority, or the cost of 0, so that they are always selected as the best MAC addresses.

If a MAC address moves from a CCEP port to a CEP port, a MAC move message is sent to the peer, and the peer moves the MAC address from its CCEP ports to the ICL links.

If two MAC addresses have the same cost, the address learned from the lower RBridge ID wins and is installed in the forwarding database (FDB).

MAC addresses in MCT VLANs are updated across the cluster using MAC Database Update Protocol (MDUP) messages.

Cluster MAC Types

Cluster Local MAC (CL): MAC addresses that are learned on the MCT VLAN and on CEPs locally. MAC addresses are synchronized to the cluster peer device and are subject to aging.

Cluster Remote MAC (CR): MAC addresses that are learned by way of MDUP messages from the peer device (CL on the peer). The MAC addresses are always programmed on the ICL port and do not age. The CR is deleted only when the CL is deleted from the peer. An MDB entry is created for these MAC addresses with a cost of 1 and is associated with the peer RBridge ID.

Cluster Client Local MAC (CCL): MAC addresses that are learned on the MCT VLAN and on CCEPs. The MAC addresses are synchronized to the cluster peer device and are subject to aging. An MDB entry with a cost of 0 is created for these addresses, and they are associated with the client and cluster RBridge IDs.

Cluster Client Remote MAC (CCR): MAC addresses that are learned by way of MDUP messages from the peer device (CCL on the peer). The MAC addresses are always programmed on the corresponding CCEP and do not age. The CCR is deleted only when the CCL is deleted from the peer. An MDB entry with the cost of 1 is created for the MAC addresses, and they are associated with the client and peer RBridge IDs.

Cluster Multi-Destination Local MAC (CML): A static MAC entry that is configured locally on the MCT VLAN. Any static MAC address configured on the MCT VLAN will have the ICL added by default. Consequently, the address automatically becomes a multi-destination MAC entry. The local configuration generates a local MDB. Any CML entry can still have up to two associated MDBs, one local and one remote. The remote MDB contains the remote static configuration for the same MAC and VLAN. If the dynamic MAC and static configuration coexist, the dynamic MAC address is removed, whether it is learned locally or from MDUP. The port list of a CML entry contains an ICL port, the client ports from the client list in the local configuration and the remote configuration (if it exists), and all locally configured CEPs.

Cluster Multi-Destination Remote MAC (CMR): A static MAC entry that is configured on the MCT VLAN on the peer side and has no associated local configuration. The CMR entry has only the information from the remote MDB. The port list of a CMR entry contains an ICL port and all the client ports from the client list in the remote configuration. When there is a local configuration for the same entry, the CMR is converted to the CML.

MAC Aging

Only the local MAC entries are aged on a cluster device. The remote MAC address entries are aged based on explicit MDUP messages only.

The remote MAC addresses learned through MDUP messages are dynamic addresses, but they never age from the FDB.

MAC Flush

If the CEP is down, the MAC addresses are flushed, and individual MAC deletion messages are sent to the peer device.

If the local CCEP is down, the MAC addresses are flushed locally, and individual MAC deletion messages are sent to the peer device.

If the **clear mac-address** command is used, the MDB and FDB are rebuilt.

If the **clear mac-address vlan** command is used, the local MDB and FDB are rebuilt for the VLAN.

MAC movement happens normally on the local device.

In cases of CEP to CCEP MAC movement, the MAC movement occurs normally on the local device, and all the other MDBs from the peer are deleted to create a new local MDB.

Syncing Router MAC Addresses to Peer MCT Devices

The MCT cluster device uses a router MAC address to identify the packets that are addressed to the switch. Such packets may be received by a peer cluster device. The peer device switches packets over the ICL to the local MCT device to be routed properly.

Dynamic Trunks

The MCT client creates a single dynamic trunk group toward the MCT cluster devices. The dynamic trunk group consists of two trunk groups, each of which is configured on one of the MCT devices. A dynamic trunk group runs Link Aggregation Control Protocol (LACP).

For the two dynamic trunk groups of the MCT to behave as a single trunk group from the MCT client's perspective, both of the dynamic trunk groups must have the same LACP system ID and key (referred to as the MCT system ID and MCT key).

NOTE
The LAG IDs are only significant locally and need not match on the two ends of a LAG.
The LACP system ID in the MCT-supporting device normally comes from the port MAC address. To support LACP over MCT, the ID must be obtained in another way. MCT uses a pre-defined algorithm to obtain the ID.

NOTE
Each MCT cluster device has a unique cluster ID and one MCT client ID. The LACP key is predefined from the client ID and cluster ID. The user cannot change the key.
MCT control protocol synchronization is minimal. The LACP runs independently on the cluster devices.

Port Loop Detection

Loop detection can be used in an MCT topology to detect Layer 2 loops that occur due to misconfigurations, for example, on the client side when MCT links are not configured as trunk links on the MCT-unaware client.

In MCT, ICL links must be up at all times to prevent the cluster from going down. These links must not be shut down when a loop is detected in a network. Instead, other available ports (CCEPs) must be shut down. If loop detection BDPUs are received on the ICL port, instead of shutting down the ICL links, all CCEPs are error-disabled, and the user is notified with the following log message.

```
Loop-detection: Packet received on ICL port <port_number> for vlan <vlan_id>. Errdisable CCEPs.
```

Strict mode loop detection can be enabled on ICL ports. In strict mode, a port is disabled only if a packet is looped back to that same port. Strict mode overcomes specific hardware issues where packets are echoed back to the input port. This process assists in detecting hardware faults on ICL ports.

Loop detection can be enabled on MCT and non-MCT VLANs simultaneously. There is no change in loop detection behavior when it is enabled on non-MCT VLANs.

The following example shows how to configure loop detection on MCT and non-MCT VLANS.

```
device(config)# vlan 1905
device(config-vlan-1905)# loop-detection
device(config-vlan-1905)# end
```

MCT Layer 2 Protocols

Keep the following information in mind when configuring Layer 2 protocols with MCT:

- Metro Ring Protocol (MRP)—An ICL interface cannot be configured as an MRP secondary interface or vice versa because the ICL cannot be in a blocking state.

  MRP cannot be enabled on an MCT CCEP or vice versa.

Layer 2 Multicast Snooping over MCT

To support multicast snooping over MCT, the ICL port uses MDUP to synchronize the following information between the cluster devices:

- PIM-SM or PIM6-SM Join and Prune (control packets on MCT VLAN)

IGMP and MLD Snooping

Snooping can be configured globally or at the VLAN level. Each cluster device in the MCT VLAN can be configured as active or passive. There is no restriction for cluster devices to run active-active or passive-passive configurations.

The following examples show configuration commands for the VLAN level (IGMP and MLD), the global level (IGMP and MLD), and for PIM-SM and PIM6-SM.

NOTE
If the keepalive VLAN becomes unreachable when the ICL LAG is down, Layer 2 multicast traffic might be affected. If the peer that is the querier becomes the slave, router ports will not be learned on the clients.

- VLAN level (IGMP)

  ```
  device(config)# vlan 100
  device(config-vlan-100)# multicast active/passive
  ```

- VLAN level (MLD)

  ```
  device(config-vlan-100)# multicast6 active/passive
  ```

- Global level (IGMP and MLD)

  ```
  device(config)# ip multicast active/passive
  device(config)# ipv6 multicast active/passive
  ```

- PIM-SM snooping (configured only on a VLAN and requires IGMP snooping to run in passive mode)

```
device(config)# vlan 100
device(config-vlan-100)# multicast passive
device(config-vlan-100)# multicast pimsm-snooping
```

- PIM6-SM snooping (configured only on a VLAN and requires MLD snooping to run in passive mode)

```
device(config)# vlan 100
device(config-vlan-100)# multicast6 passive
device(config-vlan-100)# multicast6 pimsm-snooping
```

IGMP and MLD Snooping Behavior on MCT Cluster Devices

- Local information is synchronized to the MCT peer device using CCP. The information includes Mcache and FDB entry (on arrival of data traffic), joins and leaves, dynamic router ports, and PIM-SM snooping joins and prunes.
- Native control packets (joins and leaves) that are received are processed by protocol code and are forwarded if necessary.
- All control and data traffic is received on the ICL. The traffic is forwarded out of a CCEP only if the remote CCEP is down; otherwise, it is dropped by the egress filters on the CCEP.
- The ICL is added as an outgoing interface (OIF) by default whenever the CCEP is a source or a receiver. This provides faster convergence during MCT failover.
- For IGMP and MLD joins or leaves:
  - Only control packets received on a CCEP are synced to the MCT peer using CCP.
  - Control packets received on a CEP are not synced to the MCT peer.
- Static groups and static router ports configured on a CCEP are not synced across to the MCT peer. For these features to work correctly, they must be manually configured on the respective CCEPs of both the cluster nodes.

How Failovers are Handled for Layer 2 Multicast over MCT

The following failover scenarios may occur. Refer to MCT Failover Scenarios on page 107 for other types of failover scenarios.

- Local CCEP down event:
  - Outgoing traffic on local CCEP will now go through the ICL and out of the remote CCEP.
  - Incoming traffic on local CCEP will now ingress through the remote CCEP, and then ingress through the ICL locally.
- Local CCEP up event:
  - Outgoing traffic on a remote CCEP (after egressing through the local ICL) will now start going out of the local CCEP.
  - Incoming traffic from a client through the ICL (after ingressing on remote CCEP) will now switch back to the local CCEP (this is true only if the client trunk hashing sends the traffic toward the local CCEP).
- CCP (Cluster communication protocol) down event:
  - All related information (IGMP or MLD group, mcache, dynamic router port, PIM-SM snooping entry) that was synced from the peer device will now be marked for aging locally.
- CCP (Cluster communication protocol) up event:
  - All related information (IGMP or MLD group, mcache, dynamic router port, PIM-SM snooping entry) that was learned locally will be synced to the peer device.

PIM-SM and PIM6-SM Snooping over MCT

- PIM-SM snooping can be configured only on a VLAN. It requires IGMP snooping to be running in passive mode. IPv6 snooping is supported.

- PIM6-SM snooping can be configured only on a VLAN. It requires MLD snooping to be running in passive mode.
- Router ports can be configured on a VLAN or globally. They can be learned dynamically on the port where the query is received or configured statically.
- Both MCT devices must run PIM-SM snooping.
- PIM messages are forwarded by way of the hardware.
- A PIM join or prune is synced to the peer cluster device using CCP.
- A PIM prune is processed only if indicated by the peer cluster device.
- A PIM join or prune received natively on an ICL is ignored.
- A PIM hello is not synced but is received natively on the ICL.
- PIM port or source information is refreshed on both cluster devices by syncing PIM messages. The information ages out if not refreshed.

Forwarding Entries for PIM-SM and PIM6-SM Multicast Snooping

Table 9 and Table 10 list the forwarding entries for PIM-SM and PIM6-SM multicast snooping.

TABLE 9 Forwarding Entries (*,G) [a]

Event	MCT-1	MCT-2
No-Join	(*,G)->blackhole	(*,G)->blackhole
(S,G) Join on (MCT-1) CEP	(*,G)->CEP [s] [b]	(*,G)->ICL [s]
(S,G) Join on (MCT-2) CEP	(*,G)->ICL [s]	(*,G)->CEP [s]
(S,G) Join on (MCT-1) CCEP	(*,G)->CCEP [s], ICL [s]	(*,G)->CCEP [s], ICL [s]
(S,G) Join on (MCT-2) CCEP	(*,G)->CCEP[s], ICL [s]	(*,G)->CCEP [s], ICL [s]

a.) *ICL: The ICL port is added as default whenever CCEP is in OIF. The data traffic received from the ICL port will be filtered out by egress filters dynamically programmed on CCEPs.

b.) [s]: denotes sources maintained on port hash-list.

TABLE 10 Forwarding Entries (S,G) [a]

Event	MCT-1	MCT-2
No-Join	(S,G)->blackhole	(S,G)->blackhole
Join (MCT-1) CEP	(S,G)->CEP	(S,G)->ICL
Join (MCT-2) CEP	(S,G)->ICL	(S,G)->CEP
Join (MCT-1) CCEP	(S,G)->CCEP, ICL	(S,G)->CCEP, ICL
Join (MCT-2) CCEP	(S,G)->CCEP, ICL	(S,G)->CCEP, ICL

a.) *ICL: The ICL port is added as default whenever CCEP is in OIF. The data traffic received from the ICL port will be filtered out by egress filters dynamically programmed on CCEPs.

Displaying Information for Multicast Snooping

Use the **show ip pimsm-snooping cache** command to display (*,g), (s,g), and OIF information learned by way of PIM join and prune messages.

```
device(config)# show ip pimsm-snooping cache

 OIF Info:
 TR - OIF Belongs to Trunk/LAG, Primary port is displayed
 SG - (*,g)/(s,g) downstream fsm state:
     NI : No Info, J : Join, PP : Prune Pending, CLEAN : cleanup in progress
 RPT - (s,g,rpt) downstream fsm state:
```

```
      NI : No Info, P : Pruned, PP : Prune Pending, Px : Temp step in (*,G)
      join processing, PPx : Temp State in (*,G) processing, CLEAN : cleanup
      in progress.

PIMSM Snoop cache for vlan 503
1    (* 225.0.0.1) Up Time: 1d 19:41:48
    OIF: 1
    TR(e3/13) G : J(194) ET: 210, Up Time: 1d 19:41:48 , ICL, Remote

2    (* 225.1.1.1) Up Time: 5d 18:43:56
    OIFs: 2
    TR(e3/10) G : J(167) ET: 210, Up Time: 5d 18:43:56 , CCEP, Local
    TR(e3/13) G : J(200) ET: 210, Up Time: 1d 19:41:48 , ICL, Remote
```

You can also use the **show ip pimsm-snooping cache** command to display the MCT information if the VLAN is an MCT member.

In the following example, the **show ip multicast cluster** is used. YES indicates that reports or leaves were received locally (processing native control packets).

```
device(config)# show ip multicast cluster group

p-:physical, ST:static, QR:querier, EX:exclude, IN:include, Y:yes, N:no
VL100 : 1 groups, 1 group-port
group p-port          ST   QR life mode source local
1     225.1.1.1  e5/5 no   no  200   EX      0    YES
2     225.1.1.1 e5/10 no   no  200   EX      0    YES
```

In the following example, NO indicates that reports or leaves were received remotely. In this case, a join was received on the CCEP of the MCT peer device. Native control packets were processed by the peer device, and then the entries were synced over MDUP to this cluster device.

```
device(config)# show ip multicast cluster group

p-:physical, ST:static, QR:querier, EX:exclude, IN:include, Y:yes, N:no
VL100 : 1 groups, 1 group-port
group  p-port          ST QR life mode source local
1      225.1.1.1 e1/10 no no  200   EX      0    NO
2      225.1.1.1 e1/10 no no  200   EX      0    NO
```

The following example displays status about the IGMP router port.

```
device(config)# show ip multicast cluster vlan 100

Version=2, Intervals: Query=125, Group Age=260, Max Resp=10, Other Qr=260
VL100: cfg V3, vlan cfg passive, 1 grp, 2 (SG) cache, rtr ports,
router ports: e5/9(260) 100.100.100.1 (local:1, mct peer:0),
e5/4    has    1 groups,
This interface is non-Querier (passive)
default V3 trunk
(local:1, mct peer:0)
```

Use the **show ip multicast cluster pimsm-snooping** command to display detailed information about OIFs added by way of a PIM-SM snooping.

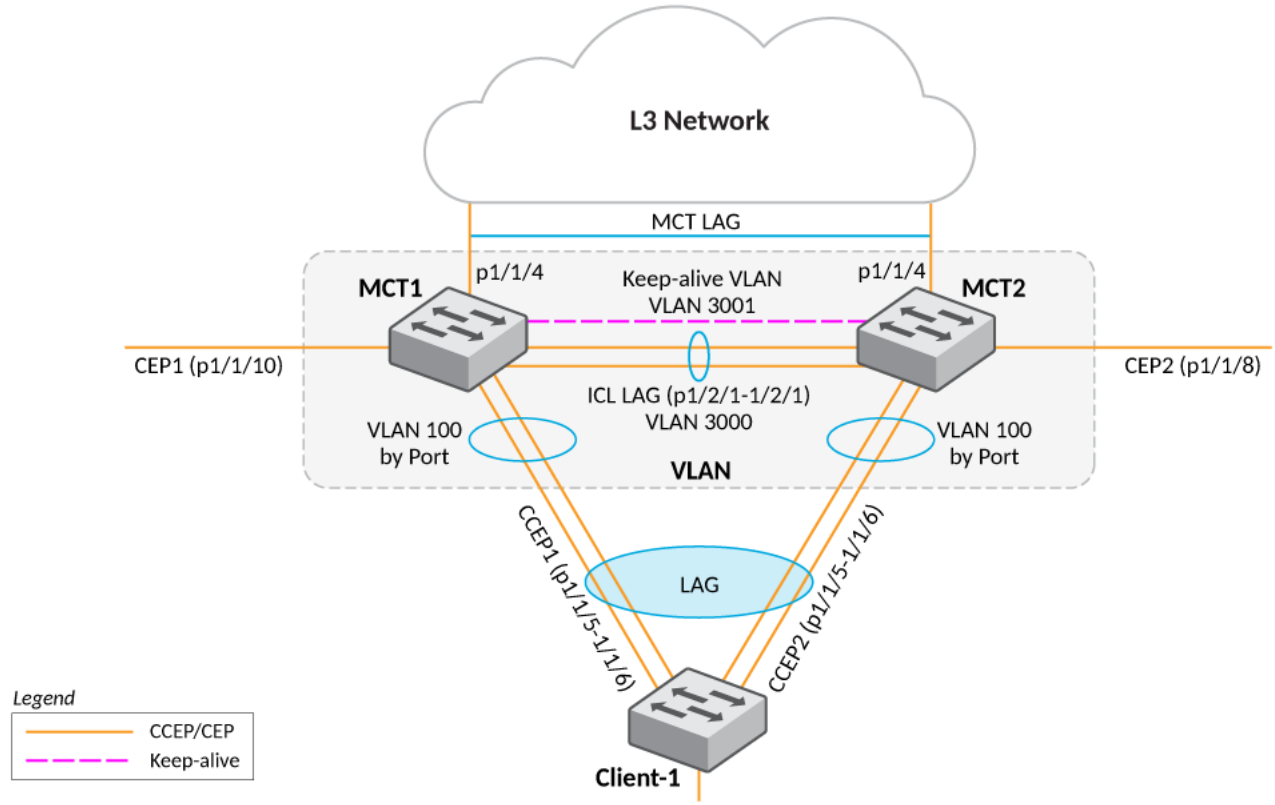
```
device(config)# show ip multicast cluster pimsm-snooping

Example: Port: 1/7/3 (age, port type, ref_count, owner flag, pruned flag)
source: 1/7/3 has 1 src: 11.0.0.5(age, ref_count, owner flag, pruned flag)
owner flag: 0x0: local, 0x1 remote cep, 0x2 remote ccep
vlan 100, has 1 caches.
1 (* 224.10.10.10) has 2 pim join ports out of 2 OIF
1/7/3 (1,ICL), 1/7/5 (1, CCEP)
```

Refer to the *RUCKUS FastIron Command Reference* for information on **show ipv6 multicast cache** and other MLD multicast commands.

Multicast Snooping Configuration Example

The following figure depicts a multicast snooping configuration. Sample configurations follow.

FIGURE 34 Multicast Snooping over MCT



The following example shows the configuration for multicast snooping for the MCT1 cluster device in Figure 34.

```
vlan 100 by port
tagged ethe 1/2/1
untagged ethe 1/1/5 ethe 1/1/6
multicast passive
multicast pimsm-snooping
multicast6 passive
multicast6 pimsm-snooping
!
vlan 3000 name session by port
tagged ethe 1/2/1
vlan 3001 name keep-alive-vlan
tagged eth 1/1/4
interface ve 3000
ip address 10.1.1.2 255.255.255.0
!
cluster ICX7850 3000
rbridge-id 2
session-vlan 3000
keep-alive-vlan 3001
icl ICL ethernet 1/2/1
peer 10.1.1.3 rbridge-id 3 icl ICL
deploy
client client-1
  rbridge-id 100
  client-interface ethernet 1/1/5
  deploy
!
```

The following example shows the configuration for multicast snooping for the MCT2 cluster device in Figure 34.

```
!
vlan 100 by port
tagged ethe 1/2/1
untagged ethe 1/1/5 ethe 1/1/6
multicast passive
multicast pimsm-snooping
multicast6 passive
multicast6 pimsm-snooping
!
vlan 3000 name session by port
tagged ethe 1/2/1
vlan 3001 name keep-alive-vlan
tagged eth 1/1/4
interface ve 3000
ip address 10.1.1.3 255.255.255.0
!
cluster ICX7850 3000
rbridge-id 3
session-vlan 3000
keep-alive-vlan 3001
icl ICL ethernet 1/2/1
peer 10.1.1.2 rbridge-id 2 icl ICL
deploy
client client-1
  rbridge-id 100
  client-interface ethernet 1/1/5
  deploy
!
```

The following example shows the global configuration for multicast snooping for the MCT1 cluster device in Figure 34.

```
vlan 100 by port
tagged ethe 1/2/1
untagged ethe 1/1/5 ethe 1/1/6
!
vlan 3000 name session by port
tagged ethe 1/2/1
vlan 3001 name keep-alive-vlan
tagged eth 1/1/4
ip multicast active
interface ve 3000
ip address 10.1.1.2 255.255.255.0
!
cluster ICX7850 3000
rbridge-id 2
session-vlan 3000
keep-alive-vlan 3001
icl ICX ethernet 1/2/1
peer 10.1.1.3 rbridge-id 3 icl ICX
deploy
client client-1
rbridge-id 100
client-interface ethernet 1/1/5
deploy
!
```

The following example shows the global configuration for multicast snooping for the MCT2 cluster device in Figure 34.

```
!
vlan 100 by port
tagged ethe 1/2/1
untagged ethe 1/1/5 ethe 1/1/6
!
vlan 3000 name session by port
tagged ethe 1/2/1
vlan 3001 name keep-alive-vlan
tagged eth 1/2/2
ip multicast passive
interface ve 3000
```

```
ip address 10.1.1.3 255.255.255.0
!
cluster ICX7850 3000
rbridge-id 3
session-vlan 3000
keep-alive-vlan 3001
icl ICL ethernet 1/2/1
peer 10.1.1.2 rbridge-id 2 icl ICL
deploy
client client-1
rbridge-id 100
client-interface ethernet 1/1/5
deploy
```

Layer 3 Behavior with MCT

The following table lists the type of Layer 3 support available with MCT.

Beginning with FastIron 10.0.00 IPv6 is supported over MCT.

TABLE 11 Layer 3 Feature Support with MCT

Feature	Sub-feature	Session VLAN VE	Member VLAN VE	Design Philosophy
ip	access-group	Yes	Yes	Only features that are relevant for MCT management are supported on session VLAN VE.
	address	Yes	Yes	
	arp-age	Yes	Yes	
	bgp	No	Yes	
	bootp-gateway	Yes	Yes	
	directed-broadcast	Yes	Yes	
	encapsulation	Yes	Yes	
	follow	No	No	
	helper-address	Yes	Yes	
	icmp	Yes	Yes	
	igmp	No	Yes	
	irdp	No	Yes	
	local-proxy-arp	No	Yes	
	metric	No	Yes	
	mld	No	Yes	
	mtu	Yes	Yes	
	multicast-boundary	No	No	
	ospf	No	Yes	
	pim	No	No	
	pim-sparse	No	Yes	
	policy	No	Yes	
	proxy-arp	No	Yes	
	redirect	No	Yes	
	rip	No	Yes	
	tcp	Yes	Yes	

TABLE 11 Layer 3 Feature Support with MCT (continued)

Feature	Sub-feature	Session VLAN VE	Member VLAN VE	Design Philosophy
	tunnel	No	Yes	
	use-acl-on-arp	Yes	Yes	
	vrrp	No	Yes	
	vrrp-extended	No	Yes	
ipv6		No	Yes	IPv6 is not supported for MCT management.
	mld	No	Yes	
	multicast-boundary	No	Yes	
	pim	No	Yes	
	pim-sparse	No	Yes	

Layer 3 Unicast Forwarding over MCT

A simple MCT topology addresses resiliency and efficient load balancing in Layer 2 network topologies. Layer 3 technologies can run in an MCT environment too. This allows various Layer 3 technologies to function while leveraging the benefits at the Layer 2 level.

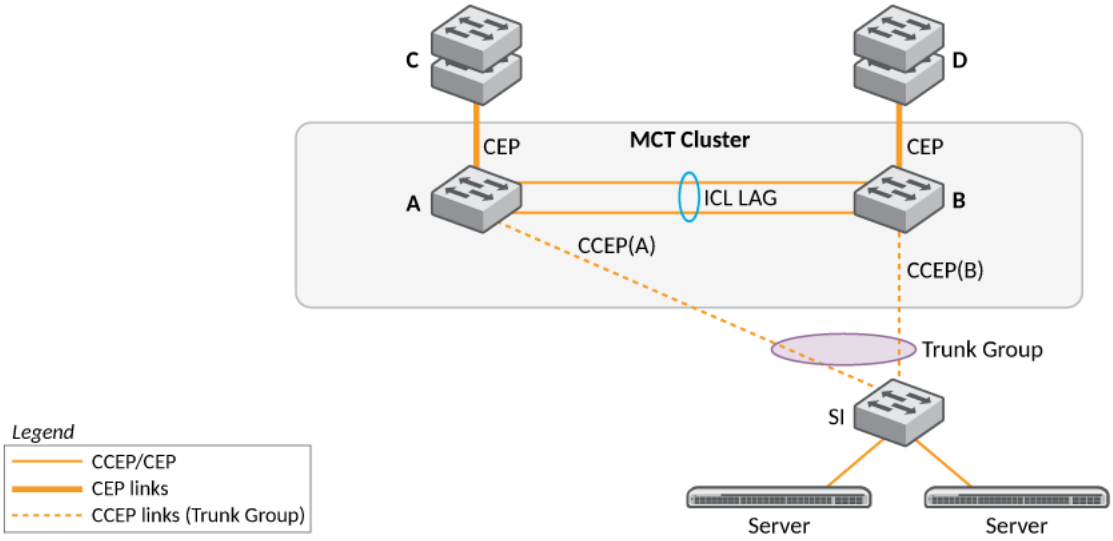
ARP Resolution

ARP resolution for the MCT client is required at the MCT cluster to forward traffic from a CEP to the CCEP. This ARP packet would normally be learned over the CCEP port. However, if the MAC address of the client is not already known on the CCEP, its ARP packet could be temporarily learned over the ICL. When the MAC Database Update Protocol (MDUP) message from the cluster peer device moves the client MAC address from the ICL to the CCEP, the ARP packet is also moved to the CCEP. During this transient time, no client ARP packet can be programmed over the ICL for a long period of time unless the local CCEP port is down.

During this transient time, the Layer 3 traffic gets forwarded toward the MCT peer. If the MCT client triggers an ARP request, it would do so for its Layer 3 next hop IP address, which generally resides on the MCT cluster devices. This address could be the default gateway on the MCT client or it could be learned through dynamic routing. If VRRP or VRRP-E is deployed on the MCT cluster devices, this IP address can be the virtual IP address.

Due to the inherent nature of LAG on the MCT client, this ARP request can reach an MCT device directly (over the CCEP) or through the MCT peer (over the ICL). In either case, the ARP response is sent out on the port where the client's MAC address is learned. If the MAC address is already learned on the MCT device at the time of receiving the ARP request, it would be over the CCEP under normal working conditions (local CCEP is in the up state). If the client's MAC address was not already learned when the ARP request is received, the client's ARP could be temporarily learned over the ICL (and is moved to the CCEP when the MDUP message from the peer is received) and the ARP response could be sent over the ICL. The cluster peer then switches the ARP response further towards the MCT client.

FIGURE 35 Configuration for Layer 3 Unicast



Layer 3 Traffic Forwarding Toward MCT Clients

Traffic destined to the MCT clients follows normal IP routing. By default, the best route must not involve the ICL link.

Traffic is rerouted to pass over the ICL only when the local CCEP is down.

Layer 3 Traffic Forwarding From MCT Clients

For Layer 3 forwarding to work on MCT devices, a dynamic trunk must be configured on the MCT client. Routes must be statically configured or dynamically learned on the MCT cluster devices.

The client routes the traffic towards its next hop, which can be either one of the MCT devices. If ECMP is deployed on the client, each MCT device can be a possible next hop. In such a deployment, the traffic can be load balanced at a Layer 3 level over the next two hops. Because a LAG is deployed at the client, this traffic is further subjected to load balancing at the Layer 2 level over the physical ports in the LAG. Thus, the traffic being sent out with the next hop as one of the MCT devices can either reach it directly or through the cluster peer (where it gets Layer 2 switched towards the intended next hop).

Therefore, almost 50 percent of traffic being forwarded from MCT clients (and as much as 100 percent of traffic in the worst case) can pass through the ICL. This fact should be considered when designing the ICL capacity in the network.

User-defined VRF Support over MCT

Virtual routing and forwarding (VRF) allows multiple instances of a routing table to coexist within a router. A router can have multiple VRF instances configured. The routing table, Forwarding Information Base (FIB), and so on are maintained separately for each VRF instance. A service provider can cater to multiple clients by keeping the routing information separate for each client, and different clients can use similar or overlapping IP addresses without the fear of information being sent to devices other than their own. For more information on VRF and related configurations, refer to the *Ruckus FastIron Layer 3 Routing Configuration Guide*.

Similar to the individual functionality of VRF, VRF over MCT allows the peer cluster devices to maintain separate routing and forwarding tables for each VRF instance, thus allowing overlapping of IP addresses, route isolation, and so on. In releases prior to FastIron 08.0.70, VEs spanning the MCT member VLAN ports supported only the default VRF. The MCT enhancement in FastIron 08.0.70 allows the MCT member ports to be added under user-defined VRF instances.

NOTE
User VRF support is for MCT member VLANs or VEs only. Session VLANs and keep-alive VLANs will remain in the default VRF.

VRF over MCT facilitates the support of Layer 3 routing protocol instances running under user VRF instances to be enabled on the MCT member ports. Thus, the supported routing protocols can maintain adjacency over each user VRF instance between the peer cluster devices, and clients and uplink devices. VRF over MCT also allows the MCT cluster peer IP address to be configured as the next hop or neighbor in different IGP protocols and helps to advertise or propagate the cluster (session VE) IP addresses.

VRRP or VRRP-E over an MCT-Enabled Network

To interface a Layer 2 MCT deployment with a Layer 3 network and add redundancy at the Layer 3 level, MCT can be configured with the Virtual Router Redundancy Protocol (VRRP). The standard VRRP mode is master-backup, and all traffic is forwarded through the master. In VRRP-E server virtualization, multiple VRRP standby devices are supported, and each device can be configured to route to an upstream Layer 3 network. This provides efficient deployment for both Layer 2 and Layer 3 networks.

The MCT device acting as a backup router will forward all packets destined to VRRP or VRRP-E virtual MAC address to the VRRP or VRRP-E master router for routing. The VRRP or VRRP-E backup learns the virtual MAC address while processing the VRRP hello message from the VRRP master. Both data traffic and VRRP or VRRP-E control traffic travel through the ICL unless the short-path forwarding feature is enabled (VRRP-E only).

VRRP or VRRP-E and VRRP-E2 short-path forwarding (SPF) should be enabled, if required. If VRRP is deployed or VRRP-E is deployed without the short-path forwarding feature on the VRRP-E backup, it is likely that almost 50 percent of CCEP to CEP traffic (and as much as 100 percent of traffic in the worst case) can pass through the ICL from the backup to the master device. This fact should be considered when designing ICL capacity in the network.

When one MCT device acts as a VRRP or VRRP-E master and the peer device is the VRRP or VRRP-E backup, the following behavior is observed:

- Frames sent to the VRRP or VRRP-E virtual MAC address are forwarded to the VRRP or VRRP-E master device for routing. The VRRP-E MAC address is learned by the other MCT device that acts as a backup router.
- Both data traffic and VRRP-E control traffic received by the VRRP backup from an MCT client must travel through the ICL, unless the short-path forwarding feature is enabled.

When both MCT devices act as the VRRP or VRRP-E backup, the following traffic behavior is observed:

- Frames sent to the VRRP or VRRP-E virtual MAC address are forwarded to the VRRP or VRRP-E master router for routing.
- The VRRP-E MAC address is learned by both MCT devices acting as backup routers.
- Both data traffic and VRRP-E control traffic travel through the links connecting them to the VRRP or VRRP-E master.

OSPF and BGP Over an MCT-Enabled Network

OSPF and BGP adjacencies can be established over the MCT member VLANs between any combinations of network elements in the MCT topology.

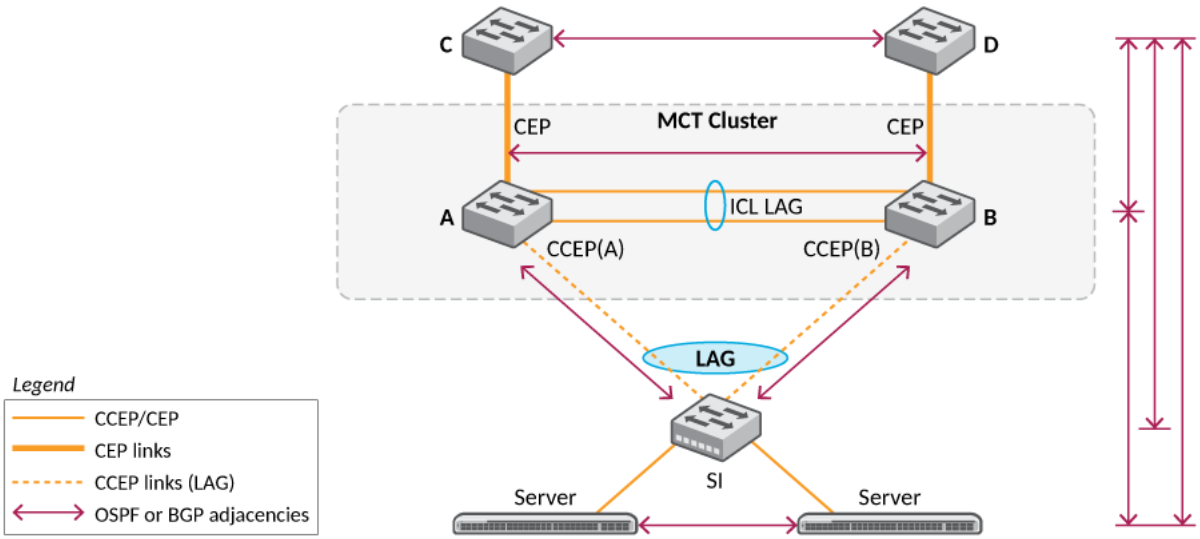
The following combinations can be established:

- Devices connected to the MCT cluster over CEP ports
- Devices connected to the MCT cluster over non-MCT ports
- MCT cluster devices
- MCT clients
- Devices behind MCT clients

  In such a deployment, the MCT clients and the devices behind them form separate protocol adjacencies with each MCT cluster device. These multiple Layer 3 next hops can be utilized by deploying ECMP on the MCT client device.

NOTE
The MCT failover will not be a hitless one for Layer 3 traffic because each MCT cluster device forms an independent adjacency. When one of the MCT devices goes down, a Layer 3 reconvergence is required and traffic loss is expected during this time.

FIGURE 36 OSPF and BGP Configuration in an MCT-Enabled Network



Layer 3 with MCT Configuration Considerations

The following configurations apply to Layer 3 behavior with MCT:

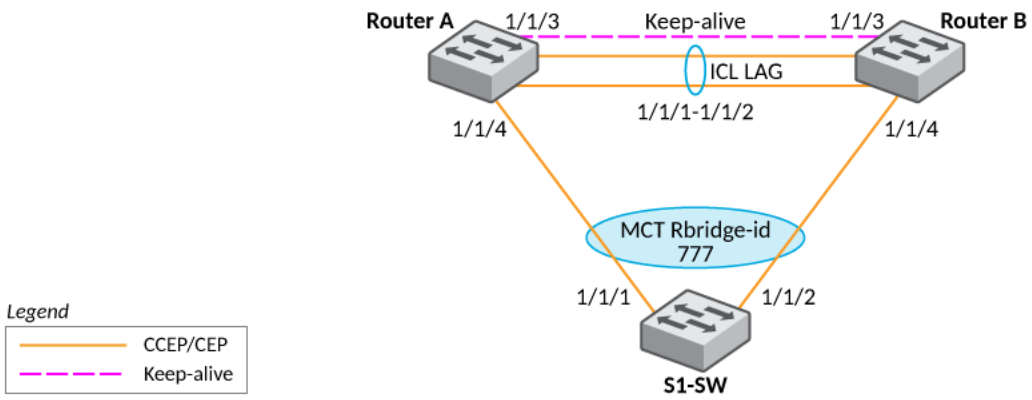
- Not all Layer 3 features on MCT management interface are supported. If a VLAN is already configured with these Layer 3 features, it cannot be made the session VLAN. To see the list of unsupported features on the MCT management interface, refer to Layer 3 Behavior with MCT on page 119.
- IPv6 configurations are supported on VEs of member VLANs.
- Route-only ports cannot be used as CCEPs or ICL ports.
- Global route-only configuration and MCT cluster configuration are mutually exclusive.
- Using MCT management interface IPs for a tunnel source is not supported.
- Configuring static and policy-based routes using the MCT management interface is not supported.
- Configurations to redistribute connected routes will not advertise IP addresses on an MCT management interface.
- IP addresses on the MCT management interface must not be used for BGP peers on neighboring devices.
- IP addresses on the MCT management interface must not be used for static configurations on neighboring devices.
- For MCT devices configured with VRRP or VRRP-E, track-port features can be enabled to track the link status to the core devices on the VRRP master, so the VRRP or VRRP-E failover can be triggered on the VRRP backup, so as to disable short-path forwarding when it loses its relevance.
- VRRP or VRRP-E must not be used along with OSPF or BGP on the same MCT member VE.

NOTE
To prevent unintended traffic forwarding by the CPU, RUCKUS recommends disabling ICMP redirect globally when VRRP or VRRP-E is configured.

MCT Configuration for a Single-Level MCT Deployment

The following figure shows a sample configuration for a single-level MCT deployment. The associated configuration follows.

FIGURE 37 Sample Configuration for a Single-Level MCT Deployment



MCT Configuration: RouterA

This example presents the MCT configuration for the RouterA cluster device.

```
!
lag lag_routera static id 55
 ports ethernet 1/1/1 to 1/1/2

!
 port-name "ICL-To_routerB_eth1/1/1" ethernet 1/1/1
 port-name "ICL-To_routerB_eth1/1/2" ethernet 1/1/2
!
!
vlan 110 name Member-vlan by port
 tagged ethe 1/1/4 lag 55
 interface ve 110
!
vlan 1000 name ICL-Session-vlan by port
 tagged lag 55
 interface ve 1000
!
vlan 1001 name MCT-Keep-Alive by port
 tagged ethe 1/1/3
!
interface ve 1001
 ip address 10.0.0.254 255.255.255.252
!
cluster FI-MCT 1750
 rbridge-id 801
 session-vlan 1000
 keep-alive-vlan 1001
 icl FI_SWR-MCT lag 55
 peer 10.0.0.253 rbridge-id 800 icl FI_SWR-MCT
 deploy
 client S1-SW
  rbridge-id 777
  client-interface ethe 1/1/4
  deploy
!
interface ve 110
 port-name S1-SW
 ip address 10.110.0.253 255.255.255.0
!
```

MCT Configuration: RouterB

This example presents the MCT configuration for the RouterB cluster device.

```
lag lag_routerb static id 55
ports ethernet 1/1/1 to 1/1/2

!
vlan 110 name Member-vlan by port
 tagged ethe 1/1/4 lag 55
 interface ve 110
!
vlan 1000 name ICL-Session-vlan by port
 tagged lag 55
 interface ve 1000
!
vlan 1001 name MCT-Keep-Alive by port
 tagged ethernet 1/1/3
!
interface ve 1001
 ip address 10.0.0.253 255.255.255.252
!
cluster FI-MCT 1750
 rbridge-id 800
 session-vlan 1000
 keep-alive-vlan 1001
 icl FI_SWR-MCT lag 55
 peer 10.0.0.254 rbridge-id 801 icl FI_SWR-MCT
 deploy
 client S1-SW
  rbridge-id 777
  client-interface ethernet 1/1/4
  deploy
!
interface ve 110
 port-name S1-SW
 ip address 10.110.0.252 255.255.255.0
!
```

MCT Configuration: S1-SW Device

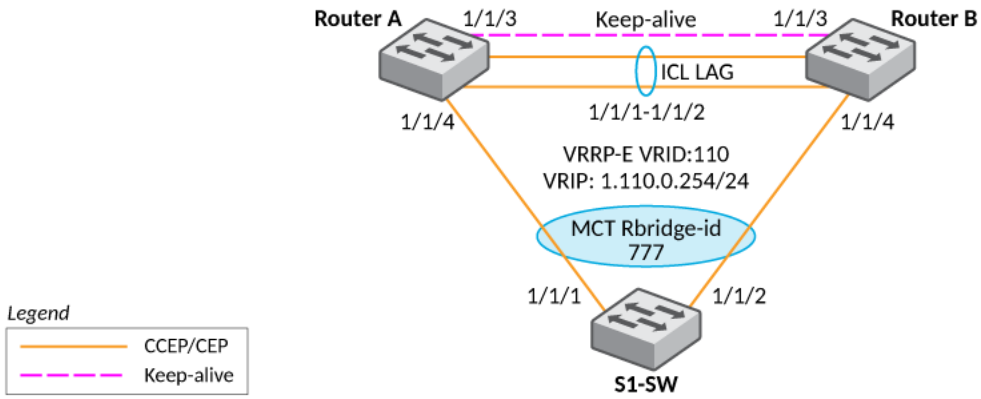
This example presents the configuration for the MCT deployment on the S1-SW device.

```
!
lag lag_s1_sw static id 60
 ports ethe 1/1/1 to 1/1/2
!
vlan 110 by port
tagged lag 60
!
interface ve 110
 ip address 10.110.0.1 255.255.255.0
!
```

MCT Configuration with VRRP-E

The following figure shows a sample MCT configuration with VRRP-E. The associated configuration follows. The configuration for VRRP is similar.

FIGURE 38 Sample MCT Configuration with VRRP-E



VRRP-E Configuration: RouterA

This example presents the VRRP-E configuration for the RouterA cluster device.

```
!
router vrrp-extended
!
interface ve 110
 port-name S1-SW
 ip address 10.110.0.253 255.255.255.0
 ip vrrp-extended vrid 110
  backup
  ip-address 10.110.0.254
  short-path-forwarding
  enable
!
```

VRRP-E Configuration: RouterB

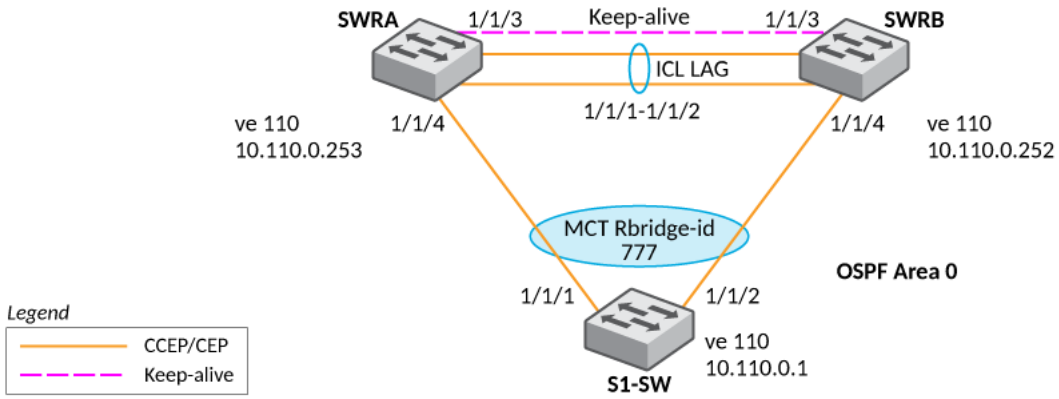
This example presents the VRRP-E configuration for the RouterB cluster device.

```
!
router vrrp-extended
!
interface ve 110
 port-name S1-SW
 ip address 10.110.0.252 255.255.255.0
 ip vrrp-extended vrid 110
  backup
  ip-address 10.110.0.254
  short-path-forwarding
  enable
!
```

MCT Configuration with OSPF

The following examples describe sample MCT configurations with OSPF.

FIGURE 39 MCT Configuration with OSPF



OSPF Configuration: SWRA

This example presents the OSPF configuration for the SWRA cluster device.

```
!
router ospf
area 0
!
interface ve 110
ip address 10.110.0.253 255.255.255.0
ip ospf area 0
!
```

OSPF Configuration: SWRB

This example presents the OSPF configuration for the SWRB cluster device.

```
!
router ospf
area 0
!
interface ve 110
ip address 10.110.0.252 255.255.255.0
ip ospf area 0
!
```

OSPF Configuration: S1-SW

This example presents the configuration for the S1-SW device.

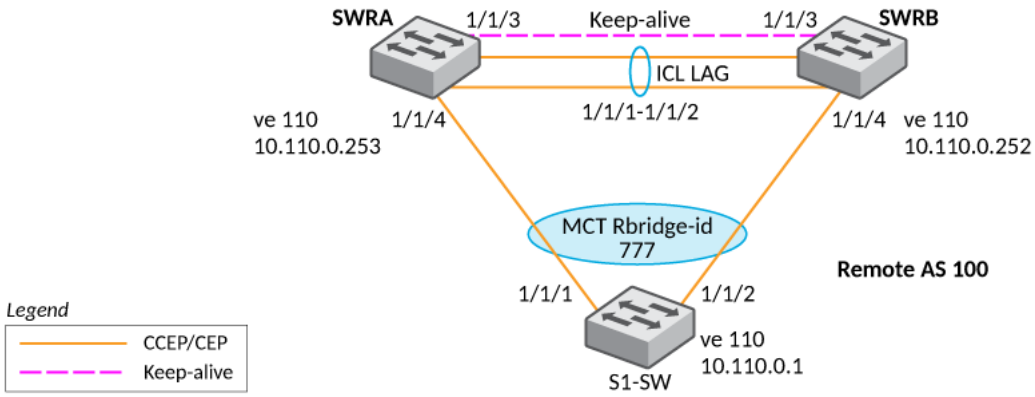
```
!
lag lag_s1_sw static id 60
ports ethernet 1/1/1 to 1/1/2

!
vlan 110 by port
tagged lag 60
!
router ospf
area 0
```

```
!
interface ve 110
ip address 10.110.0.1 255.255.255.0
ip ospf area 0
!
```

MCT Configuration with BGP

The following examples describe sample MCT configurations with BGP.

FIGURE 40 MCT Configuration with BGP



BGP Configuration: SWRA

This example presents the BGP configuration for the SWRA cluster device.

```
!
interface ve 110
ip address 10.110.0.253 255.255.255.0
!
router bgp
local-as 100
neighbor 10.110.0.252 remote-as 100
neighbor 10.110.0.1 remote-as 100
!
```

BGP Configuration: SWRB

This example presents the BGP configuration for the SWRB cluster device.

```
!
interface ve 110
ip address 10.110.0.252 255.255.255.0
!
router bgp
local-as 100
neighbor 10.110.0.253 remote-as 100
neighbor 10.110.0.1 remote-as 100
!
```

BGP Configuration: S1-SW

This example presents the BGP configuration for the S1-SW device.

```
!
lag lag_s1_sw static id 60
ports ethernet 1/1/1 to 1/1/2

!
vlan 110 by port
tagged ethernet lag 60
!
interface ve 110
ip address 10.110.0.1 255.255.255.0
!
router bgp
local-as 100
neighbor 10.110.0.253 remote-as 100
neighbor 10.110.0.252 remote-as 100
!
```

PIM Over MCT Intermediate Router Functionality

MCT peers support intermediate router functionality by accepting PIM neighbors on specific interfaces, thus routing multicast traffic as fully functional PIM devices acting as upstream and downstream routers.

MCT peers support Protocol Independent Multicast routing (PIM) on Cluster Client Edge Port (CCEP) and Inter-Chassis Link (ICL) interfaces.

PIM states between MCT peers are synchronized by sending the control packets natively over the ICL. The nature of the MCT LAG requires this. Packets from the MCT client on the CCEPs are received by only one of the MCT peers. Therefore the control packets that are received natively on the CCEPs are sent over the ICL to synchronize the states. The Join, Prune, and Assert packets are synchronized to maintain the Outgoing Interface (OIF) state for the CCEPs on both peers. For CCEP OIFs created by PIM joins, only one of the MCT peers forwards the traffic and the other peer drops the traffic.

These are the general rules followed for the control packet handling algorithm:

- Control packets originated from MCT peers will be flooded on the MCT VLAN. Exceptions are assert packets and join packets triggered only for ICL OIFs.
- Control packets received on any port of the MCT VLAN are flooded on the MCT VLAN.
- Control packets received on the ICL are flooded in a controlled manner on the MCT VLAN based on remote CCEP status, that is, based on whether they are up or down.

Control and data packets received on an ICL port are processed by searching the source MAC address of the packet in the MAC table to determine the packet ingress port as follows:

- If the source MAC address is learned on a CCEP, the packet ingress port will be a CCEP.
- If the source MAC address is not learned on a CCEP, the packet ingress port will be an ICL port.

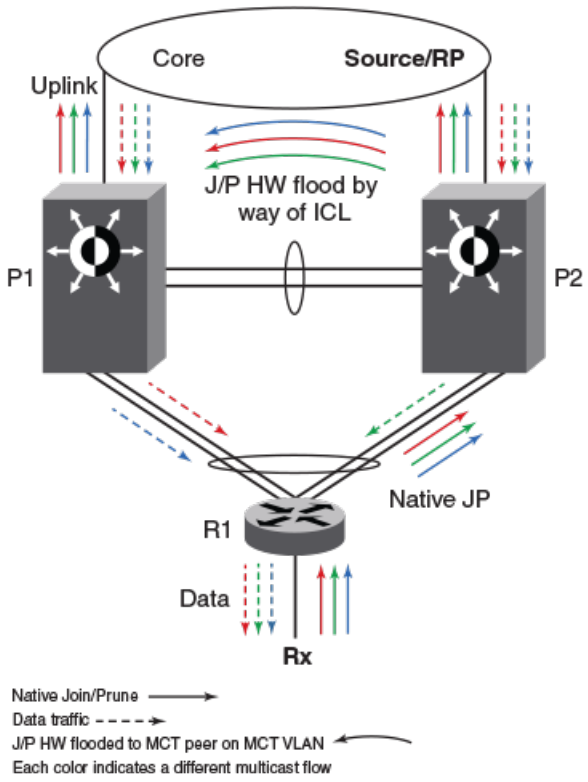
In the following figures, P1 and P2 are MCT peers and R1 is the MCT client. P1, P2, and R1 are configured with PIM on the MCT VE interface. MCT peers act as PIM intermediate routers with respect to R1.

MCT Peer as Intermediate Upstream Router

P1 and P2 are the MCT peers and are acting as upstream routers for R1. R1 is the last-hop router (LHR).

P1, P2, and R1 are configured with PIM on the MCT virtual Ethernet (VE) interface. Rendezvous Point (RP) and source are in the core and the connectivity to the core is by way of an uplink.

FIGURE 41 MCT Peer as Intermediate Upstream Router



Hello Exchange and Neighbor State

- In MCT topology, the CCEP links going out of P1 and P2 to R1 are treated as a single LAG at R1. When R1 sends multicast packets (either control or data packets), they reach only one of the peers. These control packets (hellos, joins, prunes, and others) received by one peer are flooded on the MCT VLAN including the ICL port to the other peer.
- Hellos sent by R1 can reach either P1 or P2 due to nature of the MCT LAG.
- Hellos that reach P2 are sent to P1 natively over ICL. P1 learns about R1 (by searching the source-MAC address of the hello packet in its MAC table), and it treats the hello as if it was received on its CCEP interface. Thus, both P1 and P2 learn about the PIM neighbors across the CCEP links and create a neighbor state for R1.
- Hellos originating from P1 and P2 are flooded on the MCT VLAN (for example on the ICL, CEP, and local CCEPs). This enables R1 to learn that both the MCT peers are PIM neighbors and also enables P1 and P2 to learn about each other as PIM neighbors on an ICL link and create a neighbor state for each other.

Join or Prune Exchange and Mcache State

- Because receivers are connected to R1, R1 creates the *,G state and sends a join state toward the RP and sends it on the MCT LAG. This join, like any other packet, is received by only one of the MCT peers.
- If P2 receives the *,G join natively, the join is processed or consumed and flooded to P1 over the ICL.
- P1 processes the join received over the ICL as if it is received on the CCEP.
- Both P1 and P2 create the *,G state with the CCEP as the OIF.
- Both the peers send the *,G join toward the RP, and both the peers pull the traffic.

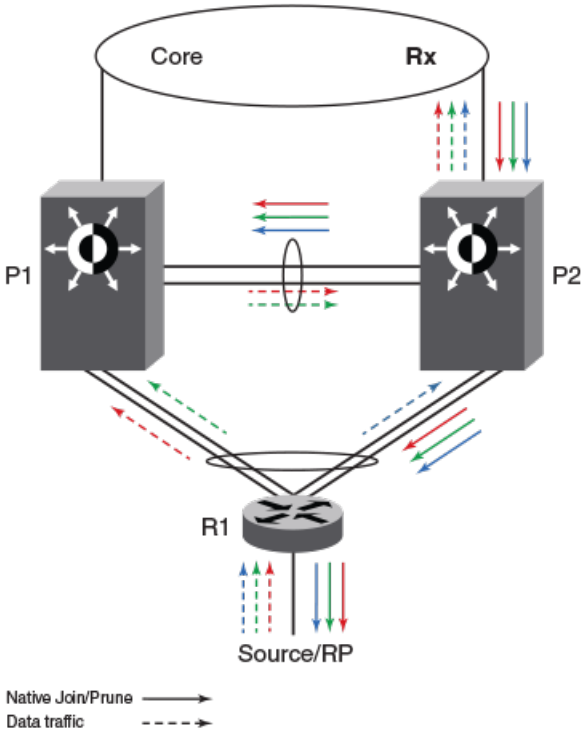
- When the traffic arrives, the S,G state is created on both the peers, but only one of them forwards the traffic based on the software hashing algorithm.

MCT Peer as Intermediate Downstream Router

P1 and P2 are the MCT peers and are acting as downstream routers for R1. R1 is the intermediate router.

P1, P2, and R1 are configured with PIM on the MCT VE interface. The RP and source are beyond R1.

FIGURE 42 MCT Peer as Intermediate Downstream Router



Hello Exchange and Neighbor State

The RP acts and works as the upstream router.

Join or Prune Exchange and Mcache State

- The *,G joins come from the core to P2.
- P2 creates the *,G state with the uplink as the OIF by consuming the join state.
- Due to its *,G state, P2 originates a join towards the RP. This join is flooded on the MCT VLAN and R1 creates the *,G state.
- Upon receiving the join natively by way of ICL, P1 creates the *,G state and adds ICL as the OIF. As a special case, P1 will not include the *,G in the join it generates towards the RP because in this case the IIF is the CCEP and ICL is the only OIF and the remote CCEP is up. This is to avoid P1 pulling traffic from P2 unnecessarily on the ICL link because of P1 sending joins flooded on the VLAN and in turn P2 adds ICL as an OIF.
- R1 sends the join toward the RP and pulls the traffic. Because the OIF at R1 is a LAG, traffic pulled by R1 will be load-shared among the member links.

- Therefore traffic for S,G will reach only one of the MCT peers. Assuming the traffic reaches P2, the S,G state will be created on P2 and P2 will be forwarding the traffic.
- Assuming the traffic reaches P1, the traffic will be forwarded by way of the ICL to P2 and P2 will forward it to its OIF which is the link connecting to the core.

Load Sharing of Multicast Traffic by MCT-Cluster on CCEP Links

MCT peers load-share multicast traffic on both the local and the remote CCEP links when both are available.

Loads are only shared and may or may not be balanced across the CCEP links. An MCT peer selects a stream for forwarding based on a software hash function that uses the source and group addresses. You can have one MCT peer forwarding more multicast streams than another.

The load is assigned without regard to the capacity of the CCEP links, so MCT works best when both CCEP links have the same capacity and the source and group addresses are evenly distributed. This situation avoids the timing synchronization between the MCT peer routers, which would be very hard to achieve.

The sharing is done at the stream level (not the packet level) using the following software hash algorithm:

((source address + group address) & 0x00000001) ^ ((local_bridge_id > remote_bridge_id))

If the result is 1, the local CCEP forwards the traffic; if the result is 0, the remote CCEP forwards the traffic.

Fast Convergence of Multicast Traffic

Multicast routing on MCT provides sub-second convergence of traffic in the event of CCEP or MCT peer failures and recoveries.

When a CCEP or MCT peer fails, multicast traffic that used to go through the failed CCEP link or node switches to the surviving CCEP link in approximately one second or less.

Sub-second convergence requires both MCT peers to maintain the state for traffic and pull down traffic for all multicast flows from the core, regardless of whether the chassis is forwarding this stream out of the local CCEP. This means that streams forwarded by the remote CCEP are pulled down to the local MCT peer but dropped in the absence of other receivers on the local router, thus potentially wasting the bandwidth inside the core on uplink. This is deemed a fair trade-off because otherwise the MCT peer that takes over the job of forwarding a stream when the remote CCEP or peer fails must establish a new multicast path through the core, which can potentially black out the stream for many seconds.

First Hop and Last Hop Router Support on MCT Cluster Devices for MCT VLANs

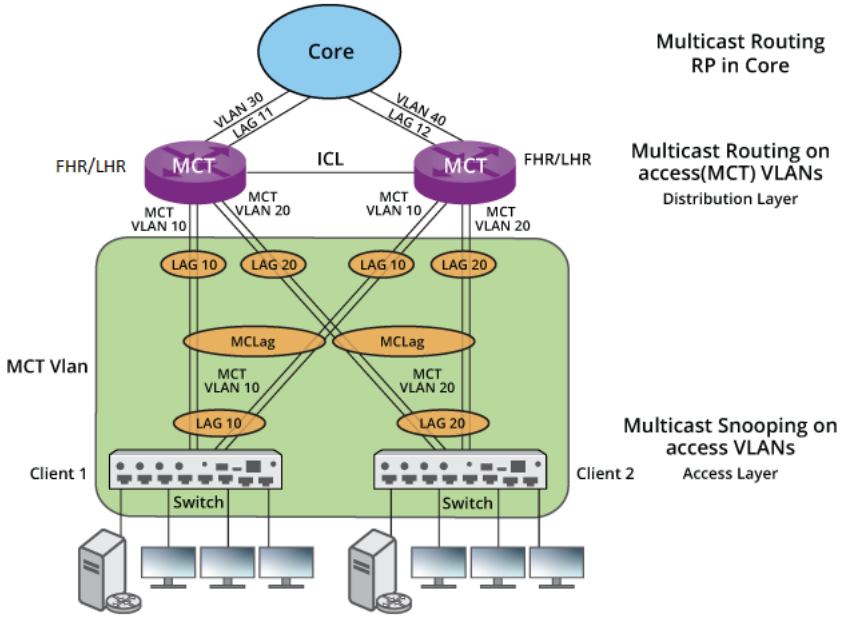
A First Hop Router (FHR), a multicast source that is directly connected to a router, via cable or a switch, is supported on PIM cluster peers for MCT.

A multicast source device, acting as an MCT client, or connected to an MCT client switch, can forward the multicast traffic to any of the cluster devices. In order to provide sub-second convergence, the cluster device that is receiving the multicast traffic from a source on its CCEP, forwards the traffic over ICL to the cluster peer. The traffic is forwarded to the cluster peer irrespective of whether the cluster peer has any receivers for this traffic. Therefore, the multicast traffic from a MCT source client received by a cluster device can be forwarded to other MCT receiver clients by a peer cluster device.

Similarly, a Last Hop Router (LHR), a multicast receiver that is directly connected to a router, via cable or switch, is supported for PIM cluster peers for MCT. The source traffic will ingress on the First Hop and then cross intermediate routers until it reaches the last hop router and the receiver receives the multicast packets.

The following illustration shows a single-tier MCT deployment in a 3-tier architecture. In this 1-tier MCT deployments, MCT is used at the distribution layers. Distribution switches are paired as an MCT cluster at tier-1. Each access switch is connected, via a LAG interface, as an MCT client to two distribution switches that are configured in an MCT cluster. Multicast Snooping is configured on the access switches for the access-VLANs. On the distribution switches that are MCT clusters, multicast routing is configured on these access MCT VLANs or VEs.

FIGURE 43 Sample MCT Configuration in a 3-tier Architecture



Configuration Example: Multicast Routing with FHR and LHR

The following example shows the configuration for multicast snooping for the 3-tier architecture depicted in the figure above.

Client1:

```
vlan 10 by port
 tagged ethernet 1/2/2 lag 10
 multicast passive
!
```

Client2:

```
vlan 20 by port
 tagged ethernet 1/2/2 lag 20
 multicast passive
!
```

PEER1:

```
router pim
!
router ospf
 area 0
 no graceful-restart
 nonstop-routing
!
vlan 5 by port
 tagged lag 1
!
vlan 6 by port
 tagged lag 2
!
vlan 10 by port
 tagged lag 1 lag 10
!
vlan 20 by port
 tagged lag 1 lag 20
```

```
!
vlan 30 by port
 tagged lag 11
!
interface ve 5
 ip address 5.1.1.1 255.255.255.0
!
interface ve 10
 ip address 10.1.1.1 255.255.255.0
 ip pim-sparse
 ip ospf area 0
!
interface ve 20
 ip address 20.1.1.1 255.255.255.0
 ip pim-sparse
 ip ospf area 0
!
interface ve 30
 ip address 30.1.1.1 255.255.255.0
 ip pim-sparse
 ip ospf area 0
!
cluster mcast_cluster 4095
 rbridge-id 4090
 session-vlan 5
 keep-alive-vlan 6
 icl-fwd-delay 5
 ccep-up-delay 300
 icl icl-lag lag 1
 peer 5.1.1.2 rbridge-id 4091 icl icl-lag
 peer 5.1.1.2 timers keep-alive 10 hold-time 300
 deploy
 client ccep-1
  rbridge-id 1
  client-interface lag 10
  deploy
 client ccep-2
  rbridge-id 2
  client-interface lag 20
  deploy
!
```

PEER2 :

```
router pim
!
router ospf
 area 0
 no graceful-restart
 nonstop-routing
!
vlan 5 by port
 tagged lag 1
!
vlan 6 by port
 tagged lag 2
!
vlan 10 by port
 tagged lag 1 lag 10
!
vlan 20 by port
 tagged lag 1 lag 20
!
vlan 40 by port
 tagged lag 12
!
interface ve 5
 ip address 5.1.1.2 255.255.255.0
!
interface ve 10
 ip address 10.1.1.2 255.255.255.0
 ip pim-sparse
```

```
 ip ospf area 0
!
interface ve 20
 ip address 20.1.1.2 255.255.255.0
 ip pim-sparse
 ip ospf area 0
!
interface ve 40
 ip address 40.1.1.1 255.255.255.0
 ip pim-sparse
 ip ospf area 0
!

cluster mcast_cluster 4095
 rbridge-id 4091
 session-vlan 5
 keep-alive-vlan 6
 icl-fwd-delay 5
 ccep-up-delay 300
 icl icl-lag lag 1
 peer 5.1.1.1 rbridge-id 4090 icl icl-lag
 peer 5.1.1.1 timers keep-alive 10 hold-time 300
 deploy
 client ccep-1
  rbridge-id 1
  client-interface lag 10
  deploy
 client ccep-2
  rbridge-id 2
  client-interface lag 20
  deploy
!
```

CORE:

```
interface loopback 1
 ip address 1.1.1.1 255.255.255.255
 ip pim-sparse
 ip ospf area 0
!
router pim
 bsr-candidate loopback 1 32 255
 rp-candidate loopback 1
 rp-candidate add 224.0.0.0 4
!
router ospf
 area 0
 no graceful-restart
 nonstop-routing
!
vlan 30 by port
 tagged lag 11
!
vlan 40 by port
 tagged lag 12
!
vlan 50 by port
 tagged ethernet 1/2/2
!
interface ve 30
 ip address 30.1.1.2 255.255.255.0
 ip pim-sparse
 ip ospf area 0
!
interface ve 40
 ip address 40.1.1.2 255.255.255.0
 ip pim-sparse
 ip ospf area 0
!
interface ve 50
 ip address 50.1.1.1 255.255.255.0
 ip pim-sparse
```

```
 ip ospf area 0
!
```

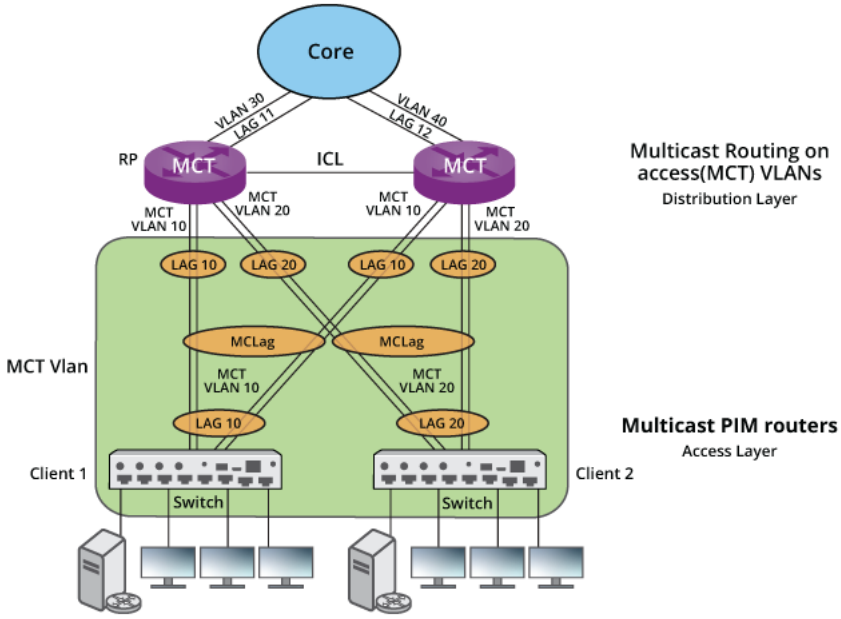
RP Support on MCT Cluster Devices

Rendezvous point (RP) is supported on MCT Cluster peer.

MCT cluster devices can be configured to be RP for multicast groups. Any one peer device can act as an RP. This can be used to facilitate traffic flow between source and receiver. For example, a receiver can join a multicast group after contacting the RP to show it is interested in the multicast group. Similarly, a source device can send traffic to an RP. The RP consolidates all traffic received from source and receiver devices, forwarding traffic as necessary.

Therefore, the RP acts as a meeting point for source and receiver devices. The following illustration shows an MCT deployment where an MCT cluster device is configured for an RP.

FIGURE 44 Sample MCT Configuration Using an RP



Configuration Example: Multicast Routing with an RP

The following example shows the configuration for multicast snooping where an MCT cluster device is configured as an RP, as depicted in the figure above.

Client1:

```
router pim
!
router ospf
 area 0
 no graceful-restart
 nonstop-routing
!
vlan 10 by port
 tagged lag 10
!
vlan 15 by port
 tagged ethe 1/2/2
```

```
!
interface ve 15
 ip address 15.1.1.1 255.255.255.0
 ip pim-sparse
 ip ospf area 0
!
interface ve 10
 ip address 10.1.1.3 255.255.255.0
 ip pim-sparse
 ip ospf area 0
!
```

Client2:

```
router pim
!
router ospf
 area 0
 no graceful-restart
 nonstop-routing
!
vlan 20 by port
 tagged lag 20
!
vlan 25 by port
 tagged ethe 1/2/2
!
interface ve 25
 ip address 25.1.1.1 255.255.255.0
 ip pim-sparse
 ip ospf area 0
!
interface ve 20
 ip address 20.1.1.3 255.255.255.0
 ip pim-sparse
 ip ospf area 0
!
```

PEER1:

```
interface loopback 1
 ip address 1.1.1.1 255.255.255.255
 ip pim-sparse
 ip ospf area 0
!
router pim
 bsr-candidate loopback 1 32 255
 rp-candidate loopback 1
 rp-candidate add 224.0.0.0 4
!
router ospf
 area 0
 no graceful-restart
 nonstop-routing
!
vlan 5 by port
 tagged lag 1
!
vlan 6 by port
 tagged lag 2
!
vlan 10 by port
 tagged lag 1 lag 10
!
vlan 20 by port
 tagged lag 1 lag 20
!
vlan 30 by port
 tagged lag 11
!
interface ve 5
 ip address 5.1.1.1 255.255.255.0
```

```
!
interface ve 10
 delay-notifications 15
 ip address 10.1.1.1 255.255.255.0
 ip pim-sparse
 ip pim dr-priority 10
 ip ospf area 0
 ip ospf priority 10
!
interface ve 20
 delay-notifications 15
 ip address 20.1.1.1 255.255.255.0
 ip pim-sparse
 ip pim dr-priority 10
 ip ospf area 0
 ip ospf priority 10
!
interface ve 30
 ip address 30.1.1.1 255.255.255.0
 ip pim-sparse
 ip ospf area 0
!
cluster mcast_cluster 4095
 rbridge-id 4090
 session-vlan 5
 keep-alive-vlan 6
 icl-fwd-delay 5
 ccep-up-delay 120
 icl icl-lag lag 1
 peer 5.1.1.2 rbridge-id 4091 icl icl-lag
 peer 5.1.1.2 timers keep-alive 10 hold-time 300
 deploy
 client ccep-1
  rbridge-id 1
  client-interface lag 10
  deploy
 client ccep-2
  rbridge-id 2
  client-interface lag 20
  deploy
!
```

PEER2 :

```
router pim
!
router ospf
 area 0
 no graceful-restart
 nonstop-routing
!
vlan 5 by port
 tagged lag 1
!
vlan 6 by port
 tagged lag 2
!
vlan 10 by port
 tagged lag 1 lag 10
!
vlan 20 by port
 tagged lag 1 lag 20
!
vlan 40 by port
 tagged lag 12
!
interface ve 5
 ip address 5.1.1.2 255.255.255.0
!
interface ve 10
 delay-notifications 15
 ip address 10.1.1.2 255.255.255.0
```

```
 ip pim-sparse
 ip pim dr-priority 10
 ip ospf area 0
 ip ospf priority 10
!
interface ve 20
 delay-notifications 15
 ip address 20.1.1.2 255.255.255.0
 ip pim-sparse
 ip pim dr-priority 10
 ip ospf area 0
 ip ospf priority 10
!
interface ve 40
 ip address 40.1.1.1 255.255.255.0
 ip pim-sparse
 ip ospf area 0
!

cluster mcast_cluster 4095
 rbridge-id 4091
 session-vlan 5
 keep-alive-vlan 6
 icl-fwd-delay 5
 ccep-up-delay 120
 icl icl-lag lag 1
 peer 5.1.1.1 rbridge-id 4090 icl icl-lag
 peer 5.1.1.1 timers keep-alive 10 hold-time 300
 deploy
 client ccep-1
  rbridge-id 1
  client-interface lag 10
  deploy
 client ccep-2
  rbridge-id 2
  client-interface lag 20
  deploy
!
```

CORE:

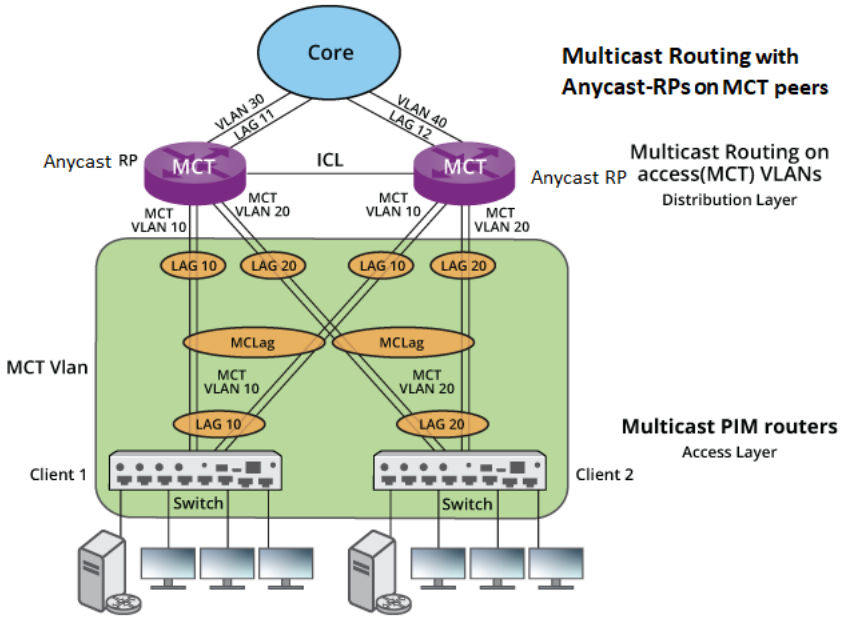
```
router pim
!
router ospf
 area 0
 no graceful-restart
 nonstop-routing
!
vlan 30 by port
 tagged lag 11
!
vlan 40 by port
 tagged lag 12
!
vlan 50 by port
 tagged ethernet 1/2/2
!
interface ve 30
 ip address 30.1.1.2 255.255.255.0
 ip pim-sparse
 ip ospf area 0
!
interface ve 40
 ip address 40.1.1.2 255.255.255.0
 ip pim-sparse
 ip ospf area 0
!
interface ve 50
 ip address 50.1.1.1 255.255.255.0
 ip pim-sparse
 ip ospf area 0
!
```

Anycast RP Support for MCT Cluster Devices

MCP cluster devices can be configured as anycast RP peers, thus providing redundancy to the RP.

RP is necessary in a PIM domain. Therefore, if the cluster peer which is an RP goes down, PIM can continue to function as both cluster peers are configured as RPs. The following illustration shows an MCT deployment where MCT cluster devices are configured for anycast RPs.

FIGURE 45 Sample MCT Configuration Using Anycast RPs



Configuration Example: Multicast Routing with anycasat RPs

The following example shows the configuration for multicast snooping where MCT cluster devices are configured as anycast RPs, as depicted in the figure above.

Client1:

```
router pim
 rp-address 1.1.1.1
!
router ospf
 area 0
 no graceful-restart
 nonstop-routing
!
vlan 10 by port
 tagged lag 10
!
vlan 15 by port
 tagged ethe 1/2/2
!
interface ve 15
 ip address 15.1.1.1 255.255.255.0
 ip pim-sparse
 ip ospf area 0
!
interface ve 10
 ip address 10.1.1.3 255.255.255.0
 ip pim-sparse
 ip ospf area 0
```

```
!
```

Client2:

```
router pim
 rp-address 1.1.1.1
!
router ospf
 area 0
 no graceful-restart
 nonstop-routing
!
vlan 20 by port
 tagged lag 20
!
vlan 25 by port
 tagged ethe 1/2/2
!
interface ve 25
 ip address 25.1.1.1 255.255.255.0
 ip pim-sparse
 ip ospf area 0
!
interface ve 20
 ip address 20.1.1.3 255.255.255.0
 ip pim-sparse
 ip ospf area 0
!
```

PEER1:

```
ip access-list standard anycast_rp_default_vrf
 sequence 10 permit host 2.2.2.1
 sequence 20 permit host 2.2.2.2
!
interface loopback 1
 ip address 1.1.1.1 255.255.255.255
 ip pim-sparse
 ip ospf area 0
!
interface loopback 2
 ip address 2.2.2.1 255.255.255.255
 ip pim-sparse
 ip ospf area 0
!
router pim
 rp-address 1.1.1.1
 anycast-rp 1.1.1.1 anycast_rp_default_vrf
!
router ospf
 area 0
 no graceful-restart
 nonstop-routing
!
vlan 5 by port
 tagged lag 1
!
vlan 6 by port
 tagged lag 2
!
vlan 10 by port
 tagged lag 1 lag 10
!
vlan 20 by port
 tagged lag 1 lag 20
!
vlan 30 by port
 tagged lag 11
!
interface ve 5
 ip address 5.1.1.1 255.255.255.0
!
```

```
interface ve 10
 delay-notifications 15
 ip address 10.1.1.1 255.255.255.0
 ip pim-sparse
 ip pim dr-priority 10
 ip ospf area 0
 ip ospf priority 10
!
interface ve 20
 delay-notifications 15
 ip address 20.1.1.1 255.255.255.0
 ip pim-sparse
 ip pim dr-priority 10
 ip ospf area 0
 ip ospf priority 10
!
interface ve 30
 ip address 30.1.1.1 255.255.255.0
 ip pim-sparse
 ip ospf area 0
!
cluster mcast_cluster 4095
 rbridge-id 4090
 session-vlan 5
 keep-alive-vlan 6
 icl-fwd-delay 5
 ccep-up-delay 120
 icl icl-lag lag 1
 peer 5.1.1.2 rbridge-id 4091 icl icl-lag
 peer 5.1.1.2 timers keep-alive 10 hold-time 300
 deploy
 client ccep-1
  rbridge-id 1
  client-interface lag 10
  deploy
 client ccep-2
  rbridge-id 2
  client-interface lag 20
  deploy
!
```

PEER2 :

```
ip access-list standard anycast_rp_default_vrf
 sequence 10 permit host 2.2.2.1
 sequence 20 permit host 2.2.2.2
!
interface loopback 1
 ip address 1.1.1.1 255.255.255.255
 ip pim-sparse
 ip ospf area 0
!
interface loopback 2
 ip address 2.2.2.2 255.255.255.255
 ip pim-sparse
 ip ospf area 0
!
router pim
 rp-address 1.1.1.1
 anycast-rp 1.1.1.1 anycast_rp_default_vrf
!
router ospf
 area 0
 no graceful-restart
 nonstop-routing
!
vlan 5 by port
 tagged lag 1
!
vlan 6 by port
 tagged lag 2
!
```

```
vlan 10 by port
 tagged lag 1 lag 10
!
vlan 20 by port
 tagged lag 1 lag 20
!
vlan 40 by port
 tagged lag 12
!
interface ve 5
 ip address 5.1.1.2 255.255.255.0
!
interface ve 10
 delay-notifications 15
 ip address 10.1.1.2 255.255.255.0
 ip pim-sparse
 ip pim dr-priority 10
 ip ospf area 0
 ip ospf priority 10
!
interface ve 20
 delay-notifications 15
 ip address 20.1.1.2 255.255.255.0
 ip pim-sparse
 ip pim dr-priority 10
 ip ospf area 0
 ip ospf priority 10
!
interface ve 40
 ip address 40.1.1.1 255.255.255.0
 ip pim-sparse
 ip ospf area 0
!

cluster mcast_cluster 4095
 rbridge-id 4091
 session-vlan 5
 keep-alive-vlan 6
 icl-fwd-delay 5
 ccep-up-delay 120
 icl icl-lag lag 1
 peer 5.1.1.1 rbridge-id 4090 icl icl-lag
 peer 5.1.1.1 timers keep-alive 10 hold-time 300
 deploy
 client ccep-1
  rbridge-id 1
  client-interface lag 10
  deploy
 client ccep-2
  rbridge-id 2
  client-interface lag 20
  deploy
!
```

CORE:

```
router pim
 rp-address 1.1.1.1
!
router ospf
 area 0
 no graceful-restart
 nonstop-routing
!
vlan 30 by port
 tagged lag 11
!
vlan 40 by port
 tagged lag 12
!
vlan 50 by port
 tagged ethernet 1/2/2
```

```
!
interface ve 30
 ip address 30.1.1.2 255.255.255.0
 ip pim-sparse
 ip ospf area 0
!
interface ve 40
 ip address 40.1.1.2 255.255.255.0
 ip pim-sparse
 ip ospf area 0
!
interface ve 50
 ip address 50.1.1.1 255.255.255.0
 ip pim-sparse
 ip ospf area 0
!
```

Requirements for Multicast MCT

- OSPF must be supported on MCT member VLAN virtual Ethernet (VE) interfaces; that is, on CCEP, CEP, and ICL links.
- The **ccep-up-delay** command should be configured in order to get better convergence when the whole cluster peer goes down or is reloaded, or when the complete ICL LAG flaps. The recommended value to configure is 300s (75k mac scale, 1200 IGMP snoop mcache, 1200 MLS snoop mcache , 1200 PIMv4 mcache).
- The **delay-notifications** command should be configured to avoid a VE interface flap when the whole cluster peer goes down or is reloaded, or when a complete ICL LAG goes down. The recommended value to configure is 15s.
- It is recommended that MCT cluster peers are PIM, OSPF DR or BDR.
- It is recommended to use Loose mode with a keep-alive VLAN.

Limitations

The following limitations apply for MCT peers to support intermediate router functionality. The limitations are due to load-sharing and fast convergence trade-offs.

- PIM-DM is not supported.
- A few packets may be lost during the convergence interval or forwarding duplication may occur.
- An MCT client will do flow-based load-sharing, not per-packet load-sharing.
- Traffic loss or duplication will occur when the keep-alive VLAN, CCP, or the ICL between MCT peers is not up.
- Multicast routing configurations on a session VLAN is not supported and restricted in configuration.
- The load will only be shared and may or may not be balanced across the CCEPs.
- During the convergence interval, a few packets may be lost. In the case of recoveries, some packets may end up being forwarded by both cluster routers during the interval.
- Both the MCT peers maintain the state for traffic and pull down traffic for all multicast flows from the core, whether or not the chassis is forwarding this stream to the local CCEP. This could potentially waste the bandwidth inside the core and on the uplink.
- You can configure both MCT peers to do either PIM routing or multicast snooping in MCT VLANs. However, configuring one MCT peer to do PIM routing and the other to do multicast snooping in the same MCT VLAN is not supported.
- A PIM neighbor on the CEP in an MCT VLAN is not supported if the MCT cluster is running PIM on the same MCT VLAN.
- If loose mode is configured without a keep-alive VLAN, packet forwarding will be unpredictable. It is therefore recommended not to use loose mode without a keepalive VLAN.
- If a core device is not a MCT client, convergence may not be optimal during cluster peer reload or failover, or when the ICL lag flaps. For better convergence it is recommended that the core is also an MCT client.

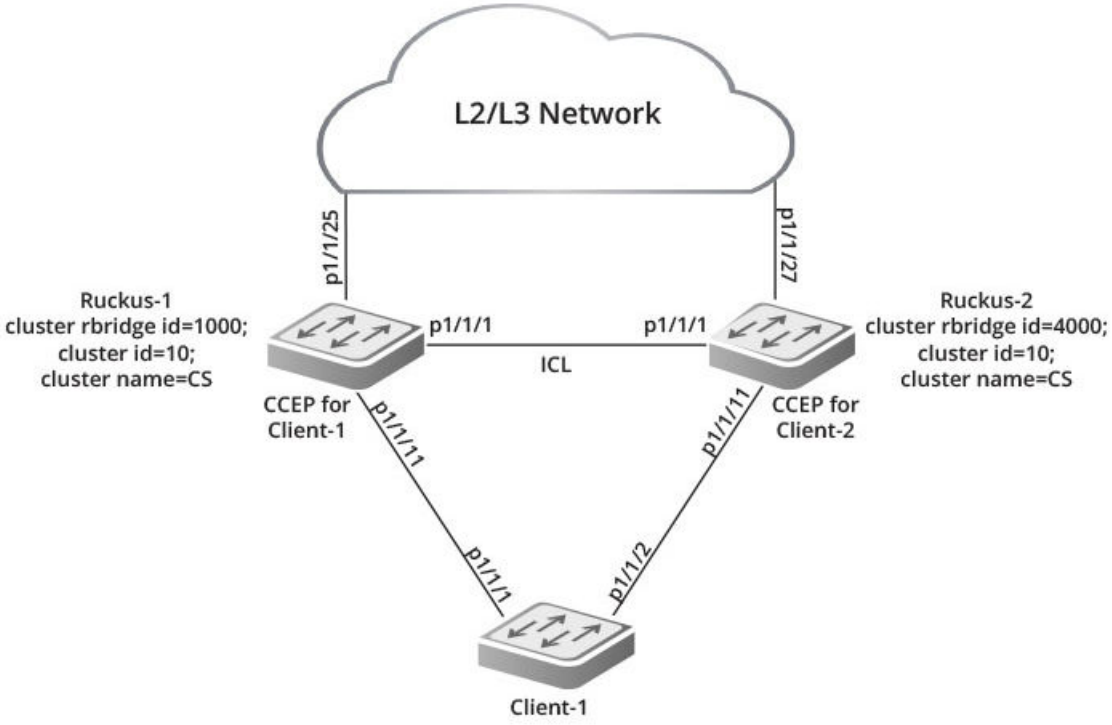
- When clients are PIMrouters, a momentary traffic spike may occasionally occur when an ICL lag flaps, or when a cluster peer rejoins after a reload.

Configuring Multicast Routing Over MCT for IPv4

Complete the following three steps to configure multicast routing over MCT for IPv4.

> **NOTE**
> Similar configuration steps can be used for configuring multicast routing over MCT for IPv6.

FIGURE 46 Multicast Routing Over MCT for IPv4



1. Configure an MCT cluster.
2. Configure an MCT member VLAN.
3. Configure multicast routing (PIM) over MCT member VE.

The following example shows the configuration of an MCT cluster, MCT member VLAN with router interface (VE), and PIM configuration over MCT member VE on MCT Peer 1.

```
cluster cs 10
 rbridge-id 1000
 session-vlan 4
 keep-alive-vlan 5
 icl MCT ethernet 1/1/1
 peer 5.5.5.100 rbridge-id 4000 icl MCT
 deploy
 client client-100
  rbridge-id 100
  client-interface ethernet 1/1/11
  deploy
!
!
!
!
!
!
end


vlan 10 name member-vlan by port
 tagged ethe 1/1/1 ethe 1/1/11 ethe 1/1/25
!
!

interface ve 10
 ip address 10.10.10.100 255.255.255.0
 ip pim-sparse
 ip ospf area 0
```

The following example shows the configuration of an MCT cluster, MCT member VLAN with router interface (VE), and PIM configuration over MCT member VE on MCT Peer 2.

```
cluster cs 10
 rbridge-id 4000
 session-vlan 4
 keep-alive-vlan 5
 icl MCT ethernet 1/1/1
 peer 5.5.5.10 rbridge-id 1000 icl MCT
 deploy
 client client-100
  rbridge-id 100
  client-interface ethernet 1/1/11
  deploy
!
!
!
!
!
!
end


vlan 10 name member-vlan by port
tagged ethe 1/1/1 ethe 1/1/11 ethe 1/1/27
!
!


interface ve 10
ip address 10.10.10.1 255.255.255.0
ip pim-sparse
ip ospf area 0
```

Displaying MCT Information

You can display the following information about MCT configuration and operation:

- Peer and client states
- State machine information
- Cluster, peer, and client states
- MCT-related information for Ethernet interfaces
- STP information

The following example displays the peer device and client states.

```
device# show cluster SXR122 config

cluster SXR122 100
rbridge-id 100
session-vlan 1
keep-alive-vlan 3
icl SXR122-MCT ethernet 1/1/1
peer 172.17.0.2 rbridge-id 101 icl SXR122-MCT
deploy
client KL134
rbridge-id 14
client-interface ethernet 1/1/23
deploy
client AGG131
rbridge-id 10
client-interface ethernet 1/2/2
deploy
client FOX135
rbridge-id 15
client-interface ethernet 1/2/5
deploy
```

The following example displays additional state machine information, including the reason a local CCEP has gone down. You can specify an individual cluster and client as an option.

```
device# show cluster 1 client

Cluster 1 1
===================
Rbridge Id: 101, Session Vlan: 3999, Keep-Alive Vlan: 4001
Cluster State: Deploy
Client Isolation Mode: Loose
Configured Member Vlan Range: 100 to 105
Active Member Vlan Range: 100 to 105
MCT Peer's Reachability status using Keep-Alive Vlan: Peer Reachable
 Client Info:
 ------------
 Client: c1, rbridge-id: 300, Deployed
 Client Port: 1/3/11
 State: Up
 Number of times Local CCEP down: 0
 Number of times Remote CCEP down: 0
 Number of times Remote Client undeployed: 0
 Total CCRR packets sent: 4
 Total CCRR packets received: 3
```

The following table shows the messages that may be displayed to explain why the local CCEP is down.

TABLE 12 Reasons for Local CCEP Down

Message for Local CCEP Down	Meaning
client-isolation strict	Command is configured.

TABLE 12 Reasons for Local CCEP Down (continued)

Message for Local CCEP Down	Meaning
Deploy mismatch	Client is not deployed remotely.
Slave state	Client is in slave state when CCP is down.
cluster and client undeployed	Neither the cluster nor client is deployed.
cluster undeployed	Cluster is not deployed.
client undeployed	Client is not deployed.
client-interfaces shutdown	Command is configured.

The following example displays cluster, peer device, and client states. As an option, you can specify an individual cluster and request additional details.

```
device# show cluster 1 ccp peer

...
PEER IP ADDRESS          STATE           UP TIME
---------------                         -------------    --------------
 10.1.1.1                   OPERATIONAL       0 days: 2 hr:25 min:16 sec

device (config-cluster-SX_1)# show cluster 1 ccp peer detail
**************Peer Session Details*********************
IP address of the peer                          10.1.1.1
Rbridge ID of the peer                          100
Session state of the peer                       OPERATIONAL
Next message ID to be send                      287
Keep Alive interval in seconds                  30
Hold Time Out in seconds                        90
Fast Failover is enable for the session
UP Time                                          0 days: 2 hr:22 min:58 sec
Number of  tcp packet allocations failed           0
Message    Init          Keepalive     Notify        Application   Badmessages
Send       3             2421          2             53            0
Receive    3             2415          0             37            0
TCP connection is up
TCP connection is initiated by                  10.1.1.2
TCP connection tcbHandle not pending
TCP connection  packets not received
**************TCP Connection Details*********************
    TCP Connection state: ESTABLISHED      Maximum segment size: 1436
       Local host:  10.1.1.2, Local  Port: 12203
       Remote host: 10.1.1.1, Remote Port: 4175
       ISentSeq: 1867652277  SendNext: 1867660731  TotUnAck:          0
       TotSent:        8454  ReTrans:           9  UnAckSeq: 1867660731
       IRcvSeq:  3439073167  RcvNext:  3439078415  SendWnd:       16384
       TotalRcv:       5248  DupliRcv:         16  RcvWnd:        16384
       SendQue:           0  RcvQue:            0  CngstWnd:       1452
```

The following example displays information about Ethernet interfaces. The MCT-related information is shown in bold in the following example.

```
device# show interface ethernet 1/7/1

...
GigabitEthernet1/7/1 is disabled, line protocol is down
  Hardware is GigabitEthernet, address is 0024.3822.8260 (bia 0024.3822.8260)
  Configured speed auto, actual unknown, configured duplex fdx, actual unknown
  Configured mdi mode AUTO, actual unknown
  Member of L2 VLAN ID 1, port is untagged, port state is DISABLED
  BPDU guard is Disabled, ROOT protect is Disabled
  Link Error Dampening is Disabled
  STP configured to ON, priority is level0
  Flow Control is config enabled, oper disabled, negotiation disabled
  Mirror disabled, Monitor disabled
  Not member of any active trunks
  Not member of any configured trunks
```

```
  No port name
  IPG MII 96 bits-time, IPG GMII 96 bits-time
  MTU 1500 bytes, encapsulation Ethernet
  ICL port for icl1 in cluster id 1
  300 second input rate: 0 bits/sec, 0 packets/sec, 0.00% utilization
  300 second output rate: 0 bits/sec, 0 packets/sec, 0.00% utilization
  0 packets input, 0 bytes, 0 no buffer
  Received 0 broadcasts, 0 multicasts, 0 unicasts
  0 input errors, 0 CRC, 0 frame, 0 ignored
  0 runts, 0 giants
  0 packets output, 0 bytes, 0 underruns
  Transmitted 0 broadcasts, 0 multicasts, 0 unicasts
  0 output errors, 0 collisions
  Relay Agent Information option: Disabled
show interface ethernet 1/7/3
GigabitEthernet1/7/3 is disabled, line protocol is down
  Hardware is GigabitEthernet, address is 0024.3822.8262 (bia 0024.3822.8262)
  Configured speed auto, actual unknown, configured duplex fdx, actual unknown
  Configured mdi mode AUTO, actual unknown
  Member of L2 VLAN ID 1, port is untagged, port state is DISABLED
  BPDU guard is Disabled, ROOT protect is Disabled
  Link Error Dampening is Disabled
  STP configured to ON, priority is level0
  Flow Control is config enabled, oper disabled, negotiation disabled
  Mirror disabled, Monitor disabled
  Not member of any active trunks
  Not member of any configured trunks
  No port name
  IPG MII 96 bits-time, IPG GMII 96 bits-time
  MTU 1500 bytes, encapsulation Ethernet
  CCEP for client c149_150 in cluster id 1
  300 second input rate: 0 bits/sec, 0 packets/sec, 0.00% utilization
  300 second output rate: 0 bits/sec, 0 packets/sec, 0.00% utilization
  0 packets input, 0 bytes, 0 no buffer
  Received 0 broadcasts, 0 multicasts, 0 unicasts
  0 input errors, 0 CRC, 0 frame, 0 ignored
  0 runts, 0 giants
  0 packets output, 0 bytes, 0 underruns
  Transmitted 0 broadcasts, 0 multicasts, 0 unicasts
  0 output errors, 0 collisions
  Relay Agent Information option: Disabled
```

The following example displays the statistics of MDUP packets.

```
device# show mac-address mdup-stats

MDUP Information
================
MDUP Data buffers in queue : 0
MDUP Statistics
===============
MDUP Update Messages sent: 7
Add Mac sent: 20
Del Mac sent: 0
Move Mac sent: 0
MDUP Mac Info Messages sent: 1
MDUP Flush Messages sent: 1
MDUP Synch Messages sent: 0
MDUP Update Messages received: 3
Add Mac received: 40
Del Mac received: 0
Move Mac received: 0
MDUP Mac Info Messages received: 0
MDUP Flush Messages received: 0
MDUP Synch Messages received: 0
```

The following example displays all local MAC address entries for a cluster.

```
device# show mac-address cluster 1000

Total Cluster Enabled(CL+CR+CCL+CCR) MACs: 1
```

```
Total Cluster Local(CL) MACs: 1
CCL: Cluster Client Local CCR:Cluster Client Remote CL:Local CR:Remote
Total active entries from all ports = 1
Total static entries from all ports = 3
MAC-Address     Port           Type          MCT-Type VLAN
0000.0022.3333  1/2/1            Static      CML       20
0000.0022.3333  1/2/3            Static      CML       20
0000.0022.3333  1/2/13           Static      CML       20
```

MAC Clear Commands

Cluster-specific **clear** commands can be used to remove information about MAC addresses.

To clear all MAC addresses in the system, enter the **clear mac-address** command.

```
device# clear mac-address
```

NOTE
Depending on authentication protocols configuration and high traffic, the **clear mac-address** command log may create a lot of new address messages, which results in high CPU utilization for a few minutes.

To clear cluster-specific MAC addresses in the system, enter the **clear mac-address cluster** command.

```
device# clear mac-address cluster AGG-1 local
```

To clear client-specific MAC addresses in the system, enter the **clear mac-address cluster** command with **client** and **local** options.

```
device# clear mac-address cluster AGG-1 client 1 local
```

To clear VLAN-specific MAC addresses in the system, enter the **clear mac vlan** command.

```
device# clear mac-address vlan 2
```

To clear MCT VLAN-specific MAC addresses in the system, enter the **clear mac-address cluster** command with **vlan** and **local** options.

```
device# clear mac-address cluster AGG-1 vlan 1 local
```

To clear cluster client VLAN-specific MAC addresses in the system, enter the **clear mac-address cluster** command with **vlan**, **client**, and **local** options.

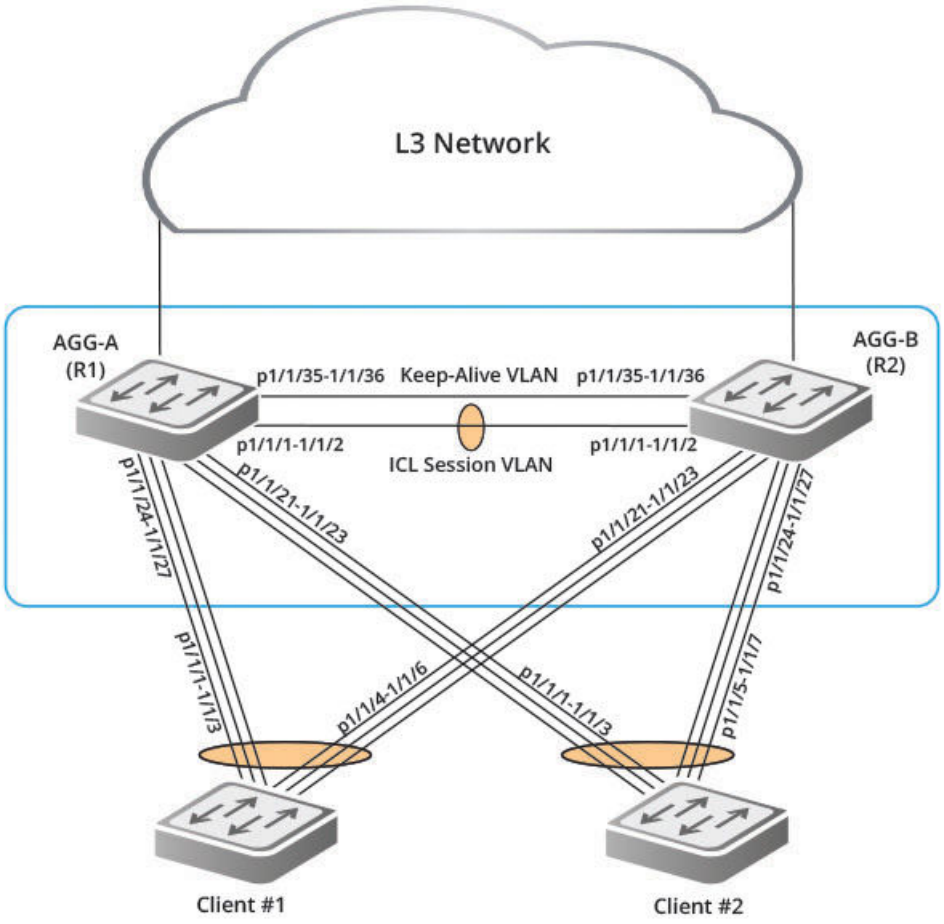
```
device# clear mac-address cluster AGG-1 vlan 2 client 1 local
```

Single-Level MCT Configuration Example

The following figure depicts a single-level MCT configuration. The clients can be server hosts or networking devices. The associated configuration follows.

NOTE
The LAG IDs are locally significant only and need not match on the two ends of a LAG.

FIGURE 47 Single-Level MCT Configuration



Client 1 Configuration

If client 1 is a RUCKUS ICX switch in Figure 47 on page 151, you can use the following configuration.

```
!
lag lag_client1_1 dynamic id 100
ports ethe 1/1/1 to 1/1/6

!
vlan 1905 by port
tagged lag 100
spanning-tree

!
```

Client 2 Configuration

If client 2 is a RUCKUS ICX switch in Figure 47 on page 151, you can use the following configuration.

```
!

lag lag_client2_1 dynamic id 200
 ports ethe 1/1/1 to 1/1/3 ethe 1/1/5 to 1/1/7
```

```
!
vlan 1905 by port
 tagged lag 200
 spanning-tree


!
```

AGG-A (R1) Configuration

The following example presents the configuration for the AGG-A (R1) cluster device in Figure 47 on page 151.

```
lag lag_agg_a_1 static id 103
 ports ethe 1/1/1 to 1/1/2

!
lag lag_agg_a_2 dynamic id 104
 ports ethe 1/1/24 to 1/1/27

!
lag lag_agg_a_3 dynamic id 105
 ports ethe 1/1/21 to 1/1/23

!
vlan 2 name session-vlan by port
 tagged lag 103
 !
vlan 3 name keep-alive-vlan by port
 tagged ethe 1/1/35-1/1/36
 !
vlan 1905 by port
 tagged lag 103 to 105
!
hostname R1
!
interface ve 2
 ip address 10.1.1.1 255.255.255.0
!
interface ve 3
 ip address 10.1.2.1 255.255.255.0
!
!
cluster MCT1 1
 rbridge-id 1
 session-vlan 2
 keep-alive-vlan 3
 icl BH1 lag 103
 peer 10.1.1.2 rbridge-id 2 icl BH1
 deploy
 client client-1
  rbridge-id 1901
  client-interface lag 104
  deploy
 client client-2
  rbridge-id 1902
  client-interface lag 105
  deploy
!
```

AGG-B (R2) Configuration

The following example presents the configuration for the AGG-B (R2) cluster device in Figure 47 on page 151.

```
lag lag_agg_b_1 static id 103
 ports ethe 1/1/1 to 1/1/2
```

```
!
lag lag_agg_b_2 dynamic id 105
 ports ethe 1/1/24 to 1/1/27

!
lag lag_agg_b_3 dynamic id 104
 ports ethe 1/1/21 to 1/1/23

!
vlan 2 name session-vlan by port
 tagged lag 103
!
vlan 3 by port
 tagged ethe 1/1/35-1/1/36
!
!
vlan 1905 by port
 tagged lag 103 to 105
!
hostname R2
!
interface ve 2
 ip address 10.1.1.2 255.255.255.0
!
interface ve 3
 ip address 10.1.2.2 255.255.255.0
!
cluster MCT1 1
 rbridge-id 2
 session-vlan 2
 keep-alive-vlan 3
 icl BH1 lag 103
 peer 10.1.1.1 rbridge-id 1 icl BH1
 deploy
 client client-1
  rbridge-id 1901
  client-interface lag 104
  deploy
 client client-2
  rbridge-id 1902
  client-interface lag 105
!
```

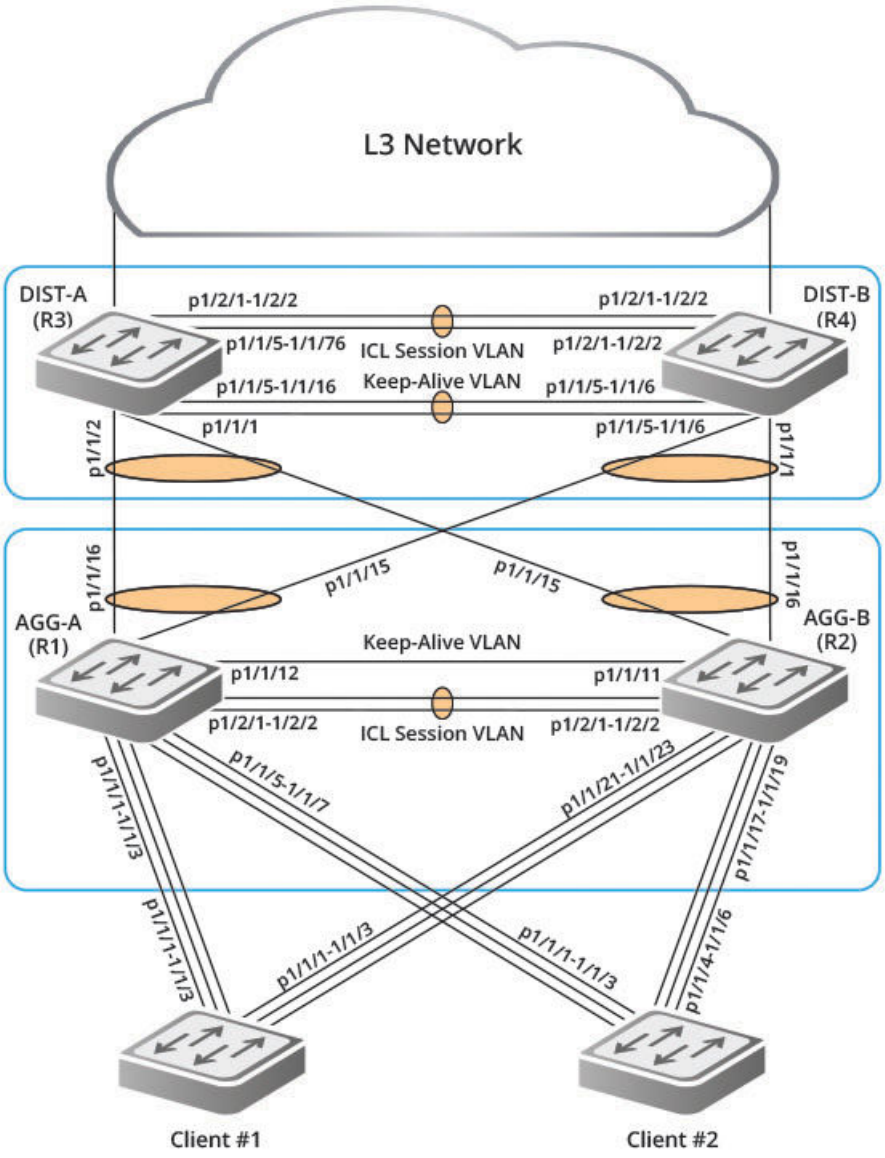
Two-Level MCT Configuration Example

The following figure depicts a two-level MCT configuration. The clients can be server hosts or networking devices. The associated configuration follows.

NOTE

The LAG IDs are locally significant only and need not match on the two ends of a LAG.

FIGURE 48 Two-Level MCT Configuration



NOTE

In a two-level MCT configuration using dynamic LAGs, ensure that the upper and lower clusters have different Cluster IDs because the Cluster LACP module uses the Cluster ID as part of the LACPDU system ID.

The client configuration is the same as in the single-level example (refer to Single-Level MCT Configuration Example on page 150).

AGG-A (R1) Configuration

The following example presents the configuration for the AGG-A (R1) cluster device in Figure 48 on page 154.

```
lag lag_agg_a_1 static id 103
 ports ethe 1/2/1 to 1/2/2
```

```
!
lag lag_agg_a_2 dynamic id 104
 ports ethe 1/1/1 to 1/1/3

!
lag lag_agg_a_3 dynamic id 105
 ports ethe 1/1/5 to 1/1/7

!
lag lag_agg_a_4 dynamic id 106
 ports ethe 1/1/15 to 1/1/16

!
vlan 2 name session-vlan by port
 tagged lag 103
!
vlan 3 name keep-alive-vlan by port
 tagged ethe 1/1/12
!
!
vlan 1905 by port
 tagged lag 103 to 106
!
hostname R1
!
interface ve 2
 ip address 10.1.1.1 255.255.255.0
!
interface ve 3
 ip address 10.1.2.1 255.255.255.0
!
!
cluster MCT1 1
 rbridge-id 1
 session-vlan 2
 keep-alive-vlan 3
 icl BH1 lag 103
 peer 10.1.1.2 rbridge-id 2 icl BH1
 deploy
 client client-1
  rbridge-id 1901
  client-interface lag 104
  deploy
 client client-2
  rbridge-id 1902
  client-interface lag 105
  deploy
 client DIST_Cluster
  rbridge-id 1903
  client-interface lag 106
  deploy
!
```

AGG-B (R2) Configuration

The following example presents the configuration for the AGG-B (R2) cluster device in Figure 48 on page 154.

```
lag lag_agg_b_1 static id 106
 ports ethe 1/2/1 to 1/2/2

!
lag lag_agg_b_2 dynamic id 107
 ports ethe 1/1/17 to 1/1/19

!
lag lag_agg_b_3 dynamic id 108
 ports ethe 1/1/21 to 1/1/23

!
```

```
lag lag_agg_b_4 dynamic id 109
 ports ethe 1/1/15 to 1/1/16

!
vlan 2 name session-vlan by port
 tagged lag 106
!
vlan 3 name keep-alive-vlan by port
 tagged ethe 1/1/11
!
!
vlan 1905 by port
 tagged lag 106 to 109
!
hostname R2
!
interface ve 2
 ip address 10.1.1.2 255.255.255.0
!
interface ve 3
 ip address 10.1.2.2 255.255.255.0
!
cluster MCT1 1
 rbridge-id 2
 session-vlan 2
 keep-alive-vlan 3
 icl BH1 lag 106
 peer 10.1.1.1 rbridge-id 1 icl BH1
 deploy
 client client-1
  rbridge-id 1901
  client-interface lag 108
  deploy
 client client-2
  rbridge-id 1902
  client-interface lag 107
  deploy
 client DIST_Cluster
  rbridge-id 1903
  client-interface lag 109
  deploy
 !
```

DIST-A (R3) Configuration

The following example presents the configuration for the DIST-A (R3) cluster device in Figure 48 on page 154.

```
!
lag lag_dist_a_1 static id 15
 ports ethe 1/2/1 to 1/2/2

lag lag_dist_a_2 dynamic id 16
 ports ethe 1/1/1 to 1/1/2

!
lag keep-alive static id 200
ports ether 1/1/5 to 1/1/16

!
vlan 5 name session-vlan by port
 tagged lag 15
 interface ve 5
!
vlan 6 name keep-alive-vlan by port
 tagged ethe 1/1/5 to 1/1/16
 interface ve 6
 spanning-tree
!
vlan 1905 by port
```

```
 tagged lag 15 to 16
!
hostname R3
hitless-failover enable
!
interface ve 5
 ip address 10.2.1.1 255.255.255.0
!
interface ve 6
 ip address 10.2.2.1 255.255.255.0
!
cluster MCT2 2
 rbridge-id 3
 session-vlan 5
 keep-alive-vlan 6
 icl BH3 lag 15
 peer 10.2.1.2 rbridge-id 4 icl BH3
 deploy
 client AGG_Cluster
rbridge-id 1801
client-interface lag 16
deploy
```

DIST-B (R4) Configuration

The following example presents the configuration for the DIST-B (R4) cluster device in Figure 48 on page 154.

```
lag lag_dist_b_1 static id 40
 ports ethe 1/2/1 to 1/2/2

!
lag dist_b_2 dynamic id 41
ports ethe 1/1/1 to 1/1/2

!
lag keep-alive static id 201
ports ethe 1/1/5 to 1/1/16

!
vlan 5 name session-vlan by port
 tagged lag 40
 interface ve 5
!
vlan 6 name keep-alive-vlan by port
 tagged ethe 1/1/5 to 1/1/16
 interface ve 6
 spanning-tree
!
vlan 1905 by port
 tagged lag 40 to 41
!
hostname R4
hitless-failover enable
!
interface ve 5
 ip address 10.2.1.2 255.255.255.0
!
interface ve 6
 ip address 10.2.2.2 255.255.255.0
!
cluster MCT2 2
 rbridge-id 4
 session-vlan 5
 keep-alive-vlan 6
 icl BH3 lag 40
 peer 10.2.1.1 rbridge-id 3 icl BH3
 deploy
client AGG_Cluster
rbridge-id 1801
```

```
client-interface lag 41
deploy
```

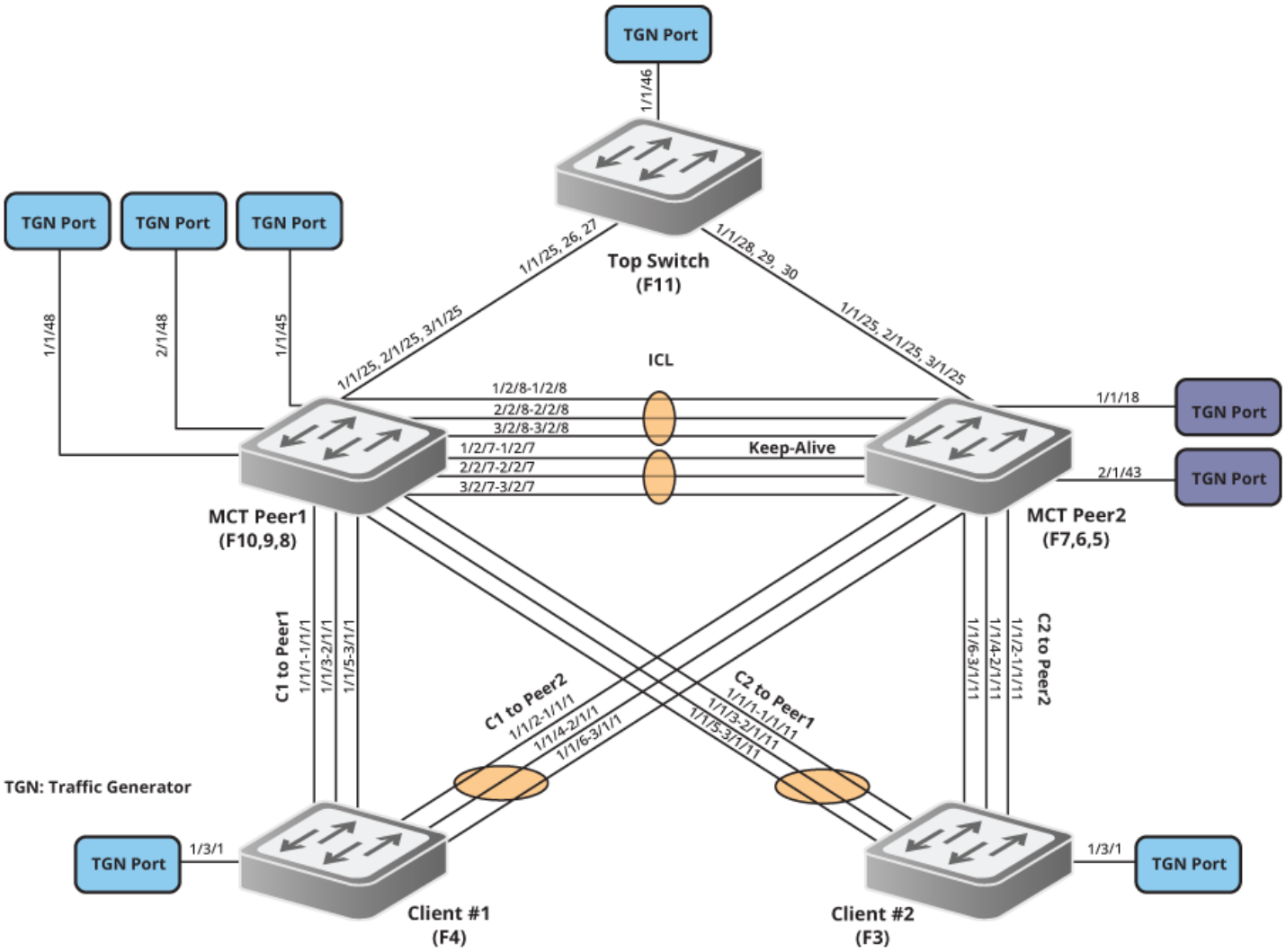
MCT Configuration Example Using ICX 7850 Stacks as Cluster Peers

NOTE

This example is using an ICX 7850 stack, but is applicable to all stackable ICX devices.

The following figure depicts an MCT configuration with ICX stacks used as cluster peers. The related configuration is included following the figure.

FIGURE 49 MCT-Stack Configuration



Peer 1 Configuration

Cluster Configuration

```
P1(config)# show cluster 1000 config
cluster mct_mm_stk 1000
rbridge-id 1019
session-vlan 1000
keep-alive-vlan 1999
icl-fwd-delay 10
icl icl_mm lag 1
peer 11.1.1.2 rbridge-id 1029 icl icl_mm
peer 11.1.1.2 timers keep-alive 30 hold-time 90
deploy

client C1
rbridge-id 11
client-interface lag 11
deploy

client C2
rbridge-id 12
client-interface lag 12
deploy

### Lag interfaces  ######
LAG            Type    Deploy Trunk Intf     Port List     Logical-block
icl_mm         static   Y    1     lg1      e 1/2/8 e 2/2/8 e 3/2/8    >>>>>>>>>>>>>ICL
ka             dynamic  Y    2     lg2      e 1/2/7 e 2/2/7 e 3/2/7    >>>>>>>>>>>>>>>>keepalive link
11             static   Y    11    lg3      e 1/1/1 e 2/1/1 e 3/1/1    >>> lag to client-1
12             dynamic  Y    12     lg4      e 1/1/11 e 2/1/11 e 3/1/11 >>> lag to client-2
p1_uplink      static   Y    100   lg100    e 1/1/25 e 2/1/25 e 3/1/25  >>> lag to uplink switch

### vlans ###
vlan 3001 by port  >>>>> mct vlan
tagged ethe 1/1/48 lag 1 lag 11 to 12 lag 100
!
vlan 1000 by port  >>>>>> session vlan
tagged lag 1
!
vlan 1999 by port  >>>>>> keep-alive vlan
tagged lag 2
```

Peer 2 Configuration

```
P2# show cluster 1000 config
cluster mct_mm_stk 1000
rbridge-id 1029
session-vlan 1000
keep-alive-vlan 1999
icl icl_mm lag 1
peer 11.1.1.1 rbridge-id 1019 icl icl_mm
peer 11.1.1.1 timers keep-alive 30 hold-time 90
deploy

client C1
rbridge-id 11
client-interface lag 3
deploy

client C2
rbridge-id 12
client-interface lag 4
deploy
!

#### Lag configs ####
```

```
LAG             Type    Deploy Trunk Intf     Port List     Logical-block
icl_mm          static    Y    1     lg1      e 1/2/8 e 2/2/8 e 3/2/8
ka              dynamic   Y    2     lg2      e 1/2/7 e 2/2/7 e 3/2/7
11              static    Y    11    lg11     e 1/1/1 e 2/1/1 e 3/1/1
12              dynamic   Y    12    lg12     e 1/1/11 e 2/1/11 e 3/1/11
p2_uplink       dynamic   Y    200   lg200    e 1/1/25 e 2/1/25 e 3/1/25

### vlan configs ###
vlan 3001 by port
tagged ethe 1/1/48 lag 1 lag 11 to 12 lag 200
!
vlan 1000 by port
tagged lag 1
!
vlan 1999 by port
tagged lag 2
```

Client 1 Configuration

```
### vlan config ###
vlan 3001 by port
tagged ethe 1/3/1 lag 11
spanning-tree 802-1w
!
### lag config ###
LAG             Type    Deploy Trunk Intf     Port List     Logical-block
c11             static    Y    11    lg11     e 1/1/1 to 1/1/6
```

Client 2 Configuration

```
### vlan configs ###
vlan 3001 by port
tagged ethe 1/3/1 lag 12
spanning-tree 802-1w
!
### lag interfaces ###
LAG             Type    Deploy Trunk Intf     Port List     Logical-block
c12             dynamic   Y    12    lg12     e 1/1/1 to 1/1/6
```

Top Switch Configuration

```
###  vlan config ###
vlan 3001 by port
tagged ethe 1/1/48 lag 100 lag 200
spanning-tree 802-1w
!
### lag config ###

LAG             Type    Deploy Trunk Intf     Port List     Logical-block
p1_uplink       static    Y    100   lg100    e 1/1/25 to 1/1/27
uplink_p2       dynamic   Y    200   lg200    e 1/1/28 to 1/1/30
```

MCT Hitless Sequential Upgrade

NOTE

MCT hitless upgrade is not supported for upgrades from releases prior to FastIron 09.0.10a.

Prior to FastIron 09.0.10a, devices with different software versions were not compatible to form an MCT cluster. This posed a limitation for MCT peers with different software versions to provide the normal forwarding behavior during a software upgrade. To overcome this limitation and

achieve MCT hitless upgrade, the MCT protocol version check is introduced in FastIron 09.0.10a. The MCT protocol version check replaces the existing FastIron software release or build string compatibility check. The new solution allows two units running FastIron 09.0.10a and later releases to form MCT peers, as long as their MCT protocol versions match and both the peers are from the same model family. In addition to checking the MCT protocol version, the existing platform check will be performed too.

The MCT protocol version will be incremented whenever there is any enhancement or change in the MCT messages that are exchanged between the MCT peers. The MCT version starts from version 2, and the version number is displayed in the **show cluster** command output. The MCT hitless upgrade will keep CCP down when the MCT protocol version check fails.

MCT protocol version can be checked on cluster peers using the **show cluster** command output on 09.0.10a or higher releases. The following example of the **show cluster** command output indicates the MCT version is version 2.

```
device# show cluster
Cluster mct_mm_stk 1000, version 2   >>>>>> version 2 represents mct protocol version 2 here.
====================
Rbridge Id: 1019, Session Vlan: 1000, Keep-Alive Vlan: 1999
Cluster State: Deploy
Client Isolation Mode: Loose
Member Vlan Range: 1001 to 1002 2406 to 2407 3001 to 3150
MCT Peer's Reachability using Keep-Alive Vlan: Peer Reachable

ICL Info:
---------
Name            Port       Trunk
icl_mm          lg1        1

Peer Info:
----------
Peer IP: 11.1.1.2, Peer Rbridge Id: 1029, ICL: icl_mm
KeepAlive Interval: 30 , Hold Time: 90, Fast Failover
Active Vlan Range:  1001 to 1002 2406 to 2407 3001 to 3150
 Last Reason for CCP Down: ICL interface down
Peer State: CCP Up (Up Time:   0 days:10 hr:44 min:39 sec)

Client Info:   Config CCEP Up Delay 300, Oper CCEP Up Delay 0
------------
Number of Clients configured: 44
Name                     Rbridge-id Config      Port       Trunk FSM-State
 C1                       3          Deployed    lg3          3    Up
 C2                       4          Deployed    lg4          4    Up
```

In-Service Software Upgrade (ISSU) between minor releases is used for image upgrade of stacking units within the same cluster node. Stack units within the cluster node must be upgraded one after the other to minimize traffic disruption. Refer to the *RUCKUS FastIron Software Upgrade Guide* for more information on ISSU and upgrading a stack.

MVRP

MVRP Overview

In a large bridged local area network with many interconnected switches, manual management of VLANs is complicated and prone to human error. Multiple VLAN Registration Protocol (MVRP) is a Multiple Registration Protocol (MRP) application that helps to create VLANs dynamically (VLAN registration) and automate the administration of VLAN membership (distribution and deregistration) within the network without manual intervention.

MVRP provides IEEE 802.1ak-compliant VLAN pruning and dynamic VLAN creation on switch ports connecting access and core switches. An MVRP-aware switch can exchange VLAN configuration information with other MVRP-aware switches, prune unnecessary broadcast and unknown unicast traffic, and dynamically create and manage VLANs on switches. MVRP allows the propagation of VLAN information from device to device. With MVRP, an access switch can be manually configured with all the desired VLANs for the network, and all other MVRP-enabled switches on the network learn those VLANs dynamically. The network administrator does not have to manually configure the VLANs in each of the devices in the topology. When the VLAN configurations on a switch are changed, MVRP automatically changes the VLAN configurations in the required switches.

MVRP maintains the following advantages:

- Reduces the broadcast, unknown-unicast, and multicast traffic (BUM traffic) scope to the interested devices or switches in the network. If there are no active users for a VLAN at a remote destination, then traffic is dropped at source itself thereby efficiently utilizing the network bandwidth.
- Reduces the chances of errors in VLAN configuration by automatically providing VLAN ID consistency across the network. In addition to this, if the VLAN configuration on a switch changes, MVRP automatically changes the VLAN configurations in the affected devices.

MVRP allows bridges in a bridged LAN to issue and revoke declarations of a VLAN attribute.

- When a port receives a VLAN attribute declaration message, it joins the VLAN and propagates that VLAN declaration on other MVRP ports.
- When a port receives a VLAN attribute withdrawal message, it leaves the VLAN and propagates that VLAN withdrawal on other MVRP ports.

Beginning with FastIron 08.0.90, GARP VLAN Registration Protocol (GVRP) is not supported on ICX platforms. MVRP is a successor to GVRP, and RUCKUS recommends using MVRP which is more efficient than GVRP due to the protocol design. MVRP is not backward-compatible with GVRP and therefore will not interoperate with GVRP.

MRP Messages Used by MVRP

MVRP uses MRP messages to issue or withdraw declaration of VLANs to other MVRP-aware switches. The following MRP messages are used by MVRP:

- Empty: MVRP information (VLANs) is not declared and not registered.
- In: MVRP information is not declared but the VLAN is registered.
- JoinEmpty: MVRP information is declared but the VLAN is not registered.
- JoinIn: MVRP information is declared and the VLAN is registered.
- Leave: MVRP information that was previously declared is now withdrawn.
- LeaveAll: All the registered VLANs are unregistered and the VLANs need to be reregistered.
- New: The MVRP information is new and the VLAN may not be registered yet.

The JoinIn, JoinEmpty, and New messages are declarations, while Leave and LeaveAll messages are withdrawals.

MVRP Timers

MVRP uses MRP messages to make or withdraw declaration of VLANs to other MVRP-aware switches. MVRP timers define the interval at which MVRP updates (VLAN join or VLAN leave messages) are transmitted. MVRP global timers are enabled by default. Global timer configurations reflect on all MVRP-enabled ports in the system. MVRP timers are per interface level and apply only to the specified interface. MVRP timers must be set to the same values on all the devices that are participating in MVRP. Interface-level timer configuration takes precedence over global timer configurations.

MVRP has the following timers that determine the interval for VLAN declaration and withdrawal events:

- Join timer: Defines the interval for the MVRP PDU transmit that makes VLAN declaration on other MVRP-enabled interfaces.
- Leave timer: Defines the time period an MVRP-enabled interface waits after receiving a leave message on the port to remove the port from the VLAN indicated in the leave message. If the interface receives a VLAN join message before the timer expires, the VLAN remains registered.
- Leave-all timer: Defines the time interval at which a port (MVRP participant) generates LeaveAll PDUs.

MVRP Registration Modes

There are two registration modes that define the VLAN participation in MVRP:

- Normal: In this mode, the interface accepts all MVRP messages and participates in MVRP. This is the default registration mode setting.
- Forbidden: The registration mode of MVRP can be set to forbid the VLAN from participating in MVRP. If the VLAN is added to the forbidden VLAN list using the **mvrp registration-mode forbidden** command, MVRP neither declares nor registers any VLANs on the port. All the registration messages received for those VLANs are ignored. By default, the registration mode is set to Normal, which allows the VLAN to be learned or declared through MVRP.

  NOTE
  Whenever an access port is added to a VLAN or whenever MVRP ports are statically tagged to the VLAN, the registration mode of all the MVRP ports is automatically set to Fixed for that VLAN.

You can also set the applicant state of the port that defines MVRP participation of the port. If a port is configured as non-participant, MVRP PDU transmission will be prohibited on the port. By default, the applicant mode of the port is a normal participant, which allows VLAN registration and propagation of VLAN information.

MVRP with Per-VLAN STP and Per-VLAN RSTP

Beginning with FastIron 08.0.95, MVRP is enhanced to support Per-VLAN Spanning Tree (PVST). By default, all the dynamic VLANs created by MVRP run spanning tree instances.

PVST is enabled by default on MVRP untagged VLANs. You can use the **mvrp spanning-tree** command to choose the type of spanning tree to be added to the dynamic VLANs.

MVRP messages are propagated only to the ports in the active topology which comprise a set of forwarding ports in the MVRP untagged VLAN. MVRP does not support multiple dynamic VLANs to converge with different active spanning-tree topologies. The new VLANs that are learned dynamically will have the same Layer 2 data path. If you want a different path to be established for a particular PVST instance, you must manually configure the VLANs and add ports and configure the appropriate spanning tree parameters.

If you do not want spanning tree instances enabled by default on dynamic VLANs, you must remove the MVRP configuration on all interfaces, configure the **no mvrp spanning-tree** command, and then reconfigure MVRP on those interfaces.

MVRP Untagged VLAN

The MVRP untagged VLAN is the first port's untagged VLAN. All other MVRP ports added afterwards must follow the same untagged VLAN membership. The MVRP untagged VLAN is the VLAN in which all MVRP-enabled ports are present.

All the MVRP-enabled ports must have the same untagged VLAN membership. Consequently, the untagged VLAN membership of MVRP-enabled ports cannot be changed during MVRP operation. Therefore, the following operations issue an error message:

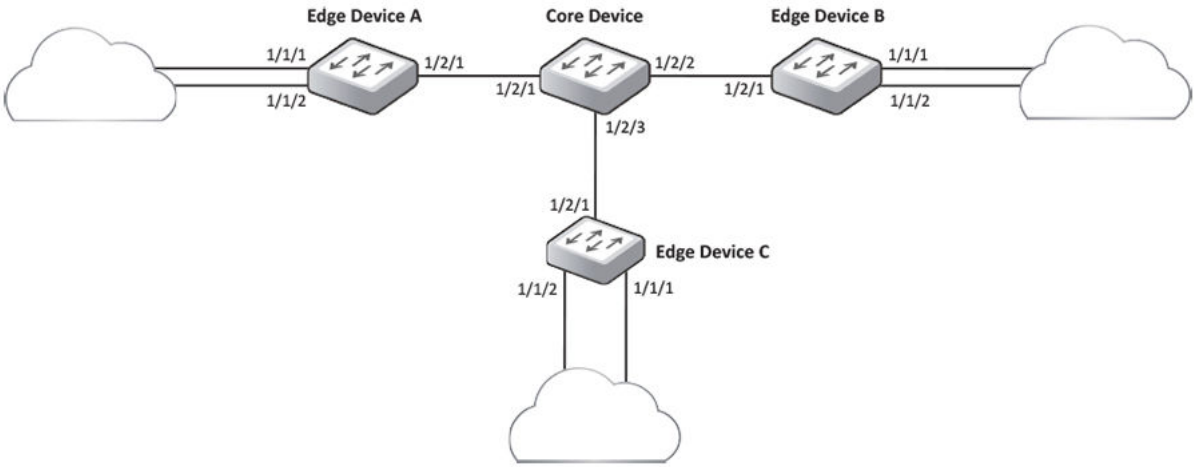
- Configuring an MVRP-enabled untagged port on a different VLAN
- Removing an MVRP untagged VLAN
- Enabling MVRP on a port with untagged membership of a different VLAN

NOTE
It is recommended to configure Per-VLAN STP/RSTP on the MVRP untagged VLAN prior to MVRP configuration on interfaces to avoid unnecessary VLAN registrations.

MVRP Application Example

The following figure shows an example of a network that uses MVRP. This section describes one of the ways you can use MVRP in a network. Configuration Example: Implementing the Applications of MVRP lists the commands and configuration steps to implement the applications of MVRP described in the following example.

FIGURE 50 Example of MVRP



In this example, a core device is attached to three edge devices. Each of the edge devices is attached to other edge devices or host stations (represented by the clouds). The effects of MVRP in this network depend on which devices the feature is enabled.

In this configuration, all ports on the core device are enabled to learn and advertise VLAN information. The edge devices are configured to advertise their VLAN configurations on the ports connected to the core device.

TABLE 13

Core Device	Edge Device A	Edge Device B	Edge Device C
MVRP is enabled on all ports.	• MVRP is enabled on port 1/2/1. • VLAN 20 • Port 1/1/1 (untagged) • Port 1/2/1 (auto-added to VLAN 20) • VLAN 40 • Port 1/1/2 (untagged) • Port 1/2/1 (auto-added to VLAN 40)	• MVRP is enabled on port 1/2/1. • VLAN 20 • Port 1/1/1 (untagged) • Port 1/2/2 (auto-added to VLAN 20) • VLAN 30 • Port 1/1/2 (untagged) • Port 1/2/2 (auto-added to VLAN 30)	• MVRP is enabled on port 1/2/1. • VLAN 30 • Port 1/1/1 (untagged) • Port 1/2/3 (auto-added to VLAN 30) • VLAN 40 • Port 1/1/2 (untagged) • Port 1/2/3 (auto-added to VLAN 30)

In this configuration, the edge devices are statically (manually) configured with VLAN information. The core device dynamically configures itself to be a member of each of the edge device VLANs. The operation of MVRP on the core device results in the following VLAN configuration on the device:

- VLAN 20
  - 1/2/1
  - 1/2/2
- VLAN 30
  - 1/2/2
  - 1/2/3
- VLAN 40
  - 1/2/1
  - 1/2/3

VLAN 20 traffic can now travel through the core between edge devices A and B. Likewise, VLAN 30 traffic can travel between B and C and VLAN 40 traffic can travel between A and C. If an edge device is moved to a different core port or the VLAN configuration of an edge device is changed, the core device automatically reconfigures itself to accommodate the change.

Notice that each of the ports in the dynamically created VLANs is tagged. All MVRP VLAN ports configured by MVRP are tagged, to ensure that the port can be configured for additional VLANs.

NOTE
VLAN configuration learned by MVRP does not show up in the running-config. It can be viewed in the output of the **show vlan** commands.

NOTE
This example assumes that the core device has no static VLANs configured. However, you can have static VLANs on a device that is running MVRP. MVRP can dynamically add other ports to the statically configured VLANs but cannot delete statically configured ports from the VLANs.

Configuration Example: Implementing the Applications of MVRP

This section provides the sample configuration steps to implement the MVRP application example.

NOTE
Although some of the devices in these configuration examples do not have statically configured VLANs, this is not a requirement. You always can have statically configured VLANs on a device that is running MVRP.

In this configuration, the edge devices advertise their statically configured VLANs to the core device. The core device does not have any statically configured VLANs but learns the VLANs from the edge devices.

Enter the following commands on the core device.

```
core-device(config)# mvrp enable
core-device(config)# interface ethernet 1/2/1 to 1/2/3
core-device(config-mif-1/2/1-1/2/3)# mvrp enable

core-device(config-mif-1/2/1-1/2/3)# mvrp point-to-point
```

The **mvrp enable** command enables the MVRP functionality globally.

Enter the following commands on edge device A.

```
deviceA(config)# vlan 20
deviceA(config-vlan-20)# untagged ethernet 1/1/1
Added untagged port(s) ethe 1/1/1 to port-vlan 20.

deviceA(config)# vlan 40
deviceA(config-vlan-40)# untagged ethernet 1/1/2
Added untagged port(s) ethe 1/1/2 to port-vlan 40.

deviceA(config)# mvrp enable

deviceA(config)# interface ethernet 1/2/1
deviceA(config-if-e40000-1/2/1)# mvrp enable

deviceA(config-if-e40000-1/2/1)# mvrp point-to-point
deviceA(config-if-e40000-1/2/1)# exit
```

These commands statically configure two port-based VLANs and enable MVRP on port 1/2/1.

Enter the following commands on edge device B.

```
deviceB(config)# vlan 20
deviceB(config-vlan-20)# untagged ethernet 1/1/1
Added untagged port(s) ethe 1/1/1 to port-vlan 20.

deviceB(config)# vlan 30
deviceB(config-vlan-30)# untagged ethernet 1/1/2
Added untagged port(s) ethe 1/1/2 to port-vlan 30.

deviceB(config)# mvrp enable

deviceB(config)# interface ethernet 1/2/1
deviceB(config-if-e40000-1/2/1)# mvrp enable

deviceB(config-if-e40000-1/2/1)# mvrp point-to-point
deviceB(config-if-e40000-1/2/1)# exit
```

Enter the following commands on edge device C.

```
deviceC(config)# vlan 30
deviceC(config-vlan-30)# untagged ethernet 1/1/1
Added untagged port(s) ethe 1/1/1 to port-vlan 30.

deviceC(config)# vlan 40
deviceC(config-vlan-40)# untagged ethernet 1/1/2
Added untagged port(s) ethe 1/1/2 to port-vlan 40.


deviceC(config)# mvrp enable

deviceC(config)# interface ethernet 1/2/1
deviceC(config-if-e40000-1/2/1)# mvrp enable

deviceC(config-if-e40000-1/2/1)# mvrp point-to-point
deviceC(config-if-e40000-1/2/1)# exit
```

MVRP Configuration Notes

- MVRP is not supported with MSTP. MVRP is only supported with single 802.1w, Single STP (SSTP), PVST, and Per-VLAN RSTP.
- All dynamic VLANs created by the MVRP run STP by default.
- The default VLAN cannot be changed if the system is MVRP-enabled. Therefore, it is recommended that you change the default VLAN (optional) prior to enabling MVRP.
- The default VLAN is always in FIXED registration. Therefore, it is recommended to retain the MVRP ports in the system default VLAN and maintain the default VLAN consistency in the network.
- MVRP is not allowed on a Management port.
- VLANs that are learned dynamically cannot be deleted manually. Static VLANs cannot be deleted manually if they have dynamic ports learned by MVRP.
- Ports that are added dynamically cannot be removed manually from a dynamic or static VLAN.
- Static VLANs cannot be configured as forbidden VLANs.
- Dynamic VLAN creation by MVRP is enabled by default. If you choose to change the default behavior using the **mvrp vlan-creation-disable** command, it is recommended that you do so prior to enabling MVRP.
- Dynamic VLANs created by MVRP can be converted to static VLANs using standard VLAN commands. Similarly, MVRP-added ports can be converted to static members by retagging the ports using VLAN port-tagging commands.
- VLANs created by MVRP do not support virtual routing interfaces. However, virtual routing interfaces are supported on statically configured VLANs even if MVRP adds ports to those VLANs.
- An MVRP periodic timer is not supported.
- Dynamic VLANs are not saved to the startup configuration and therefore will not persist across a reload. However, once the system is up, the protocol will relearn the dynamic VLANs.
- MVRP is supported on physical ports and LAG ports. However, an MVRP-enabled port cannot be added as a secondary port of a LAG.
- VLAN groups cannot be configured when MVRP is enabled.
- Maximum of 128 dynamic VLANs having Per-VLAN STP or Per-VLAN RSTP instances are supported.

MVRP Limitations

MVRP is not supported with the following features and therefore cannot coexist with the features:

- Per-VLAN protocols (MRP and VSRP)
- MSTP
- MCT
- Q-in-Q (MVRP cannot be configured on Q-in-Q-enabled ports.)
- PVLAN
- RSPAN
- Topology group
- VLAN group
- BPDU tunneling
- MAC ACLs
- MVRP and Flexible authentication cannot be enabled on the same port.
- MVRP dynamic VLANs cannot be configured as authentication VLANs.

Configuring MVRP

The following configuration steps show basic MVRP configuration in a network topology that has two devices.

1. Enable MVRP at the system level on the devices.

```
device1(config)# mvrp enable
device2(config)# mvrp enable
```

   MVRP must be enabled globally to allow the devices to participate in the protocol.

2. Enable MVRP on uplink ports on both devices.

```
device1(config)# interface ethernet 1/1/2
device1(config-if-e10000-1/1/2)# mvrp enable

device2(config)# interface ethernet 2/1/2
device2(config-if-e10000-2/1/2)# mvrp enable
device2(config)# interface ethernet 1/1/10
device2(config-if-e10000-1/1/10)# mvrp enable
device2(config-if-e10000-2/1/2)# interface lag 1
device2(config-lag-if-lg1)# mvrp enable
```

3. Create a user VLAN on device 1.

```
device1(config)# vlan 10
```

4. Add an access port to the VLAN.

```
device1(config-vlan-10)# untagged ethernet 1/1/48
```

   MVRP-enabled uplink port (1/1/2) will be auto-added to the user VLAN 10.

   Device 1 declares VLAN 10 and device 2 dynamically learns it and adds the port (2/1/2) as a dynamic member.

5. (Optional) Configure MVRP timers globally.

```
device1(config)# mvrp timer join 250 leave 1200 leave-all 12000
```

   MVRP global timers are enabled by default. Global timer configurations reflect on all MVRP-enabled ports. MVRP timers are per interface level and apply only to the specified interface. Interface-level timer configuration takes precedence over global timer configurations.

6. (Optional) Configure a port as a point-to-point interface for MVRP.

```
device1(config)# interface ethernet 1/1/2
device1(config-if-e10000-1/1/2)# mvrp point-to-point

device2(config)# interface ethernet 2/1/2
device2(config-if-e10000-2/1/2)# mvrp point-to-point
```

7. (Optional) Configure the applicant state of the port that defines the MVRP participation of the port.

```
device2(config)# interface ethernet 1/1/10
device2(config-if-e10000-1/1/10)# mvrp applicant-mode non-participant
```

   By default, the applicant mode of the port is normal.

8. (Optional) Configure registration mode to forbid the VLAN from participating in MVRP.

   By default, registration mode is set to Normal which allows the VLAN to be learned or declared through MVRP. For a static VLAN configuration, registration mode is automatically set to Fixed.

```
device2(config)# interface lag 1
device2(config-lag-if-lg1)# mvrp registration-mode forbidden vlan 10
```

9. (Optional) Disable dynamic VLAN creation by MVRP unless the VLAN that is being learned is already defined in the device.

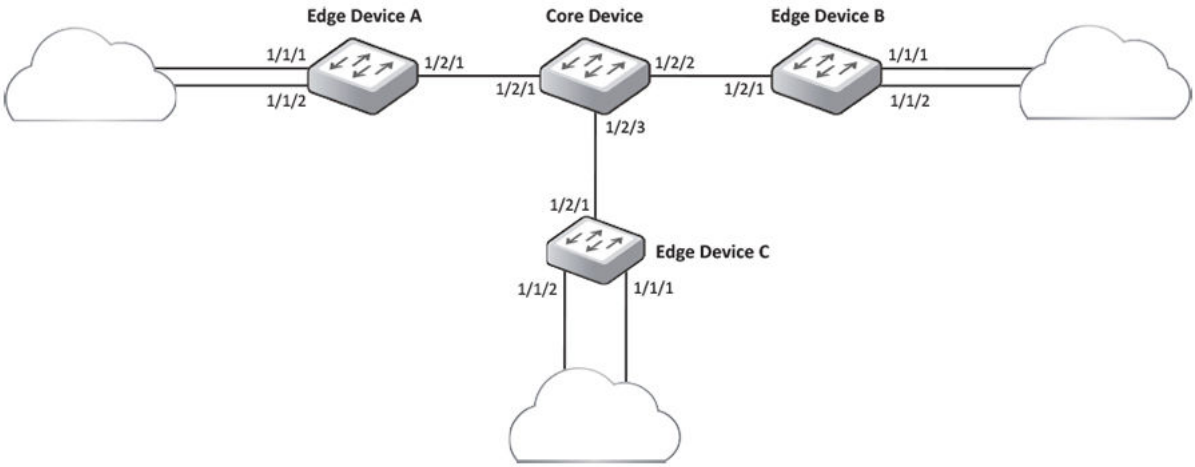
```
device2(config)# mvrp vlan-creation-disable
```

Configuration Example: Implementing the Applications of MVRP

In the following example, a core device is attached to three edge devices. Each of the edge devices is attached to other edge devices or host stations (represented by the clouds).

The effects of MVRP in this network depend on which devices the feature is enabled.

FIGURE 51 MVRP Network



NOTE
Although some of the devices in these configuration examples do not have statically configured VLANs, this is not a requirement. You always can have statically configured VLANs on a device that is running MVRP.

In this configuration, the edge devices advertise their statically configured VLANs to the core device. The core device does not have any statically configured VLANs but learns the VLANs from the edge devices.

Enter the following commands on the core device.

```
core-device(config)# mvrp enable
core-device(config)# interface ethernet 1/2/1 to 1/2/3
core-device(config-mif-1/2/1-1/2/3)# mvrp enable

core-device(config-mif-1/2/1-1/2/3)# mvrp point-to-point
```

The **mvrp enable** command enables the MVRP functionality globally.

Enter the following commands on edge device A.

```
deviceA(config)#vlan 20
deviceA(config-vlan-20)# untagged ethernet 1/1/1
Added untagged port(s) ethe 1/1/1 to port-vlan 20.

deviceA(config)#vlan 40
deviceA(config-vlan-40)# untagged ethernet 1/1/2
Added untagged port(s) ethe 1/1/2 to port-vlan 40.

deviceA(config)# mvrp enable
```

```
deviceA(config)# interface ethernet 1/2/1
deviceA(config-if-e40000-1/2/1)# mvrp enable

deviceA(config-if-e40000-1/2/1)# mvrp point-to-point
deviceA(config-if-e40000-1/2/1)# exit
```

These commands statically configure two port-based VLANs and enable MVRP on port 1/2/1.

Enter the following commands on edge device B.

```
deviceB(config)# vlan 20
deviceB(config-vlan-20)# untagged ethernet 1/1/1
Added untagged port(s) ethe 1/1/1 to port-vlan 20.

deviceB(config)# vlan 30
deviceB(config-vlan-30)# untagged ethernet 1/1/2
Added untagged port(s) ethe 1/1/2 to port-vlan 30.

deviceB(config)# mvrp enable

deviceB(config)# interface ethernet 1/2/1
deviceB(config-if-e40000-1/2/1)# mvrp enable

deviceB(config-if-e40000-1/2/1)# mvrp point-to-point
deviceB(config-if-e40000-1/2/1)# exit
```

Enter the following commands on edge device C.

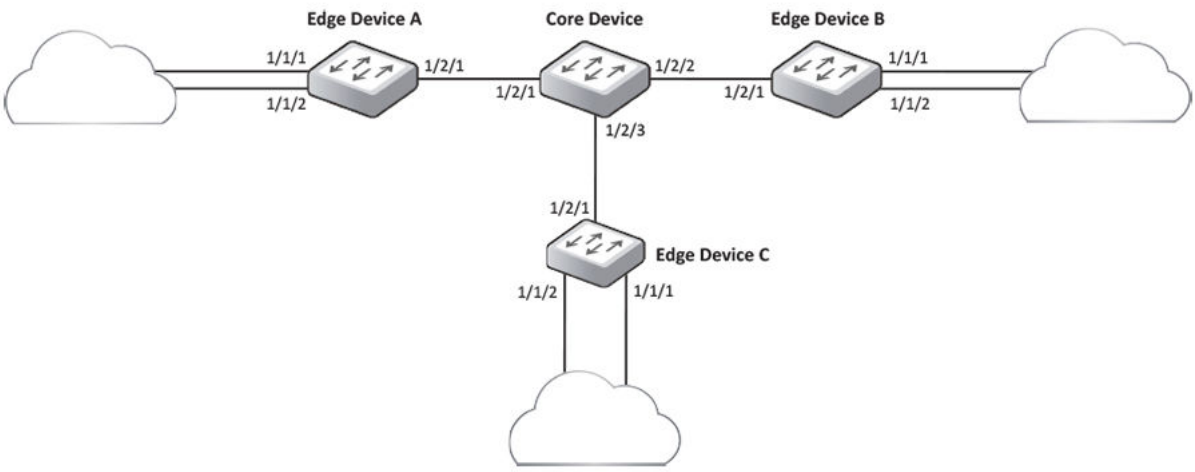
```
deviceC(config)#vlan 30
deviceC(config-vlan-30)# untagged ethernet 1/1/1
Added untagged port(s) ethe 1/1/1 to port-vlan 30.

deviceC(config)# vlan 40
deviceC(config-vlan-40)# untagged ethernet 1/1/2
Added untagged port(s) ethe 1/1/2 to port-vlan 40.


deviceC(config)# mvrp enable

deviceC(config)# interface ethernet 1/2/1
deviceC(config-if-e40000-1/2/1)#mvrp enable

deviceC(config-if-e40000-1/2/1)# mvrp point-to-point
deviceC(config-if-e40000-1/2/1)# exit
```

Configuration Example: Implementing the Applications of MVRP with Per-VLAN RSTP

In the following figure, a core device is attached to three edge devices. Each of the edge devices is attached to other edge devices or host stations (represented by the clouds).

The effects of MVRP in this network depend on which devices the feature is enabled.

FIGURE 52 MVRP Network Example



NOTE
Although some of the devices in the configuration example do not have statically configured VLANs, this is not a requirement. You always can have statically configured VLANs on a device that is running MVRP.

In this configuration, the edge devices advertise their statically configured VLANs to the core device. The core device does not have any statically configured VLANs, but learns the VLANs from the edge devices.

Per-VLAN Spanning Tree Protocol (PVST) is enabled by default on the MVRP untagged VLAN. To enable the MVRP dynamic VLAN to be Rapid Spanning Tree enabled, configure the **mvrp spanning-tree 802.1w** command.

Enter the following commands on the core device.

```
core-device(config)# mvrp enable
core-device(config)# interface ethernet 1/2/1 to 1/2/3
core-device(config-mif-1/2/1-1/2/3)# mvrp enable
core-device(config)# mvrp spanning-tree 802-1w

core-device(config-mif-1/2/1-1/2/3)# mvrp point-to-point
```

The **mvrp enable** command enables the MVRP functionality globally.

Enter the following commands on edge device A.

Enable SSTP, SRSTP, STP, or RSTP on the edge devices (where VLANs are configured statically) according to the network requirement.

```
deviceA(config)# vlan 20
deviceA(config-vlan-20)# untagged ethernet 1/1/1
Added untagged port(s) ethe 1/1/1 to port-vlan 20.
deviceA(config-vlan-20)# spanning-tree 802-1w

deviceA(config)# vlan 40
deviceA(config-vlan-40)# untagged ethernet 1/1/2
Added untagged port(s) ethe 1/1/2 to port-vlan 40.
deviceA(config-vlan-40)# spanning-tree 802-1w

deviceA(config)# mvrp enable

deviceA(config)# interface ethernet 1/2/1
deviceA(config-if-e40000-1/2/1)# mvrp enable

deviceA(config-if-e40000-1/2/1)# mvrp point-to-point
deviceA(config-if-e40000-1/2/1)# exit
```

These commands statically configure two port-based VLANs and enable MVRP on port 1/2/1.

Enter the following commands on edge device B.

```
deviceB(config)# vlan 20
deviceB(config-vlan-20)# untagged ethernet 1/1/1
Added untagged port(s) ethe 1/1/1 to port-vlan 20.
deviceB(config-vlan-20)# spanning-tree 802-1w

deviceB(config)# vlan 30
deviceB(config-vlan-30)# untagged ethernet 1/1/2
Added untagged port(s) ethe 1/1/2 to port-vlan 30.
deviceB(config-vlan-30)# spanning-tree 802-1w

deviceB(config)# mvrp enable

deviceB(config)# interface ethernet 1/2/1
deviceB(config-if-e40000-1/2/1)# mvrp enable

deviceB(config-if-e40000-1/2/1)# mvrp point-to-point
deviceB(config-if-e40000-1/2/1)# exit
```

Enter the following commands on edge device C.

```
deviceC(config)# vlan 30
deviceC(config-vlan-30)# untagged ethernet 1/1/1
Added untagged port(s) ethe 1/1/1 to port-vlan 30.
deviceC(config-vlan-30)# spanning-tree 802-1w

deviceC(config)# vlan 40
deviceC(config-vlan-40)# untagged ethernet 1/1/2
Added untagged port(s) ethe 1/1/2 to port-vlan 40.
deviceC(config-vlan-40)# spanning-tree 802-1w

deviceC(config)# mvrp enable

deviceC(config)# interface ethernet 1/2/1
deviceC(config-if-e40000-1/2/1)#mvrp enable

deviceC(config-if-e40000-1/2/1)# mvrp point-to-point
deviceC(config-if-e40000-1/2/1)# exit
```

Rapid Spanning Tree Protocol

RSTP Overview

Beginning with FastIron release 10.0.20a, Rapid Spanning Tree Protocol (RSTP) (IEEE IEEE 802.1w) replaces the Spanning Tree Protocol (STP) (IEEE 802.1D) standard. RSTP enables faster convergence times than STP. The RSTP is supported on all RUCKUS ICX devices.

Like STP, RSTP eliminates Layer 2 loops in networks by selectively blocking some ports and allowing other ports to forward traffic.

The RSTP is backward-compatible with STP. However, the faster convergence times are lost when interacting with legacy bridges.

> **NOTE**
> This rapid convergence will not occur on ports connected to shared media devices, such as hubs. To take advantage of the rapid convergence provided by IEEE 802.1w, make sure to explicitly configure all point-to-point links in a topology.

The convergence provided by the standard IEEE 802.1w protocol occurs more rapidly than the convergence provided by previous spanning tree protocols because of the following:

- Classic or legacy IEEE 802.1D STP protocol requires a newly selected Root port to go through listening and learning stages before traffic convergence can be achieved. The IEEE 802.1D traffic convergence time is calculated using the following formula.

  2 x FORWARD_DELAY + BRIDGE_MAX_AGE

- Convergence in IEEE 802.1w bridge is not based on any timer values. Rather, it is based on the explicit handshakes between Designated ports and their connected Root ports to achieve convergence in less than 500 milliseconds.

Beginning with FastIron release 10.0.20a, RSTP is enabled by default at the VLAN level.

You can enable or disable RSTP IEEE IEEE 802.1w on the following levels:

- Globally: Affects all ports and port-based VLANs on the device.
- Port-based VLAN: Affects all ports within the specified port-based VLAN. When you enable or disable RSTP within a port-based VLAN, the setting overrides the global setting. Thus, you can enable RSTP for the ports within a port-based VLAN even when RSTP is globally disabled, or disable the ports within a port-based VLAN when RSTP is globally enabled.
- Individual port: Affects only the individual port. However, if you change the RSTP state of the link aggregation group (LAG) virtual interface, the change affects all ports in the LAG.

RSTP Configuration Modes

The following configuration modes apply while configuring RSTP:

- Spanning-tree single IEEE 802.1w: This configuration can be enabled on systems running IEEE 802.1w. The single rapid spanning tree controls all 4000 VLANs. The VLAN can opt in and out of this single rapid spanning tree using the **spanning-tree 802-1w** command under the VLAN prompt. If there is a "spanning-tree 802-1w" configuration under the VLAN, that VLAN will be within that single IEEE 802.1w instance's control, which implies that the VLAN traffic is subject to blocking or forwarding by that spanning tree instance.
- Per-VLAN spanning tree: This configuration mode enables IEEE 802.1w (Rapid Spanning Tree Protocol) at the VLAN level individually.

RSTP Protection Enhancement

RSTP Protection provides the ability to prohibit an end station from initiating or participating in an RSTP topology change.

The RSTP detects and eliminates logical loops in a redundant network by selectively blocking some data paths (ports) and allowing only the best data paths to forward traffic.

In an RSTP environment, switches, end stations, and other Layer 2 devices use Bridge Protocol Data Units (BPDUs) to exchange information that RSTP will use to determine the best path for data flow. When a Layer 2 device is powered on and connected to the network, or when a Layer 2 device goes down, it sends out an RST BPDU, triggering an RSTP topology change.

In some instances, it is unnecessary for a connected device, such as an end station, to initiate or participate in an RSTP topology change. In this case, you can enable RSTP Protection on the RUCKUS port to which the end station is connected. RSTP Protection disables the ability of the connected device to initiate or participate in an RSTP topology change by dropping all BPDUs received from the connected device.

RSTP Parameter Configuration

RUCKUS devices support RSTP as described in the IEEE 802.1w specification. RSTP is disabled by default.

If you configure a port-based VLAN on the device, the VLAN has the same RSTP state as the default RSTP state on the device. You can enable or disable RSTP in each VLAN separately. In addition, you can enable or disable RSTP on individual ports.

Each port-based VLAN on a RUCKUS device runs a separate spanning tree (a separate instance of RSTP). A RUCKUS device has one port-based VLAN (VLAN 1) by default that contains all the device ports. Thus, by default, each RUCKUS device has one spanning tree. However, if you configure additional port-based VLANs on a RUCKUS device, each of those VLANs on which RSTP is enabled and VLAN 1 all run separate spanning trees.

RSTP Parameters and Defaults

The following table lists the default RSTP states for RUCKUS devices.

TABLE 14 Default RSTP States

Device Type	Default RSTP Type	Default RSTP State	Default RSTP State of New VLANs
Router image default	IEEE 802.1W	Enabled	Disabled

The following table lists the default RSTP bridge parameters. The bridge parameters affect the entire spanning tree. If you are using Multiple Spanning Tree Protocol (MSTP), the parameters affect the VLAN.

TABLE 15 Default RSTP Bridge Parameters

Parameter	Description	Default and Valid Values
Forward Delay	The period of time spent by a port in the listening and learning state before moving on to the learning or forwarding state, respectively. The forward delay value is also used for the age time of dynamic entries in the filtering database, when a topology change occurs.	5 seconds Possible values: 4 through 30 seconds
Maximum Age	The interval a bridge will wait for a configuration BPDU from the root bridge before initiating a topology change.	20 seconds Possible values: 6 through 40 seconds
Hello Time	The interval of time between each configuration BPDU sent by the root bridge.	2 seconds Possible values: 1 through 10 seconds
Priority	A parameter used to identify the root bridge in a spanning tree (instance of RSTP). The bridge with the lowest value has the highest priority and is the root. A higher numerical value means a lower priority; thus, the highest priority is 0.	32768 Possible values: 0 through 61440 (in increments of 4096)

NOTE
If a switch is to become a root bridge, before upgrading to FastIron release 10.0.20a, the switch priority must be configured to be less than 4096 and the priority of the other switches must be configured to be greater than 4096.

NOTE
If the priority of a switch is less than 4096, the priority will be updated to 0 after the upgrade to FastIron release 10.0.20a. If the switch priority is greater than 4096 and not a multiple of 4096, the priority will be updated to the lower multiple of 4096 after upgrade to FastIron release 10.0.20a.

NOTE
If you plan to change RSTP bridge timers, RUCKUS recommends that you stay within the following ranges (as defined in section 8.10.2 of the IEEE 802.1D STP specification): **2 * (forward_delay – 1 second) >= max_agemax_age >= 2 * (hello_time + 1 second**).

The following table lists the default RSTP port parameters. The port parameters affect individual ports and are separately configurable on each port.

TABLE 16 Default RSTP Port Parameters

Parameter	Description	Default and Valid Values
Priority	The preference that RSTP gives this port relative to other ports for forwarding traffic out of the spanning tree. A higher numerical value means a lower priority.	128 Possible values: 0 through 240 (configurable in increments of 16)
Path Cost	The cost of using the port to reach the root bridge. When selecting among multiple links to the root bridge, RSTP chooses the link with the lowest path cost and blocks the other paths. Each port type has its own default RSTP path cost.	10 Mbps: 2,000,000 100 Mbps: 200,000 1 Gbps: 20,000 10 Gbps: 2,000 possible values: 1 through 200000000

Configuring IEEE 802.1w Rapid Spanning Tree Protocol

The IEEE 802.1w Rapid Spanning Tree Protocol can be enabled on a port-based VLAN or on an individual port, and IEEE 802.1w port and bridge parameters can be configured.

1. Enter the global configuration mode.

```
device# configure terminal
```

2. Enable IEEE 802.1w on a port-based VLAN or on an individual port.

```
device(config)# vlan 10
device(config-vlan-10)# spanning-tree 802-1w
```

When configured on a port-based VLAN, changing the IEEE 802.1w state in a VLAN affects only that VLAN, including all the ports specified in the VLAN.

The **spanning-tree 802-1w** or **spanning-tree single 802-1w** command must be used to initially enable IEEE 802.1w on ports. Both commands enable IEEE 802.1w on all ports that belong to the VLAN or to the single spanning tree.

NOTE
When you enable or disable IEEE 802.1w within a port-based VLAN, the setting overrides the global setting. Thus, you can enable IEEE 802.1w for the ports within a port-based VLAN even when IEEE 802.1w is globally disabled, or disable the ports within a port-based VLAN when IEEE 802.1w is globally enabled. If you change the IEEE 802.1w state of the LAG virtual interface in a LAG, the change affects all ports in that LAG.

3. Change IEEE 802.1w bridge parameters by using the **max-age**, **hello-time**, and **priority** keywords.

```
device(config-vlan-10)# spanning-tree 802-1w priority 0 hello-time 3 max-age 10
```

The changes are applied to individual ports on the bridge. If you have configured a port-based VLAN on the device, you can configure the parameters only at the configuration level for individual VLANs.

4. Change IEEE 802.1w port parameters by using the **path-cost** and **priority** keywords.

```
device(config-vlan-10)# spanning-tree 802-1w ethernet 1/1/1 path-cost 15 priority 64
```

The following example shows how to enable IEEE 802.1w RSTP on a port-based VLAN, VLAN-10. Bridge and port parameters are configured.

```
device# configure terminal
device(config)# vlan 10
device(config-vlan-10)# spanning-tree 802-1w
device(config-vlan-10)# spanning-tree 802-1w priority 0 hello-time 3 max-age 10
device(config-vlan-10)# spanning-tree 802-1w ethernet 1/1/1 path-cost 15 priority 64
```

The following example shows how to enable IEEE 802.1w RSTP port parameters on an individual port, Ethernet 1/1/1, and LAG 45, respectively.

```
device# configure terminal
device(config)# interface ethernet 1/1/1
device(config-if-e1000-1/1/1)#spanning-tree 802-1w admin-edge-port

device# configure terminal
device(config)# interface lag 45
device(config-lag-if-lg45)# spanning-tree 802-1w admin-pt2pt-mac enable
```

Note Regarding Pasting IEEE 802.1w Settings into the Running Configuration

If you paste IEEE 802.1w settings into the running configuration, and the pasted configuration includes ports that are already up, the ports will initially operate in STP legacy mode before operating in IEEE 802.1w RSTP mode. For example, the following pasted configuration will cause Ethernet ports 1/1/1 and 1/1/2 to temporarily operate in STP legacy mode, because these ports are already up and running.

```
configure terminal
vlan 120
tag ethernet 1/1/1 to 1/1/2
spanning-tree 802-1w
spanning-tree 802-1w priority 4096
end
```

To avoid this issue, IEEE 802.1w commands or settings that are pasted into the configuration must be in the following order.

1. Ports that are not yet connected
2. IEEE 802.1w RSTP settings
3. Ports that are already up

In the following configuration example, untagged ethernet port 2/1/1 is added to VLAN 120 *before* the IEEE 802.1w RSTP settings, and ethernet ports 1/1/1 and 1/1/2 are added *after* the IEEE 802.1w RSTP settings. When these commands are pasted into the running configuration, the ports will properly operate in IEEE 802.1w RSTP mode.

```
configure terminal
vlan 120
untag ethernet 2/1/1
spanning-tree 802-1w
spanning-tree 802-1w priority 4096
tag ethernet 1/1/1 to 1/1/2
end
```

Path Cost Values

The **path-cost** *value* parameter specifies the cost of the port path to the root bridge. IEEE 802.1w prefers the path with the lowest cost. You can specify a value from 1 - 20,000,000. The following table shows the recommended path cost values from the IEEE standards.

TABLE 17 Recommended Path Cost Values of IEEE 802.1w

Link Speed	Recommended (Default) IEEE 802.1w Path Cost Values	Recommended IEEE 802.1w Patch Cost Range
Less than 100 kilobits per second	200,000,000	20,000,000 – 200,000,000
1 Megabit per second	20,000,000	2,000,000 – 200,000,000
10 Megabits per second	2,000,000	200,000 – 200,000,000
100 Megabits per second	200,000	20,000 – 200,000,000
1 Gbps per second	20,000	2,000 – 200,000,000
10 Gbps per second	2,000	200 – 20,000
100 Gbps per second	200	20 – 2,000
1 Terabits per second	20	2 – 200
10 Terabits per second	2	1 – 20

For example, suppose you want to enable IEEE 802.1w on a system with no active port-based VLANs and change the hello-time from the default value of 2 to 8 seconds. Additionally, suppose you want to change the path and priority costs for ethernet port 1/1/5 only. To do so, enter the following commands.

```
device(config)# spanning-tree 802-1w hello-time 8
device(config)# spanning-tree 802-1w ethernet 1/1/5 path-cost 15 priority 64
```

Displaying RSTP (IEEE 802.1w) Information

To show a summary of RSTP (IEEE 802.1w) information, enter the **show 802-1w** command.

```
device# show 802-1w
--- VLAN 1 [ STP Instance owned by VLAN 1 ] ---------------------------
VLAN 1 BPDU cam_index is 2 and the IGC and DMA master Are(HEX) 0 1 2 3
Bridge IEEE 802.1W Parameters:
Bridge            Bridge  Bridge Bridge Force    tx
Identifier        MaxAge  Hello  FwdDly Version Hold
hex               sec     sec    sec            cnt
800000e080541700  20      2      15     Default 3
RootBridge        RootPath   DesignatedBri-    Root  Max  Fwd  Hel
Identifier        Cost       dge Identifier    Port  Age  Dly  lo
hex                          hex                    sec  sec  sec
800000e0804c9c00  200000     800000e0804c9c00  1     20   15   2
Port IEEE 802.1W Parameters:
      <--- Config Params -->|<-------------- Current state ----------------->
Port  Pri PortPath P2P Edge Role       State      Designa- Designated
Num       Cost     Mac Port                      ted cost bridge
1/1/1 128 200000   F   F    ROOT       FORWARDING 0        800000e0804c9c00
1/1/2 128 200000   F   F    DESIGNATED FORWARDING 200000   800000e080541700
1/1/3 128 200000   F   F    DESIGNATED FORWARDING 200000   800000e080541700
1/1/4 128 200000   F   F    BACKUP     DISCARDING 200000   800000e080541700
```

To display detailed information about RSTP (IEEE 802.1w), enter the **show 802-1w detail** command.

```
device# show 802-1w detail
========================================================================
VLAN 1 - SPANNING TREE (IEEE 802.1W) ACTIVE
========================================================================
BridgeId 800000a0c9c002a0, forceVersion 2, txHoldCount 3

Number of topology changes: 53
Last topology change occured 13 minute(s) 30 second(s)  ago on lg40
Port 1/1/1 - Role: DESIGNATED - State: FORWARDING
  PathCost 20000, Priority 128, AdminOperEdge F, AdminPt2PtMac F
  DesignatedPriority - Root: 0x800000a0c9c002a0, Bridge: 0x800000a0c9c002a0
  ActiveTimers - helloWhen 0
  MachineStates - PIM: CURRENT, PRT: DESIGNATED_PORT, PST: FORWARDING
                  TCM: ACTIVE, PPM: SENDING_RSTP, PTX: TRANSMIT_IDLE
  Received - RST BPDUs 0, Config BPDUs 0, TCN BPDUs 0
```

Bridges and Bridge Port Roles

A bridge in an IEEE 802.1w rapid spanning tree topology is assigned as the root bridge if it has the highest priority (lowest bridge identifier) in the topology. Other bridges are referred to as non-root bridges.

Unique roles are assigned to ports on the root and non-root bridges. Role assignments are based on the following information contained in the Rapid Spanning Tree Bridge Packet Data Unit (RST BPDU):

- Root bridge ID
- Path cost value
- Transmitting bridge ID
- Designated port ID

The IEEE 802.1w algorithm uses this information to determine if the RST BPDU received by a port is superior to the RST BPDU that the port transmits. The two values are compared in the order as given above, starting with the Root bridge ID. The RST BPDU with a lower value is considered superior. The superiority and inferiority of the RST BPDU is used to assign a role to a port.

If the value of the received RST BPDU is the same as that of the transmitted RST BPDU, then the port IDs in the RST BPDUs are compared. The RST BPDU with the lower port ID is superior. Port roles are then calculated appropriately.

The port role is included in the BPDU that it transmits. The BPDU transmitted by an IEEE 802.1w port is referred to as an RST BPDU, while it is operating in IEEE 802.1w mode.

Ports can have one of the following roles:

- **Root**: Provides the lowest cost path to the root bridge from a specific bridge
- **Designated**: Provides the lowest cost path to the root bridge from a LAN to which it is connected
- **Alternate**: Provides an alternate path to the root bridge when the root port goes down
- **Backup**: Provides a backup to the LAN when the Designated port goes down
- **Disabled**: Has no role in the topology

Assignment of Port Roles

At system start-up, all IEEE 802.1w-enabled bridge ports assume a Designated role. Once start-up is complete, the IEEE 802.1w algorithm calculates the superiority or inferiority of the RST BPDU that is received and transmitted on a port.

On a root bridge, each port is assigned a Designated port role, except for ports on the same bridge that are physically connected together. In these types of ports, the port that receives the superior RST BPDU becomes the Backup port, while the other port becomes the Designated port.

On non-root bridges, ports are assigned as follows:

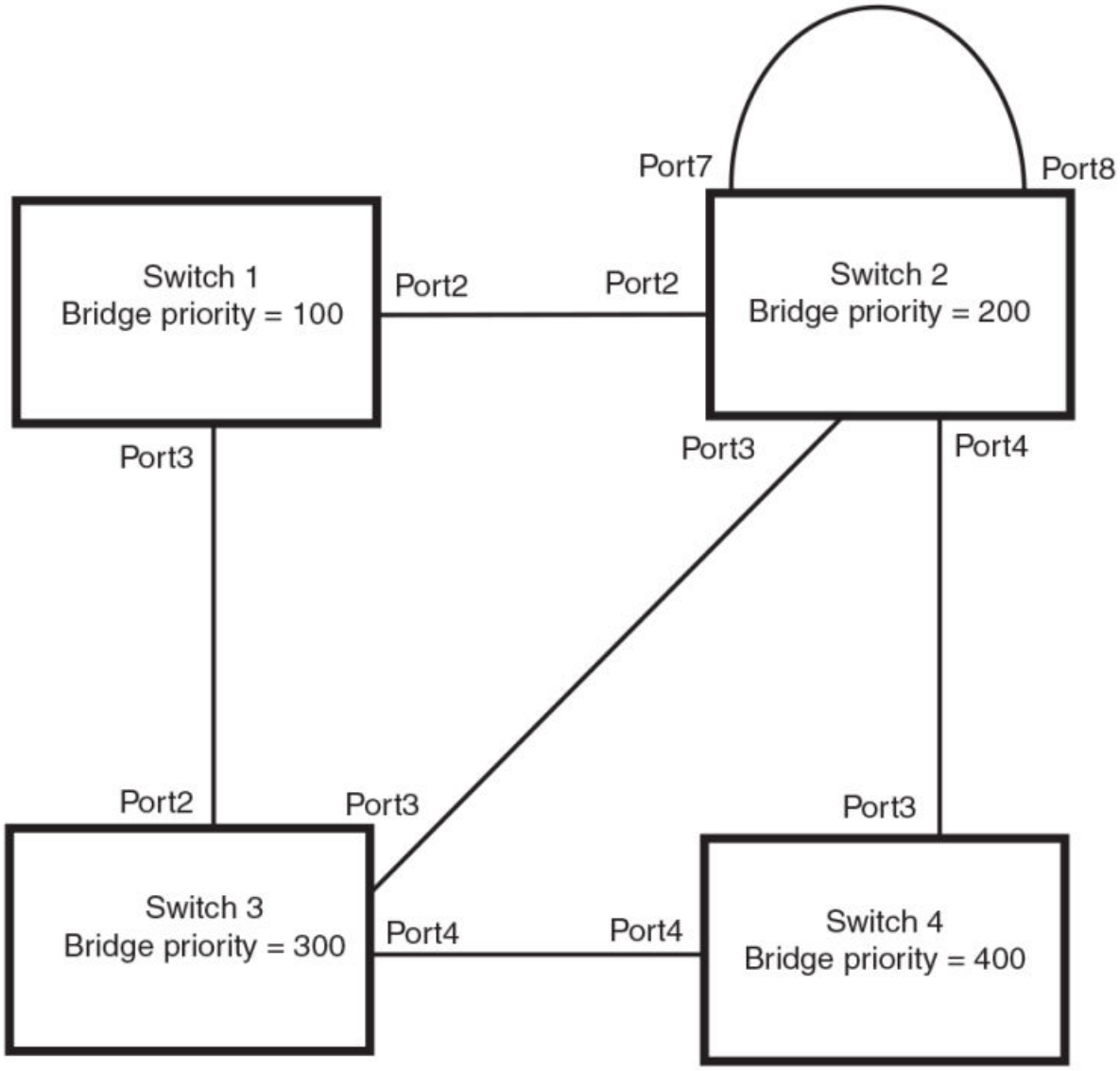
- The port that receives the RST BPDU with the lowest path cost from the root bridge becomes the Root port .
- If two ports on the same bridge are physically connected, the port that receives the superior RST BPDU becomes the Backup port, while the other port becomes the Designated port.
- If a non-root bridge already has a Root port, then the port that receives an RST BPDU that is superior to those it can transmit becomes the Alternate port.
- If the RST BPDU that a port receives is inferior to the RST BPDUs it transmits, then the port becomes a Designated port.
- If the port is down or if IEEE 802.1w is disabled on the port, that port is given the role of Disabled port. Disabled ports have no role in the topology. However, if IEEE 802.1w is enabled on a port with a link down and the link of that port comes up, then that port assumes one of the following port roles: Root, Designated, Alternate, or Backup.

The following example (Figure 53) explains role assignments in a simple RSTP topology.

> **NOTE**
> All examples in this document assume that all ports in the illustrated topologies are point-to-point links and are homogeneous (they have the same path cost value) unless otherwise specified.

The topology in the following figure contains four bridges. Switch 1 is the root bridge since it has the lowest bridge priority. Switch 2 through Switch 4 are non-root bridges.

FIGURE 53 Simple IEEE 802.1w Topology



NOTE
Port numbers are simplified.

Assignment of Ports on Switch 1

All ports on Switch 1, the root bridge, are assigned Designated port roles.

Assignment of Ports on Switch 2

Port2 on Switch 2 directly connects to the root bridge; therefore, Port2 is the Root port.

The bridge priority value on Switch 2 is superior to that of Switch 3 and Switch 4; therefore, the ports on Switch 2 that connect to Switch 3 and Switch 4 are given the Designated port role.

Furthermore, Port7 and Port8 on Switch 2 are physically connected. The RST BPDUs transmitted by Port7 are superior to those Port8 transmits. Therefore, Port8 is the Backup port and Port7 is the Designated port.

Assignment of Ports on Switch 3

Port2 on Switch 3 directly connects to the Designated port on the root bridge; therefore, it assumes the Root port role.

The root path cost of the RST BPDUs received on Port4/Switch 3 is inferior to the RST BPDUs transmitted by the port; therefore, Port4/Switch 3 becomes the Designated port.

Similarly Switch 3 has a bridge priority value inferior to Switch 2. Port3 on Switch 3 connects to Port 3 on Switch 2. This port will be given the Alternate port role, since a Root port is already established on this bridge.

Assignment of Ports on Switch 4

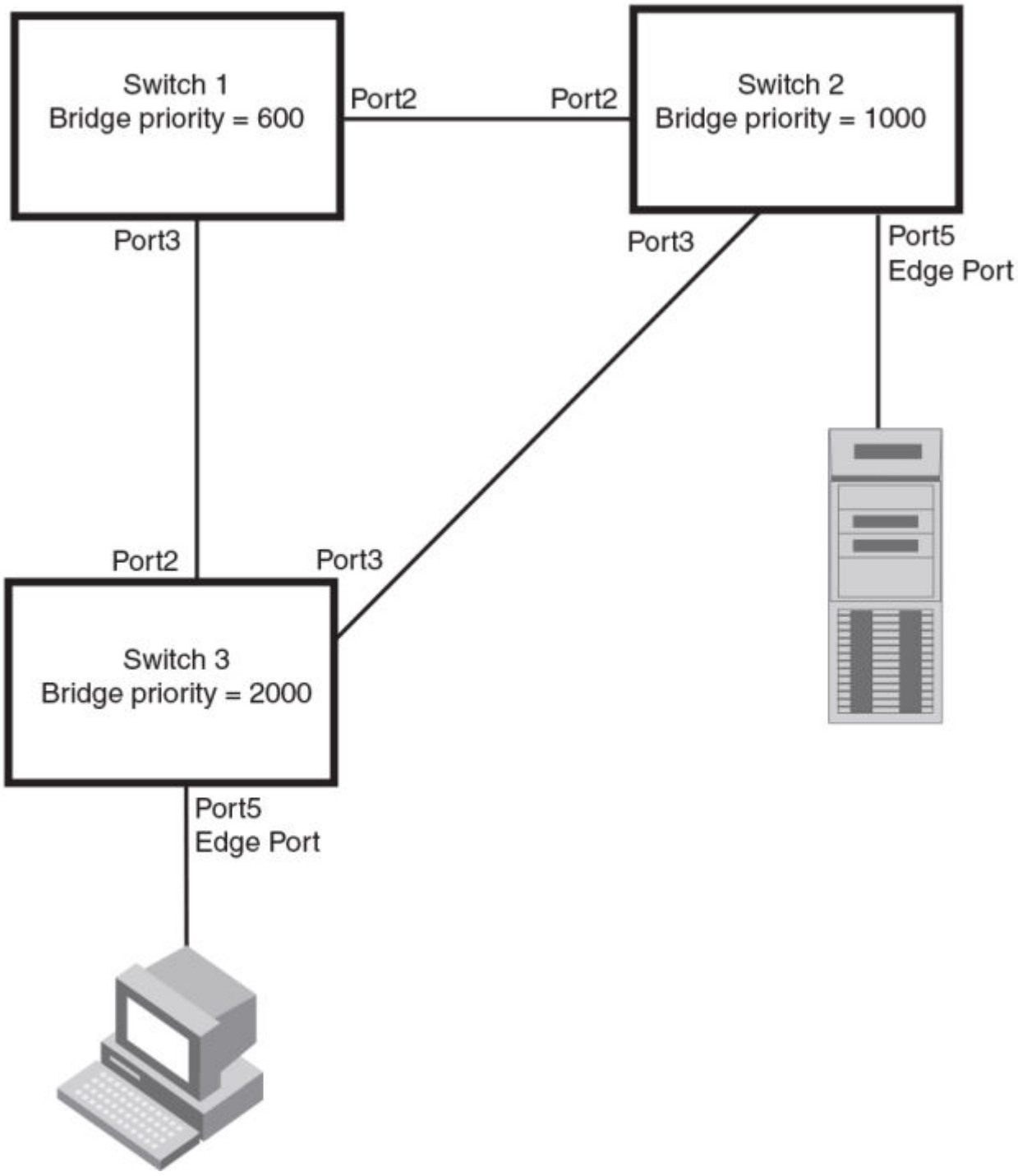
Switch 4 is not directly connected to the root bridge. It has two ports with superior incoming RST BPDUs from two separate LANs: Port3 and Port4. The RST BPDUs received on Port3 are superior to the RST BPDUs received on port 4; therefore, Port3 becomes the Root port and Port4 becomes the Alternate port.

Edge Ports and Edge Port Roles

The RUCKUS implementation of IEEE 802.1w allows ports that are configured as Edge ports to be present in an IEEE 802.1w topology. (Figure 54). Edge ports are ports of a bridge that connect to workstations or computers. Edge ports do not register any incoming BPDU activities.

Edge ports assume Designated port roles. Port flapping does not cause any topology change events on Edge ports since IEEE 802.1w does not consider Edge ports in the spanning tree calculations.

FIGURE 54 Topology with Edge Ports



NOTE
Port numbers are simplified.

However, if any incoming RST BPDU is received from a previously configured Edge port, IEEE 802.1w automatically makes the port a non-edge port. This is extremely important to ensure a loop-free Layer 2 operation since a non-edge port is part of the active RSTP topology.

The IEEE 802.1w protocol can auto-detect an Edge port and a non-edge port. An administrator can also configure a port to be an Edge port using the CLI. It is recommended that Edge ports are configured explicitly to take advantage of the Edge port feature, instead of allowing the protocol to auto-detect them.

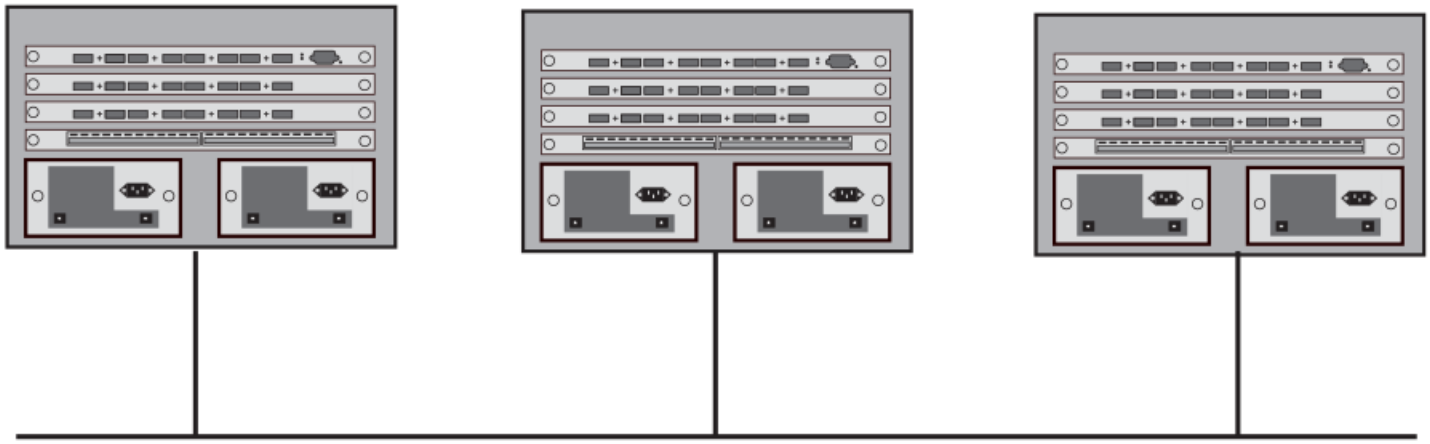
Point-to-Point Ports

To take advantage of the IEEE 802.1w features, ports on an IEEE 802.1w topology should be explicitly configured as point-to-point links using the CLI. Shared media should not be configured as point-to-point links.

NOTE
Configuring shared media or non-point-to-point links as point-to-point links could lead to Layer 2 loops.

The topology in the following figure is an example of shared media that should not be configured as point-to-point links. In this figure, a port on a bridge communicates or is connected to at least two ports.

FIGURE 55 Example of Shared Media



Bridge Port States

Ports roles can have one of the following states:

- Forwarding: IEEE 802.1w is allowing the port to send and receive all packets.
- Discarding: IEEE 802.1w has blocked data traffic on this port to prevent a loop. The device or VLAN can reach the root bridge using another port, for which the state is forwarding. When a port is in this state, the port does not transmit or receive data frames, but the port does continue to receive RST BPDUs. This state corresponds to the listening and blocking states of 802.1D.
- Learning: IEEE 802.1w is allowing MAC entries to be added to the filtering database but does not permit forwarding of data frames. The device can learn the MAC addresses of frames that the port receives during this state and make corresponding entries in the MAC table.
- Disabled: The port is not participating in IEEE 802.1w. This can occur when the port is disconnected or IEEE 802.1w is administratively disabled on the port.

A port on a non-root bridge with the role of Root port is always in a forwarding state. If another port on that bridge assumes the Root port role, then the old Root port moves into a discarding state as it assumes another port role.

A port on a non-root bridge with a Designated role starts in the discarding state. When that port becomes elected to the Root port role, IEEE 802.1w quickly places it into a forwarding state. However, if the Designated port is an Edge port, then the port starts and stays in a forwarding state and it cannot be elected as a Root port.

A port with an Alternate or Backup role is always in a discarding state. If the port role changes to Designated, then the port changes into a forwarding state.

If a port on one bridge has a Designated role and that port is connected to a port on another bridge that has an Alternate or Backup role, the port with a Designated role cannot be given a Root port role until two instances of the forward delay timer expires on that port.

Edge Port and Non-Edge Port States

As soon as a port is configured as an Edge port using the CLI, it goes into a forwarding state instantly (in less than 100 msec).

When the link to a port comes up and IEEE 802.1w detects that the port is an Edge port, that port instantly goes into a forwarding state.

If IEEE 802.1w detects that port as a non-edge port, the port state is changed as determined by the result of processing the received RST BPDU. The port state change occurs within four seconds of link up or after the hello timer expires twice on the port.

Changes to Port Roles and States

To achieve convergence in a topology, a port role and state changes as it receives and transmits new RST BPDUs. Changes in a port role and state constitute a topology change. Besides the superiority and inferiority of the RST BPDU, bridge-wide and per-port state machines are used to determine a port role as well as a port state. Port state machines also determine when port role and state changes occur.

Port Role Selection State Machines

The bridge uses the Port Role Selection state machine to determine if port role changes are required on the bridge. This state machine performs a computation when one of the following events occur:

- New information is received on any port on the bridge.
- The timer expires for the current information on a port on the bridge.

Each port uses the following state machines:

- Port Information: This state machine keeps track of spanning-tree information currently used by the port. It records the origin of the information and ages out any information that was derived from an incoming BPDU.
- Port Role Transition: This state machine keeps track of the current port role and transitions the port to the appropriate role when required. It moves the Root port and the Designated port into forwarding states and moves the Alternate and Backup ports into discarding states.
- Port Transmit: This state machine is responsible for BPDU transmission. It checks to ensure only the maximum number of BPDUs per hello interval are sent every second. Based on what mode it is operating in, it sends out either legacy BPDUs or RST BPDUs. In this document legacy BPDUs are also referred to as STP BPDUs.
- Port Protocol Migration: This state machine deals with compatibility with 802.1D bridges. When a legacy BPDU is detected on a port, this state machine configures the port to transmit and receive legacy BPDUs and operate in the legacy mode.
- Topology Change: This state machine detects, generates, and propagates topology change notifications. It acknowledges Topology Change Notice (TCN) messages when operating in 802.1D mode. It also flushes the MAC table when a topology change event takes place.

- Port State Transition: This state machine transitions the port to a discarding, learning, or forwarding state and performs any necessary processing associated with the state changes.
- Port Timers: This state machine is responsible for triggering any of the state machines described above, based on expiration of specific port timers.

In contrast to the IEEE 802.1D standard, the IEEE 802.1w standard does not have any bridge-specific timers. All timers in the CLI are applied on a per-port basis, even though they are configured under bridge parameters.

IEEE 802.1w state machines attempt to quickly place the ports into either a forwarding or discarding state. Root ports are quickly placed in the forwarding state when both of the following events occur:

- It is assigned to be the Root port.
- It receives an RST BPDU with a proposal flag from a Designated port. The proposal flag is sent by ports with a Designated role when they are ready to move into a forwarding state.

When the role of Root port is given to another port, the old Root port is instructed to reroot. The old Root port goes into a discarding state and negotiates with its peer port for a new role and a new state. A peer port is the port on the other bridge to which the port is connected. For example, in Figure 56, Port1 of Switch 200 is the peer port of Port2 of Switch 100.

A port with a Designated role is quickly placed into a forwarding state if one of the following occurs:

- The Designated port receives an RST BPDU that contains an agreement flag from a Root port.
- The Designated port is an Edge port.

However, a Designated port that is attached to an Alternate port or a Backup port must wait until the forward delay timer expires twice on that port while it is still in a Designated role, before it can proceed to the forwarding state.

Backup ports are quickly placed into discarding states.

Alternate ports are quickly placed into discarding states.

A port operating in IEEE 802.1w mode may enter a learning state to allow MAC entries to be added to the filtering database; however, this state is transient and lasts only a few milliseconds, if the port is operating in IEEE 802.1w mode and if the port meets the conditions for rapid transition.

Handshake Mechanisms

To rapidly transition a Designated or Root port into a forwarding state, the Port Role Transition state machine uses handshake mechanisms to ensure loop-free operations. It uses one type of handshake if no Root port has been assigned on a bridge, and another type if a Root port has already been assigned.

Handshake When No Root Port is Elected

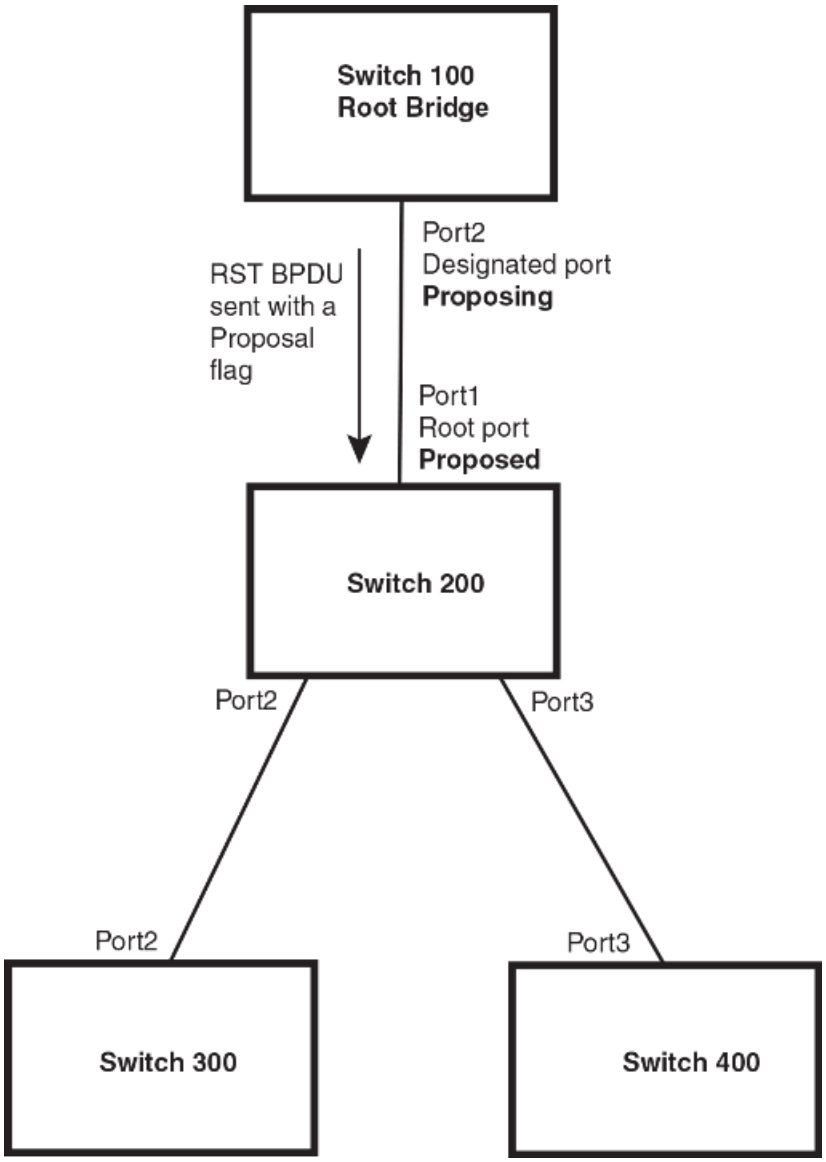
If a Root port has not been assigned on a bridge, IEEE 802.1w uses the Proposing -> Proposed -> Sync -> Synced -> Agreed handshake:

- Proposing: The Designated port on the root bridge sends an RST BPDU packet to its peer port that contains a proposal flag. The proposal flag is a signal that indicates that the Designated port is ready to put itself in a forwarding state (Figure 56). The Designated port continues to send this flag in its RST BPDU until it is placed in a forwarding state (Figure 59) or is forced to operate in 802.1D mode. Refer to #unique_220.
- Proposed: When a port receives an RST BPDU with a proposal flag from the Designated port on its point-to-point link, it asserts the Proposed signal and one of the following occurs (Figure 56):
  - If the RST BPDU that the port receives is superior to what it can transmit, the port assumes the role of a Root port. (Refer to Bridges and Bridge Port Roles on page 180.)
  - If the RST BPDU that the port receives is inferior to what it can transmit, then the port is given the role of Designated port.

NOTE
Proposed will never be asserted if the port is connected on a shared media link.

In the following figure, Port1/Switch 200 is elected as the Root port.

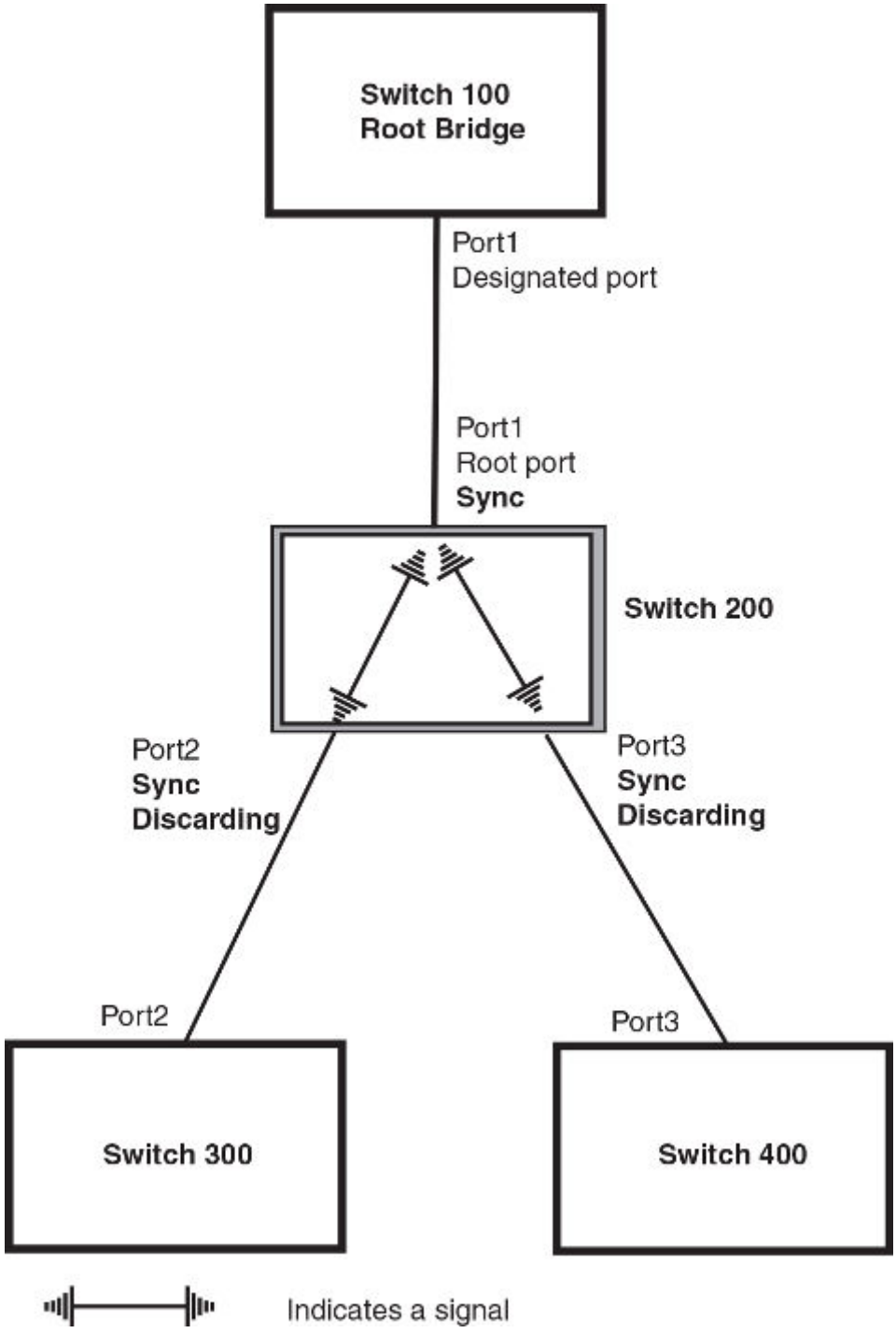
FIGURE 56 Proposing and Proposed Stage



NOTE
Port numbers are simplified.

- Sync: Once the Root port is elected, it sets a sync signal on all the ports on the bridge. The signal tells the ports to synchronize their roles and states (Figure 57). Ports that are non-edge ports with a role of Designated port change into a discarding state. These ports have to negotiate with their peer ports to establish their new roles and states.

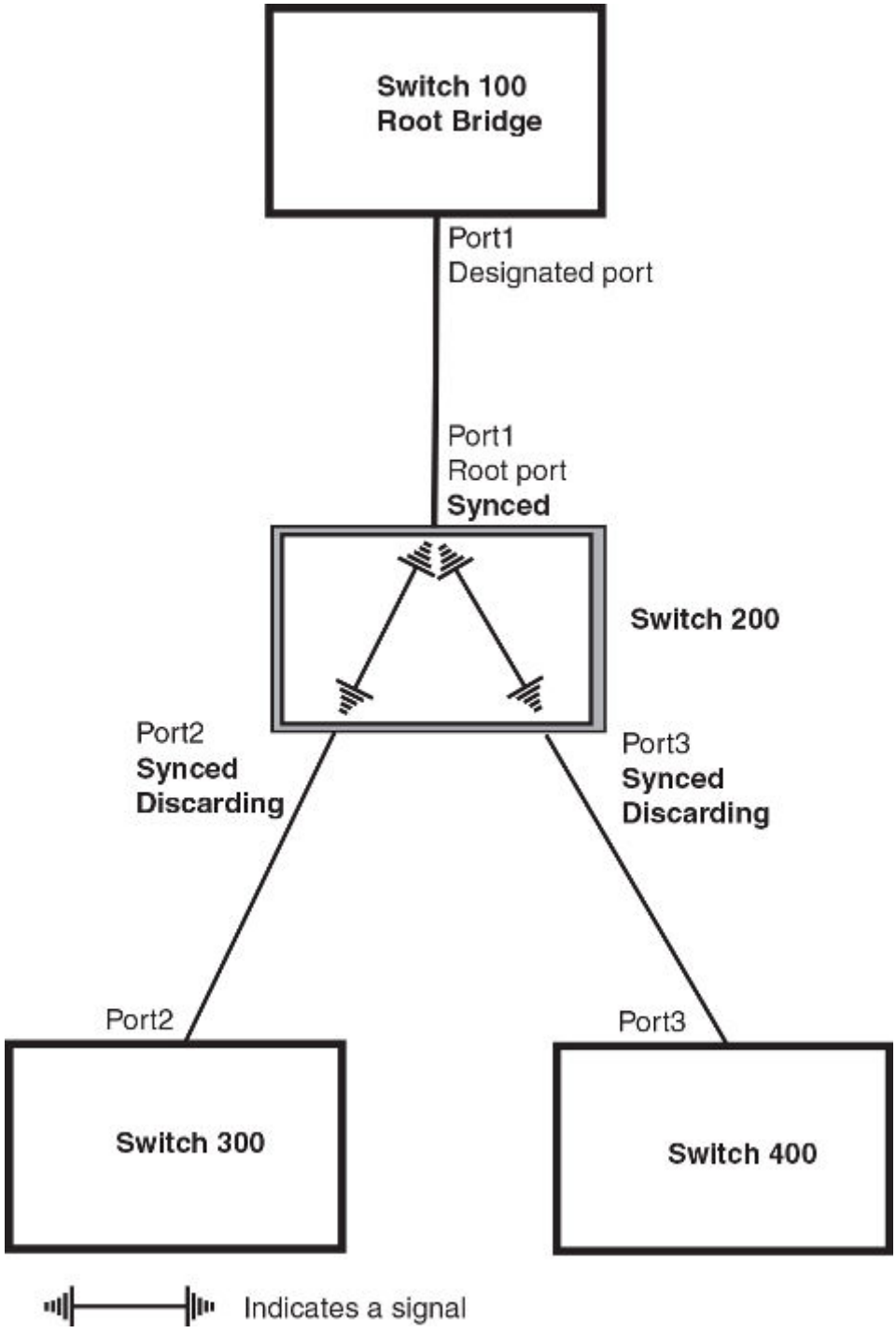
FIGURE 57 Sync Stage



NOTE
Port numbers are simplified.

- Synced: Once the Designated port changes into a discarding state, it asserts a synced signal. Immediately, Alternate ports and Backup ports are synced. The Root port monitors the synced signals from all the bridge ports. Once all bridge ports asserts a synced signal, the Root port asserts its own synced signal as shown in the following figure.

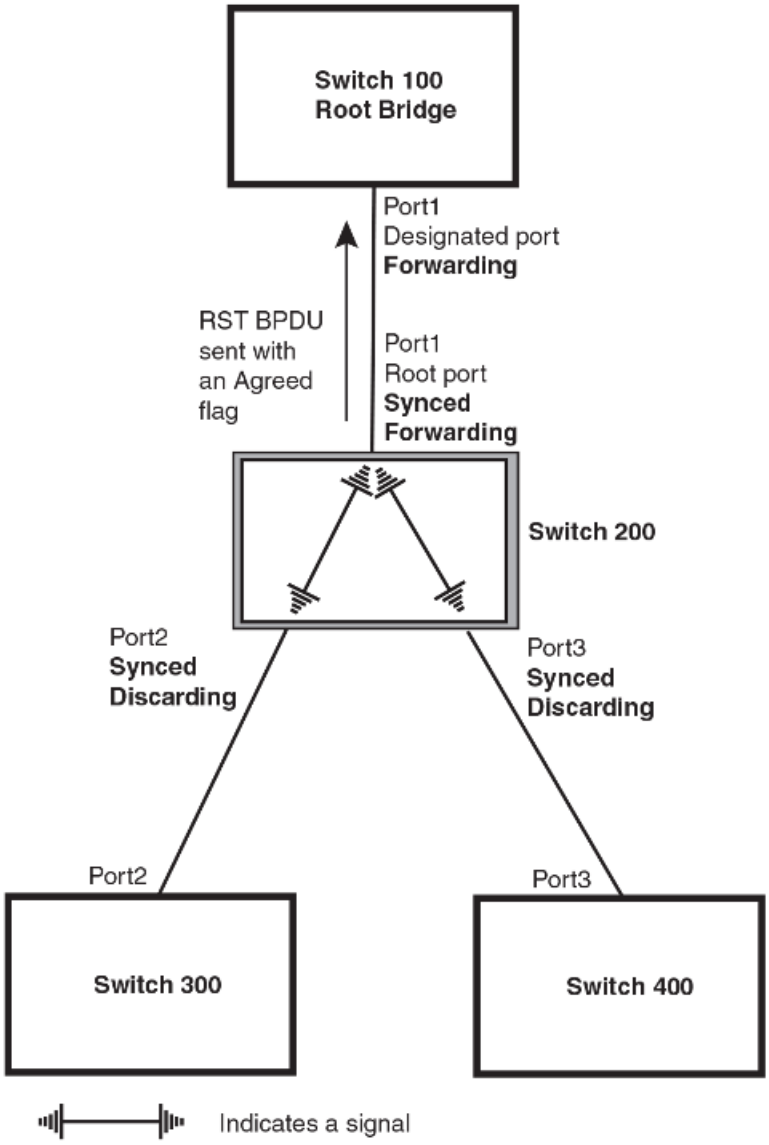
FIGURE 58 Synced Stage



NOTE
Port numbers are simplified.

- Agreed: The Root port sends back an RST BPDU containing an agreed flag to its peer Designated port and moves into the forwarding state. When the peer Designated port receives the RST BPDU, it rapidly transitions into a forwarding state.

FIGURE 59 Agree Stage



NOTE
Port numbers are simplified.

At this point, the handshake mechanism is complete between Switch 100 (the root bridge) and Switch 200.

Switch 200 updates the information on the Switch 200 Designated ports (Port2 and Port3) and identifies the new root bridge. The Designated ports send RST BPDUs, containing proposal flags, to their downstream bridges, without waiting for the hello timers to expire on them. This process starts the handshake with the downstream bridges.

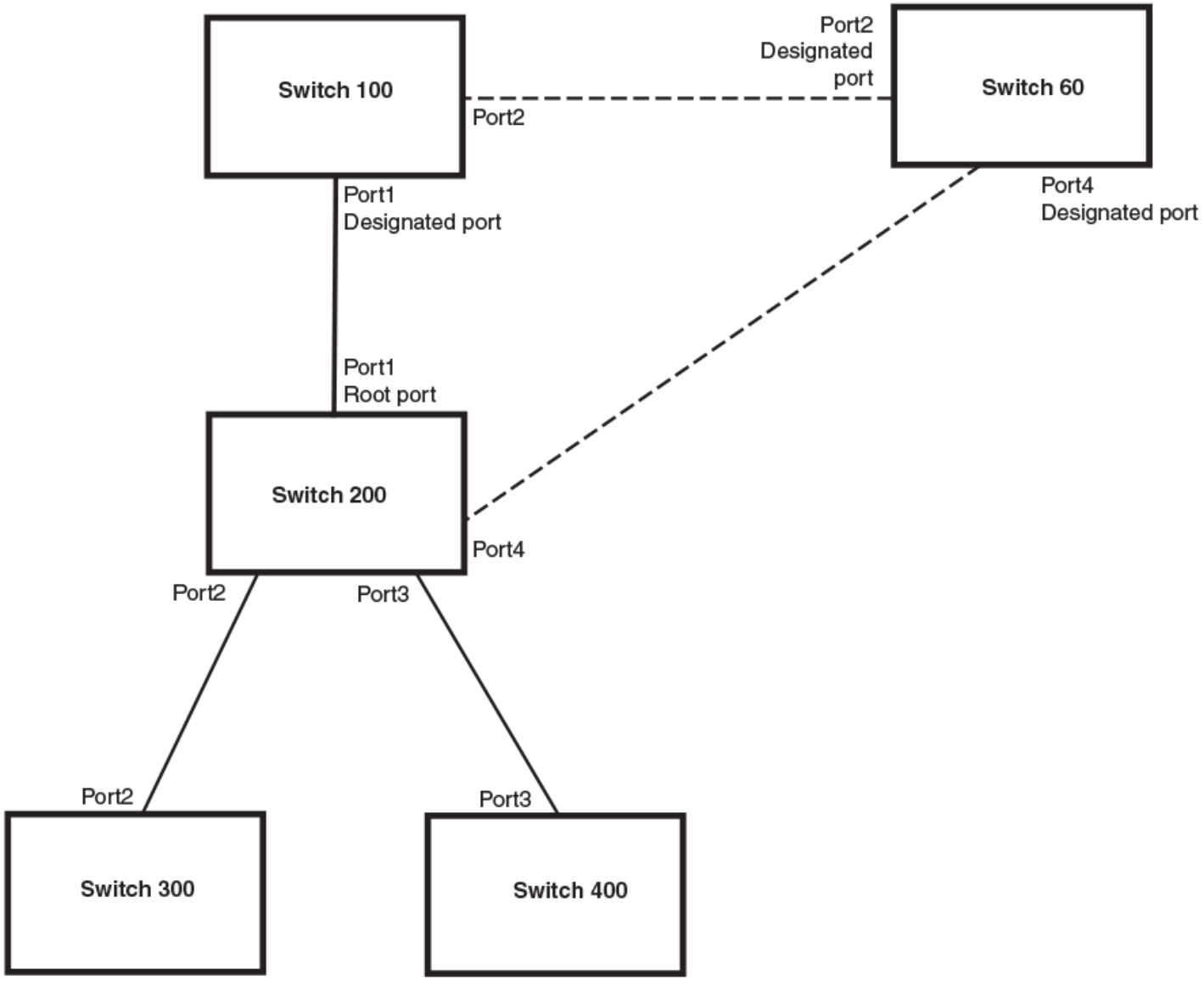
For example, Port2/Switch 200 sends an RST BPDU to Port2/Switch 300 that contains a proposal flag. Port2/Switch 300 asserts a proposed signal. Ports in Switch 300 then set sync signals on the ports to synchronize and negotiate their roles and states. Then the ports assert a synced signal and when the Root port in Switch 300 asserts its synced signal, it sends an RST BPDU to Switch 200 with an agreed flag.

This handshake is repeated between Switch 200 and Switch 400 until all Designated and Root ports are in forwarding states.

Handshake When a Root Port Has Been Elected

If a non-root bridge already has a Root port, IEEE 802.1w uses a different type of handshake. For example, in the following figure, a new root bridge is added to the topology.

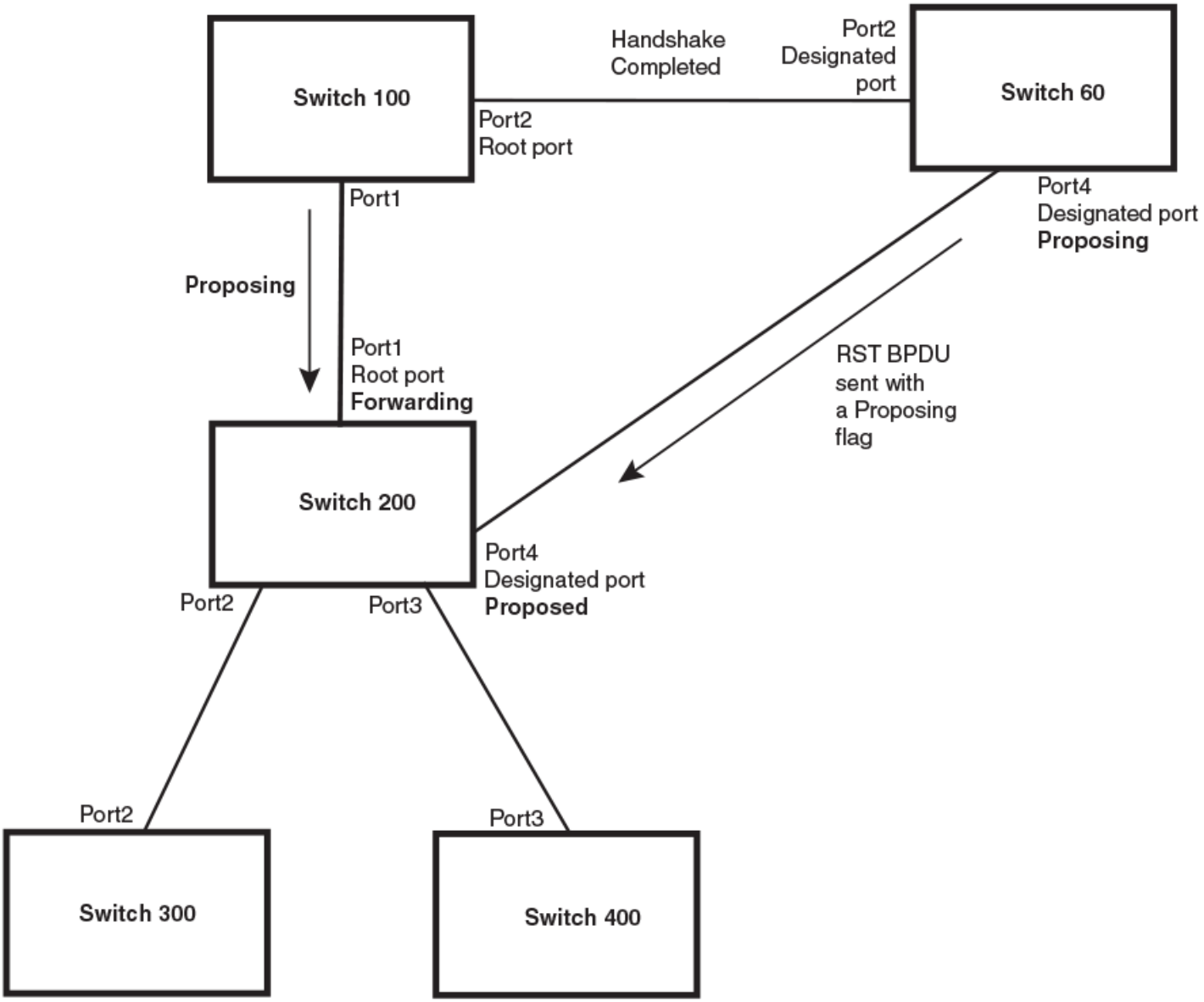
FIGURE 60 Addition of a New Root Bridge



NOTE
Port numbers are simplified.

The handshake that occurs between Switch 60 and Switch 100 follows the process described in section Handshake When No Root Port is Elected. The former root bridge becomes a non-root bridge and establishes a Root port (Figure 61).

FIGURE 61 New Root Bridge Sending a Proposal Flag



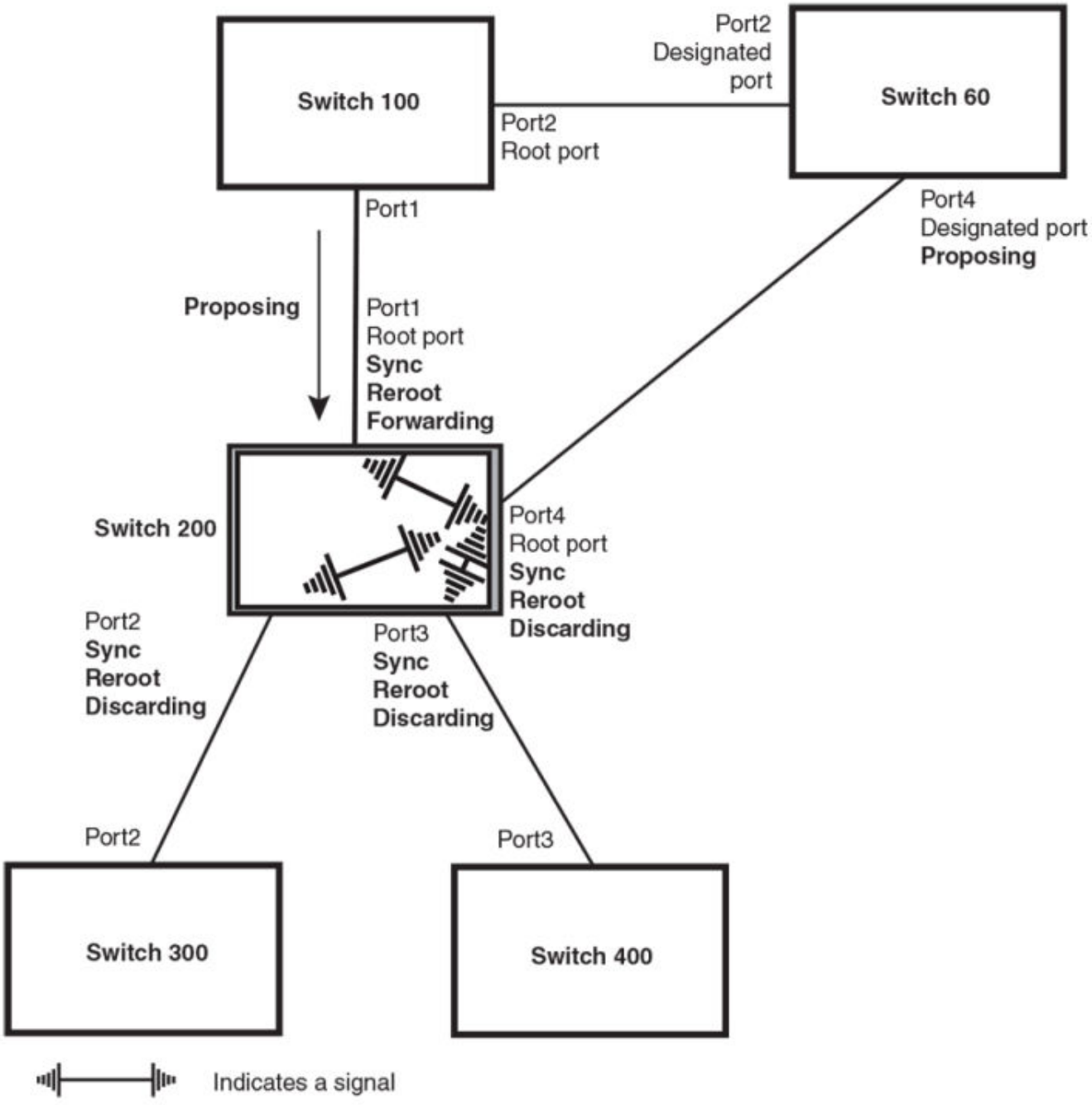
NOTE
Port numbers are simplified.

However, since Switch 200 already has a Root port in a forwarding state, IEEE 802.1w uses the Proposing -> Proposed -> Sync and Reroot -> Sync and Rerooted -> Rerooted and Synced -> Agreed handshake:

- Proposing and Proposed: The Designated port on the new root bridge (Port4/Switch 60) sends an RST BPDU that contains a proposing signal to Port4/Switch 200 to inform the port that it is ready to put itself in a forwarding state (Figure 61). IEEE 802.1w algorithm determines that the RST BPDU that Port4/Switch 200 received is superior to what it can generate, so Port4/Switch 200 assumes a Root port role.
- Sync and Reroot: The Root port then asserts a sync and a reroot signal on all the ports on the bridge. The signal tells the ports that a new Root port has been assigned and they are to renegotiate their new roles and states. The other ports on the bridge assert their sync and

reroot signals. Information about the old Root port is discarded from all ports. Designated ports change into discarding states as shown in the following figure.

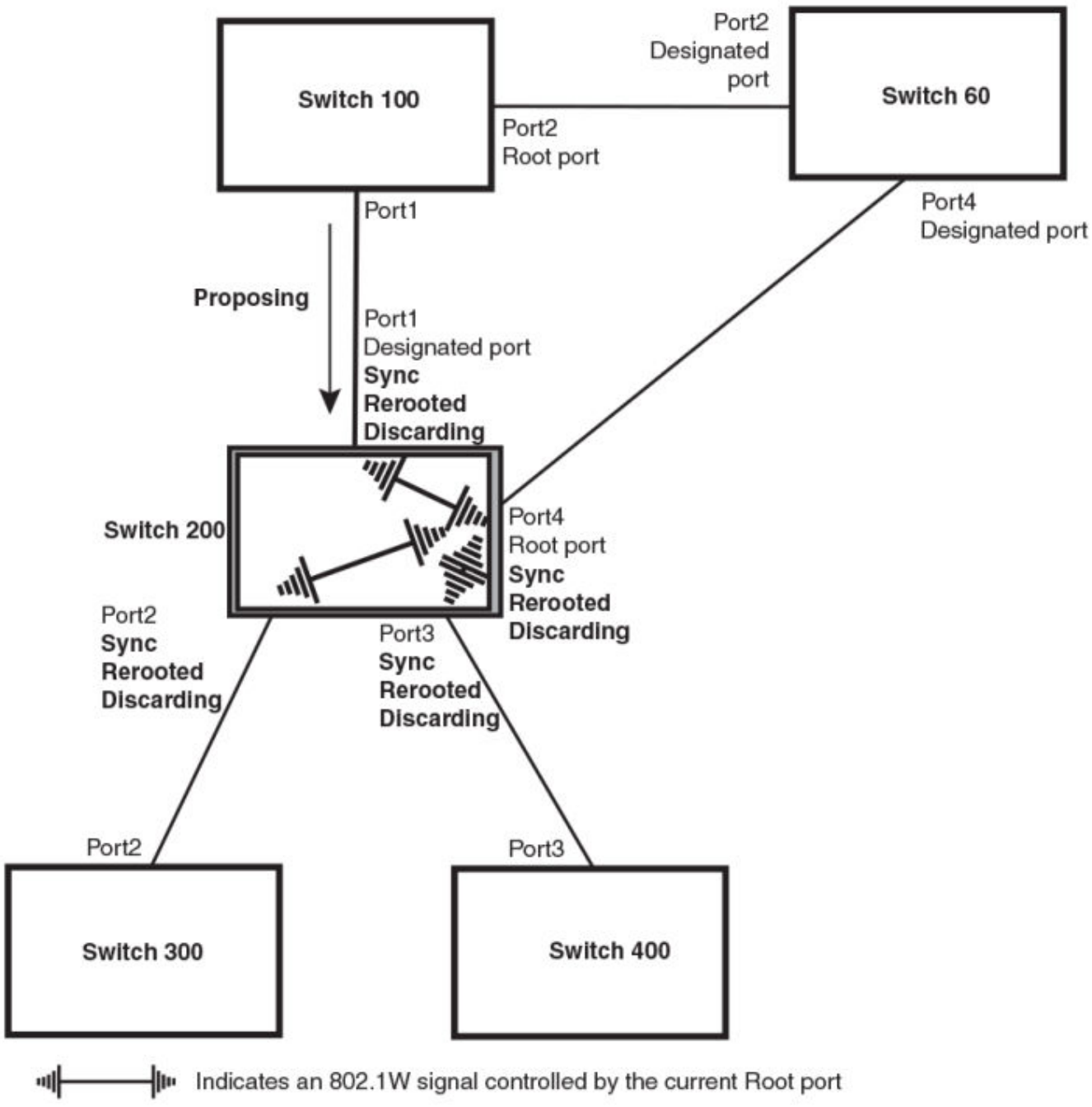
FIGURE 62 Sync and Reroot



NOTE
Port numbers are simplified.

- Sync and Rerooted: When the ports on Switch 200 have completed the reroot phase, they assert their rerooted signals and continue to assert their sync signals as they continue in their discarding states. They also continue to negotiate their roles and states with their peer ports as shown in the following figure.

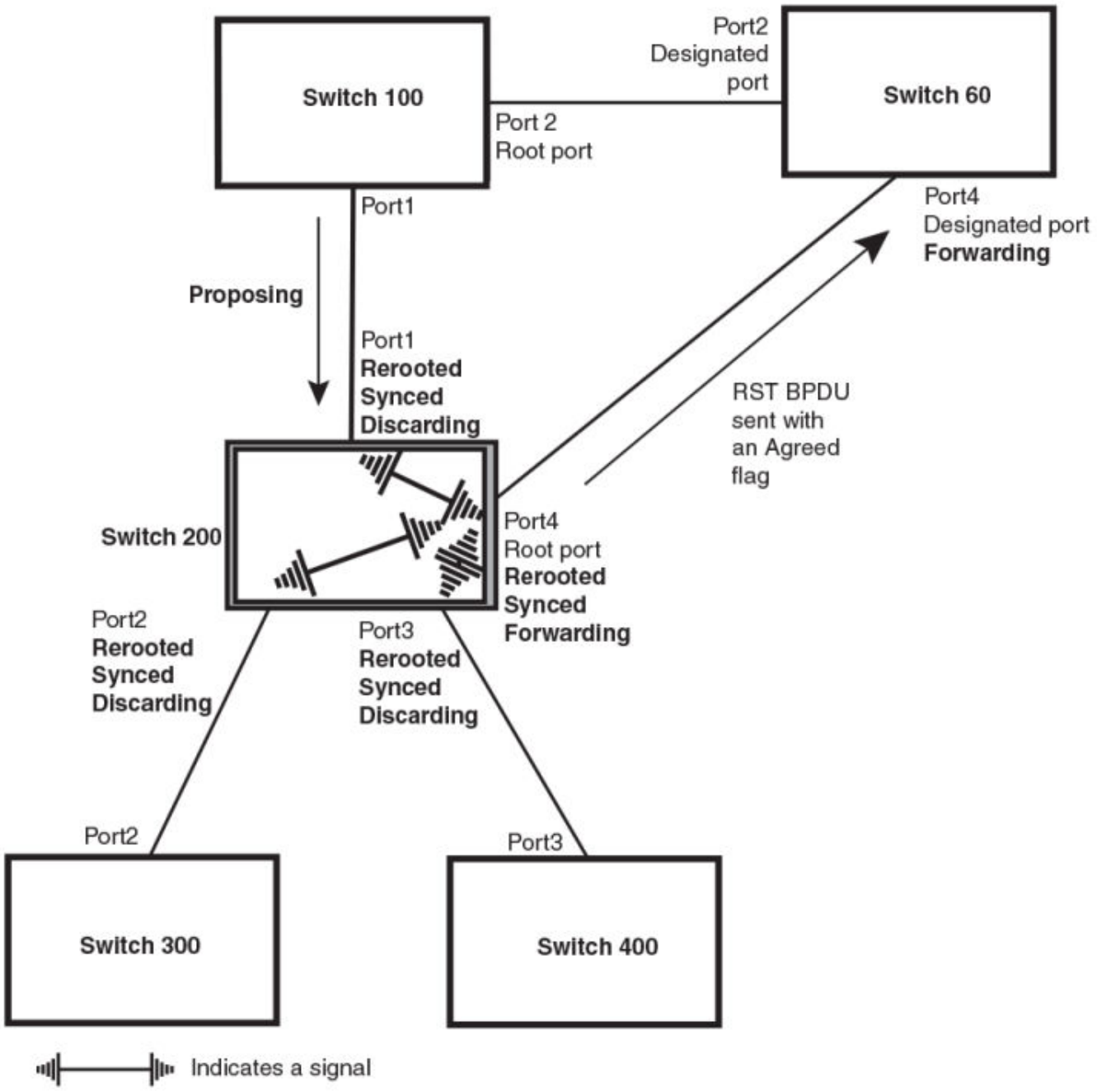
FIGURE 63 Sync and Rerooted



NOTE
Port numbers are simplified.

- Synced and Agree: When all the ports on the bridge assert their synced signals, the new Root port asserts its own synced signal and sends an RST BPDU to Port4/Switch 60 that contains an agreed flag as shown in the following figure. The Root port also moves into a forwarding state.

FIGURE 64 Rerooted, Synced, and Agreed

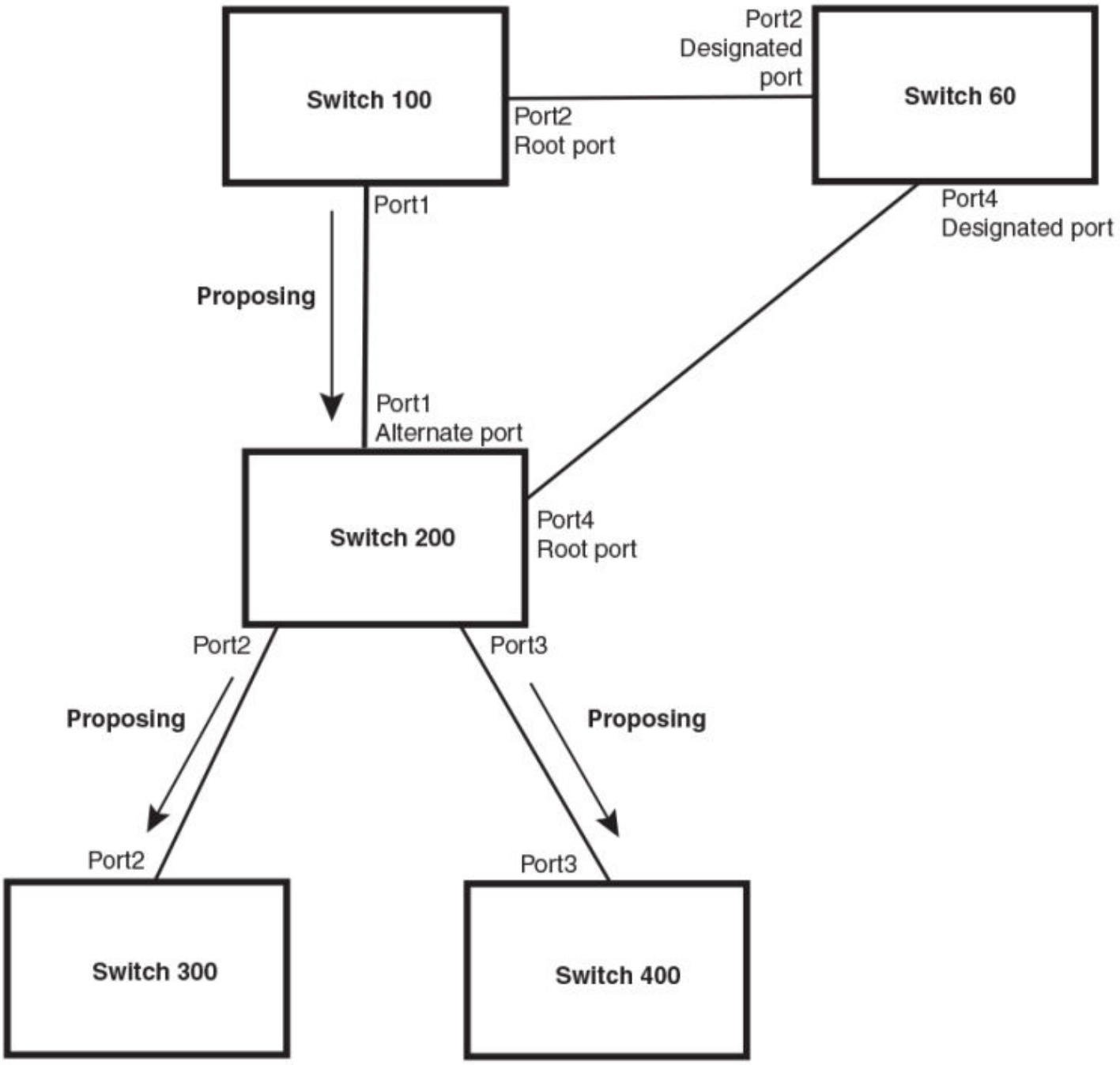


NOTE
Port numbers are simplified.

The old Root port on Switch 200 becomes an Alternate Port as shown in the following figure. Other ports on that bridge are elected to appropriate roles.

The Designated port on Switch 60 goes into a forwarding state once it receives the RST BPDU with the agreed flag.

FIGURE 65 Handshake Completed After Election of New Root Port



NOTE
Port numbers are simplified.

Recall that Switch 200 sent the agreed flag to Port4/Switch 60 and not to Port1/Switch 100 (the port that connects Switch 100 to Switch 200). Therefore, Port1/Switch 100 does not go into a forwarding state instantly. It waits until two instances of the forward delay timer expires on the port before it goes into a forwarding state.

At this point the handshake between the Switch 60 and Switch 200 is complete.

The remaining bridges (Switch 300 and Switch 400) may have to go through the reroot handshake if a new Root port needs to be assigned.

IEEE 802.1w Convergence in a Simple Topology

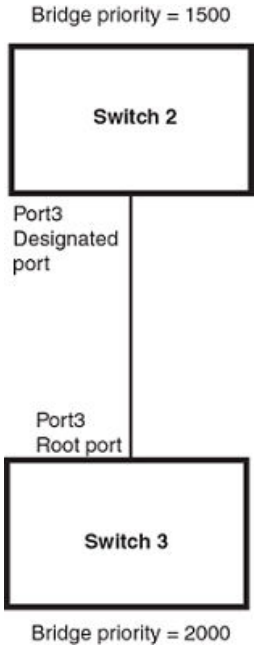
The examples in this section illustrate how IEEE 802.1w convergence occurs in a simple Layer 2 topology at start-up.

NOTE
The remaining examples assume that the appropriate handshake mechanisms occur as port roles and states change.

Convergence at Start Up

In the following figure, two bridges Switch 2 and Switch 3 are powered up. There are point-to-point connections between Port3/Switch 2 and Port3/Switch 3.

FIGURE 66 Convergence Between Two Bridges



NOTE
Port numbers are simplified.

At power up, all ports on Switch 2 and Switch 3 assume Designated port roles and are in discarding states before they receive any RST BPDU.

Port3/Switch 2, with a Designated role, transmits an RST BPDU with a proposal flag to Port3/Switch 3. A port with a Designated role sends the proposal flag in its RST BPDU when it is ready to move to a forwarding state.

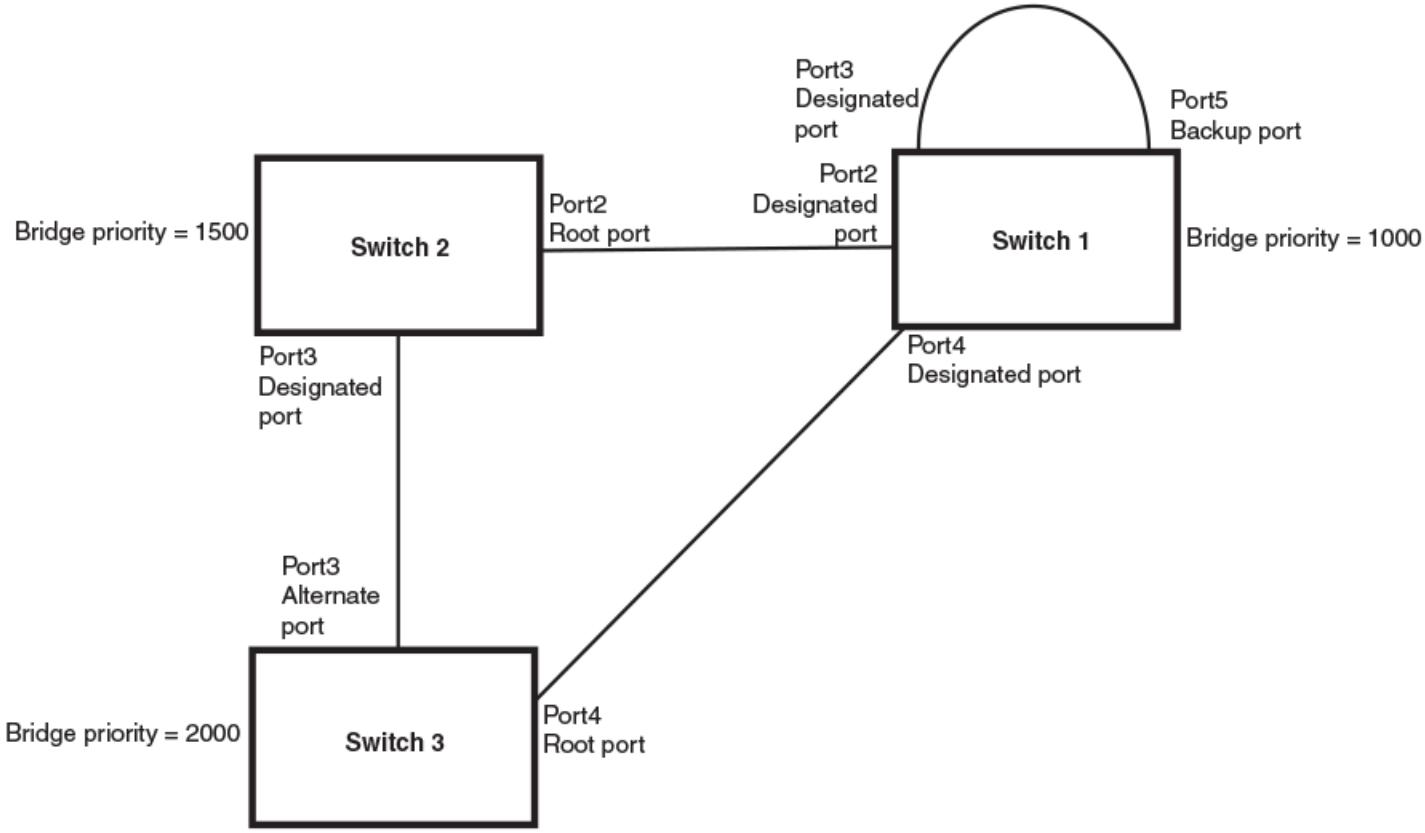
Port3/Switch 3, which starts with a role of Designated port, receives the RST BPDU and finds that it is superior to what it can transmit; therefore, Port3/Switch 3 assumes a new port role, that of a Root port. Port3/Switch 3 transmits an RST BPDU with an agreed flag back to Switch 2 and immediately goes into a forwarding state.

Port3/Switch 2 receives the RST BPDU from Port3/Switch 3 and immediately goes into a forwarding state.

Now, IEEE 802.1w has fully converged between the two bridges, with Port3/Switch 3 as an operational Root port in a forwarding state and Port3/Switch 2 as an operational Designated port in a forwarding state.

Next, Switch 1 is powered up. See the following figure.

FIGURE 67 Simple Layer 2 Topology



NOTE
Port numbers are simplified.

The point-to-point connections between the three bridges are as follows:

- Port2/Switch 1 and Port2/Switch 2
- Port4/Switch 1 and Port4/Switch 3
- Port3/Switch 2 and Port3/Switch 3

Ports 3 and 5 on Switch 1 are physically connected.

At start up, the ports on Switch 1 assume Designated port roles, which are in discarding state. The ports begin sending RST BPDUs with proposal flags. The flags indicate the ID of the bridge that the ports belong to, and the bridge that the ports understand to be the root bridge. The switch that eventually becomes the downstream neighbor is the only switch that sends a BPDU with the agreement bit set.

When Port4/Switch 3 receives these RST BPDUs, the IEEE 802.1w algorithm determines that they are better than the RST BPDUs that were previously received on Port3/Switch 3. Port4/Switch 3 is now selected as Root port. This new assignment signals Port3/Switch 3 to begin entering the discarding state and to assume an Alternate port role. As it goes through the transition, Port3/Switch 3 negotiates a new role and state with its peer port, Port3/Switch 2.

Port4/Switch 3 sends an RST BPDU with an agreed flag to Port4/Switch 1. Both ports go into forwarding states.

Port2/Switch 2 receives an RST BPDU. The IEEE 802.1w algorithm evaluates the BPDU and determines that it is superior to any BPDU that any other port on Switch 2 can transmit. Port2/Switch 2 assumes the role of a Root port.

The new Root port then signals all ports on the bridge to start synchronization. Since none of the ports are Edge ports, they all enter the discarding state and assume the role of Designated ports. Port3/Switch 2, which previously had a Designated role with a forwarding state, starts the discarding state. They also negotiate port roles and states with their peer ports. Port3/Switch 2 also sends an RST BPU to Port3/Switch 3 with a proposal flag to request permission go into a forwarding state.

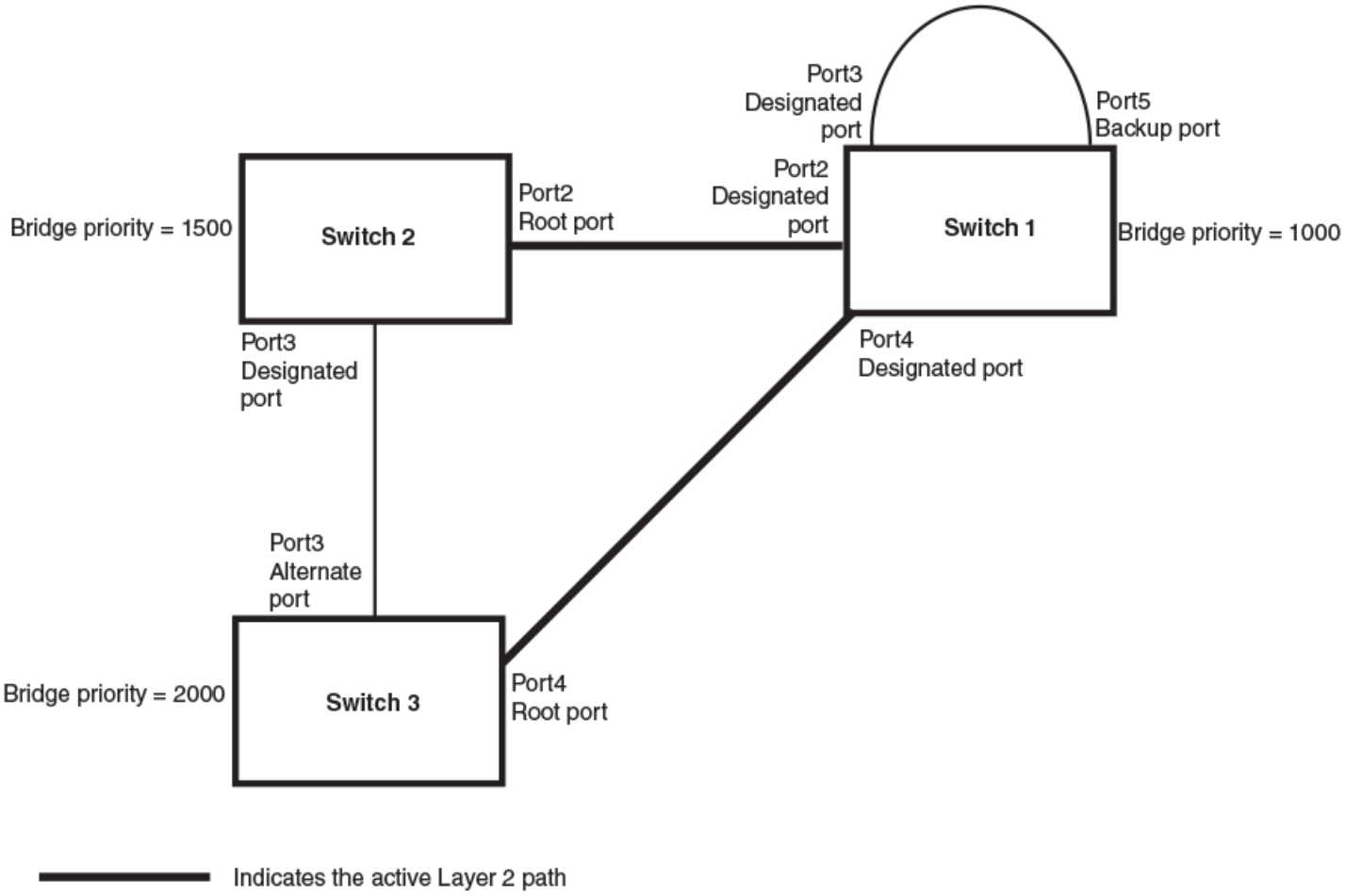
The Port2/Switch 2 bridge also sends an RST BPDU with an agreed flag Port2/Switch 1 that Port2 is the new Root port. Both ports go into forwarding states.

Now, Port3/Switch 3 is currently in a discarding state and is negotiating a port role. It received RST BPDUs from Port3/Switch 2. The IEEE 802.1w algorithm determines that the RST BPDUs Port3/Switch 3 received are superior to those it can transmit; however, they are not superior to those that are currently being received by the current Root port (Port4). Therefore, Port3 retains the role of Alternate port.

Ports3/Switch 1 and Port5/Switch 1 are physically connected. Port5/Switch 1 received RST BPDUs that are superior to those received on Port3/Switch 1; therefore, Port5/Switch 1 is given the Backup port role while Port3 is given the Designated port role. Port3/Switch 1 does not go directly into a forwarding state. It waits until the forward delay time expires twice on that port before it can proceed to the forwarding state.

Once convergence is achieved, the active Layer 2 forwarding path converges as shown in the following figure.

FIGURE 68 Active Layer 2 Path

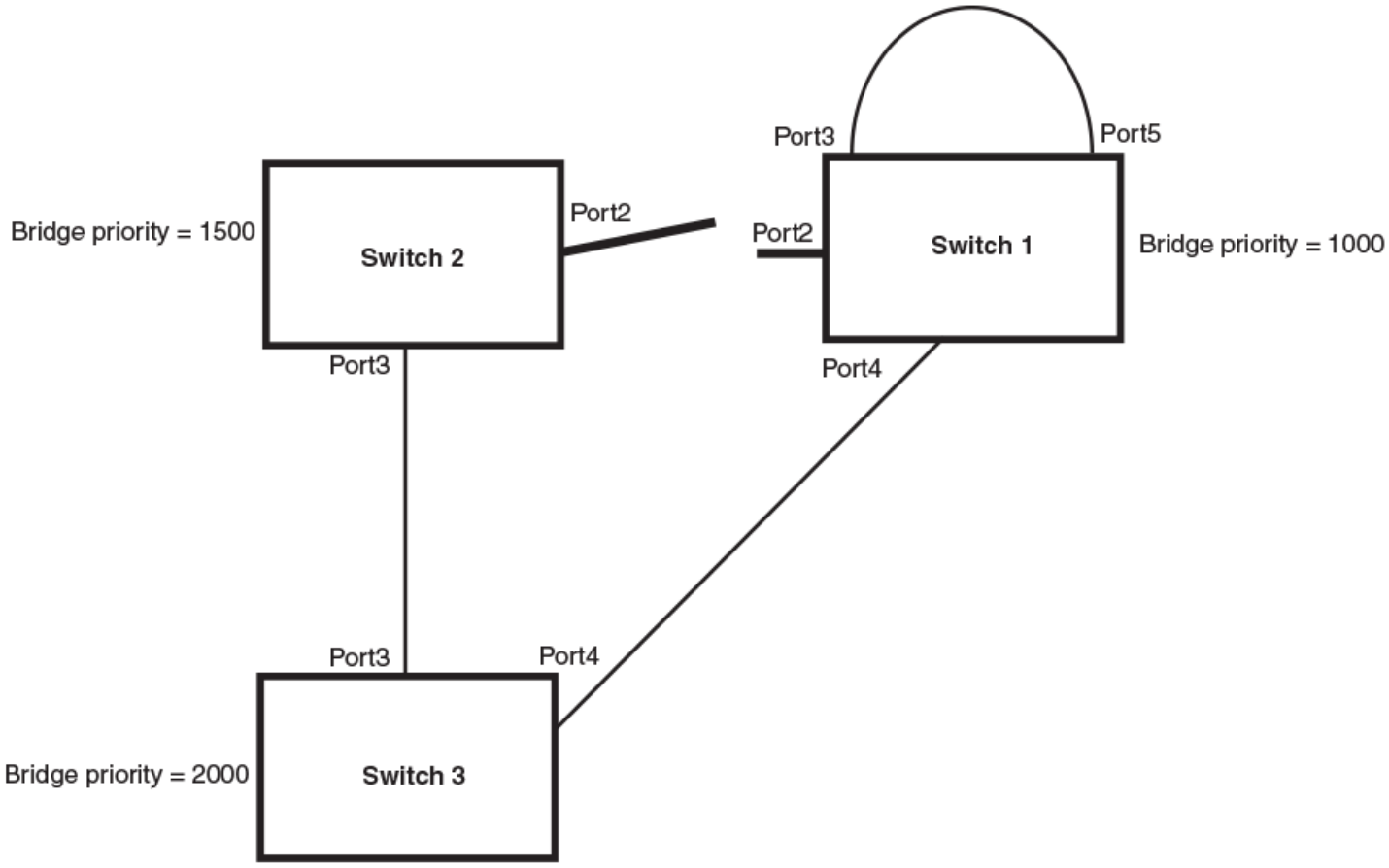


NOTE
Port numbers are simplified.

Convergence after a Link Failure

The following figure illustrates a link failure in the IEEE 802.1w topology. In this example, Port2/Switch2, which is the port that connects Switch 2 to the root bridge (Switch 1), failed and both Switch 2 and Switch 1 are affected by the topology change.

FIGURE 69 Link Failure in the Topology



NOTE
Port numbers are simplified.

Switch 1 sets its Port2 into a discarding state.

At the same time, Switch 2 assumes the role of a root bridge since its root port failed and it has no operational Alternate port. Port3/Switch 2, which currently has a Designated port role, sends an RST BPDU to Switch 3. The RST BPDU contains a proposal flag and a bridge ID of Switch 2 as its root bridge ID.

When Port3/Switch 3 receives the RST BPDUs, the IEEE 802.1w algorithm determines that they are inferior to those that the port can transmit. Therefore, Port3/Switch 3 is given a new role, that of a Designated port. Port3/Switch 3 then sends an RST BPDU with a proposal flag to Switch 2, along with the new role information. However, the root bridge ID transmitted in the RST BPDU is still Switch 1.

When Port3/Switch 2 receives the RST BPDU, the IEEE 802.1w algorithm determines that it is superior to the RST BPDU that it can transmit; therefore, Port3/Switch 2 receives a new role; that of a Root port. Port3/Switch 2 then sends an RST BPDU with an agreed flag to Port3/Switch 3. Port3/Switch 2 goes into a forwarding state.

When Port3/Switch 3 receives the RST BPDU that Port3/Switch 2 sent, Port3/Switch 3 changes into a forwarding state, which then completes the full convergence of the topology.

Convergence at Link Restoration

When Port2/Switch 2 is restored, both Switch 2 and Switch 1 recognize the change. Port2/Switch 1 starts assuming the role of a Designated port and sends an RST BPDU containing a proposal flag to Port2/Switch 2.

When Port2/Switch 2 receives the RST BPDUs, the IEEE 802.1w algorithm determines that the RST BPDUs the port received are better than those received on Port3/Switch 3; therefore, Port2/Switch 2 is given the role of a Root port. All the ports on Switch 2 are informed that a new Root port has been assigned which then signals all the ports to synchronize their roles and states. Port3/Switch 2, which was the previous Root port, enters a discarding state and negotiates with other ports on the bridge to establish its new role and state, until it finally assumes the role of a Designated port.

Next, the following happens:

- Port3/Switch 2, the Designated port, sends an RST BPDU, with a proposal flag to Port3/Switch 3.
- Port2/Switch 2 also sends an RST BPDU with an agreed flag to Port2/Switch 1 and then places itself into a forwarding state.

When Port2/Switch 1 receives the RST BPDU with an agreed flag sent by Port2/Switch 2, it puts that port into a forwarding state. The topology is now fully converged.

When Port3/Switch 3 receives the RST BPDU that Port3/Switch 2 sent, the IEEE 802.1w algorithm determines that these RST BPDUs are superior to those that Port3/Switch 3 can transmit. Therefore, Port3/Switch 3 is given a new role, that of an Alternate port. Port3/Switch 3 immediately enters a discarding state.

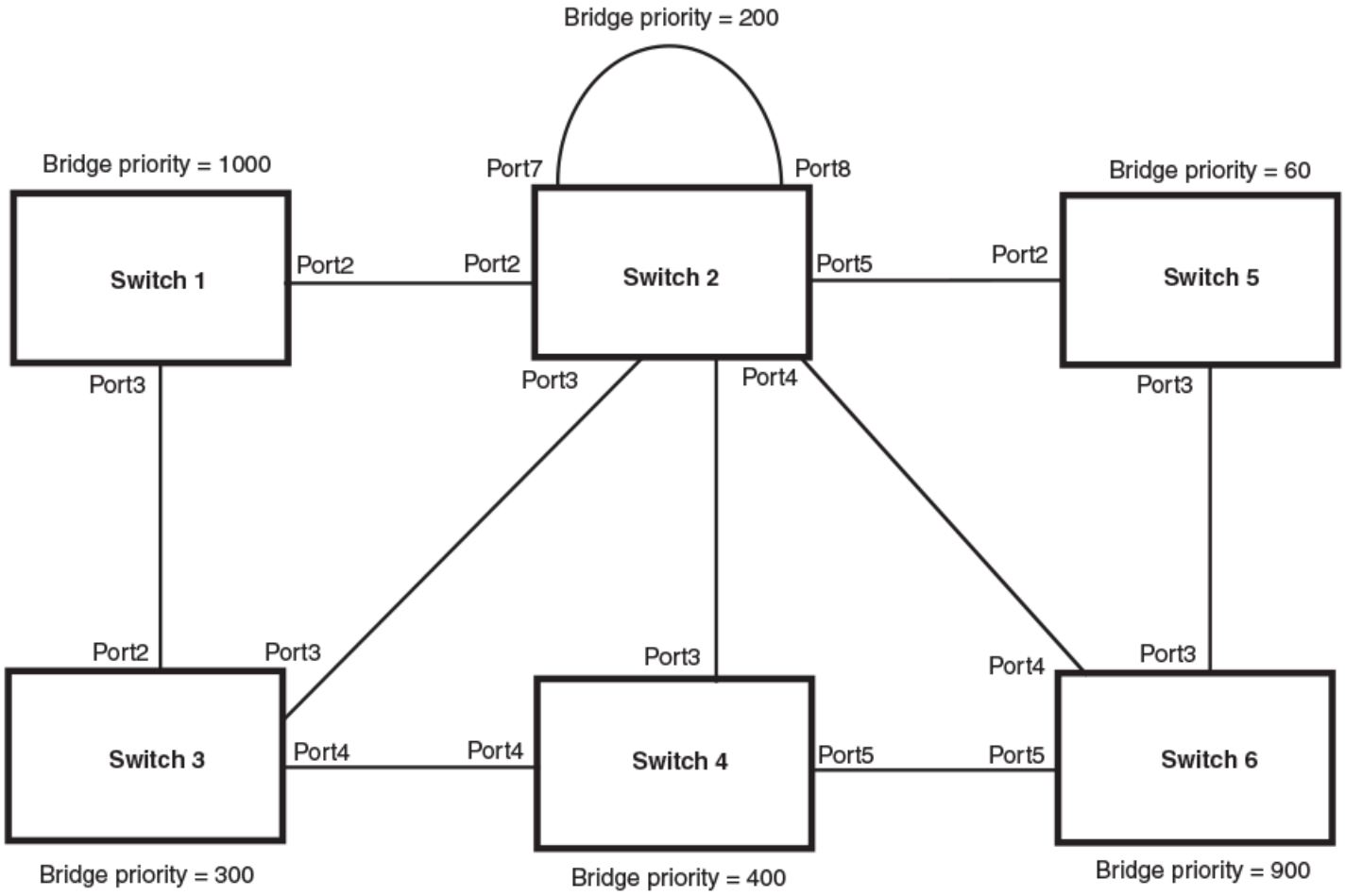
Now, Port3/Switch 2 does not go into a forwarding state instantly like the Root port. It waits until the forward delay timer expires twice on that port while it is still in a Designated role, before it can proceed to the forwarding state. The wait, however, does not cause a denial of service, since the essential connectivity in the topology has already been established.

When fully restored, the topology is the same as that shown on Figure 67 on page 201.

Convergence in a Complex IEEE 802.1w Topology

The following figure illustrates a complex IEEE 802.1w topology.

FIGURE 70 Complex IEEE 802.1w Topology



NOTE
Port numbers are simplified.

In the above figure, Switch 5 is selected as the root bridge since it is the bridge with the highest priority. Lines in the figure show the point-to-point connection to the bridges in the topology.

Switch 5 sends an RST BPDU that contains a proposal flag to Port5/Switch 2. When handshakes are completed in Switch 5, Port5/Switch 2 is selected as the Root port on Switch 2. All other ports on Switch 2 are given Designated port role with discarding states.

Port5/Switch 2 then sends an RST BPDU with an agreed flag to Switch 5 to confirm that it is the new Root port and the port enters a forwarding state. Port7 and Port8 are informed of the identity of the new Root port. The IEEE 802.1w algorithm selects Port7 as the Designated port while Port8 becomes the Backup port.

Port3/Switch 5 sends an RST BPDU to Port3/Switch 6 with a proposal flag. When Port3/Switch 5 receives the RST BPDU, handshake mechanisms select Port3 as the Root port of Switch 6. All other ports are given a Designated port role with discarding states. Port3/Switch 6 then sends an RST BPDU with an agreed flag to Port3/Switch 5 to confirm that it is the Root port. The Root port then goes into a forwarding state.

Now, Port4/Switch 6 receives RST BPDUs that are superior to what it can transmit; therefore, it is given the Alternate port role. The port remains in discarding state.

Port5/Switch 6 receives RST BPDUs that are inferior to what it can transmit. The port is then given a Designated port role.

Next, Switch 2 sends RST BPDUs with a proposal flag to Port3/Switch 4. Port3 becomes the Root port for the bridge; all other ports are given a Designated port role with discarding states. Port3/Switch 4 sends an RST BPDU with an agreed flag to Switch 2 to confirm that it is the new Root port. The port then goes into a forwarding state.

Now, Port4/Switch 4 receives an RST BPDU that is superior to what it can transmit. The port is then given an Alternate port role and remains in discarding state.

Likewise, Port5/Switch 4 receives an RST BPDU that is superior to what it can transmit. The port is also given an Alternate port role and remains in discarding state.

Port2/Switch 2 transmits an RST BPDU with a proposal flag to Port2/Switch 1. Port2/Switch 1 becomes the Root port. All other ports on Switch 1 are given Designated port roles with discarding states.

Port2/Switch 1 sends an RST BPDU with an agreed flag to Port2/Switch 2 and Port2/Switch 1 goes into a forwarding state.

Port3/Switch 1 receives an RST BPDUs that is inferior to what it can transmit; therefore, the port retains its Designated port role and goes into forwarding state only after the forward delay timer expires twice on that port while it is still in a Designated role.

Port3/Switch 2 sends an RST BPDU to Port3/Switch 3 that contains a proposal flag. Port3/Switch 3 becomes the Root port, while all other ports on Switch 3 are given Designated port roles and go into discarding states. Port3/Switch 3 sends an RST BPDU with an agreed flag to Port3/Switch 2 and Port3/Switch 3 goes into a forwarding state.

Now, Port2/Switch 3 receives an RST BPDUs that is superior to what it can transmit so that port is given an Alternate port state.

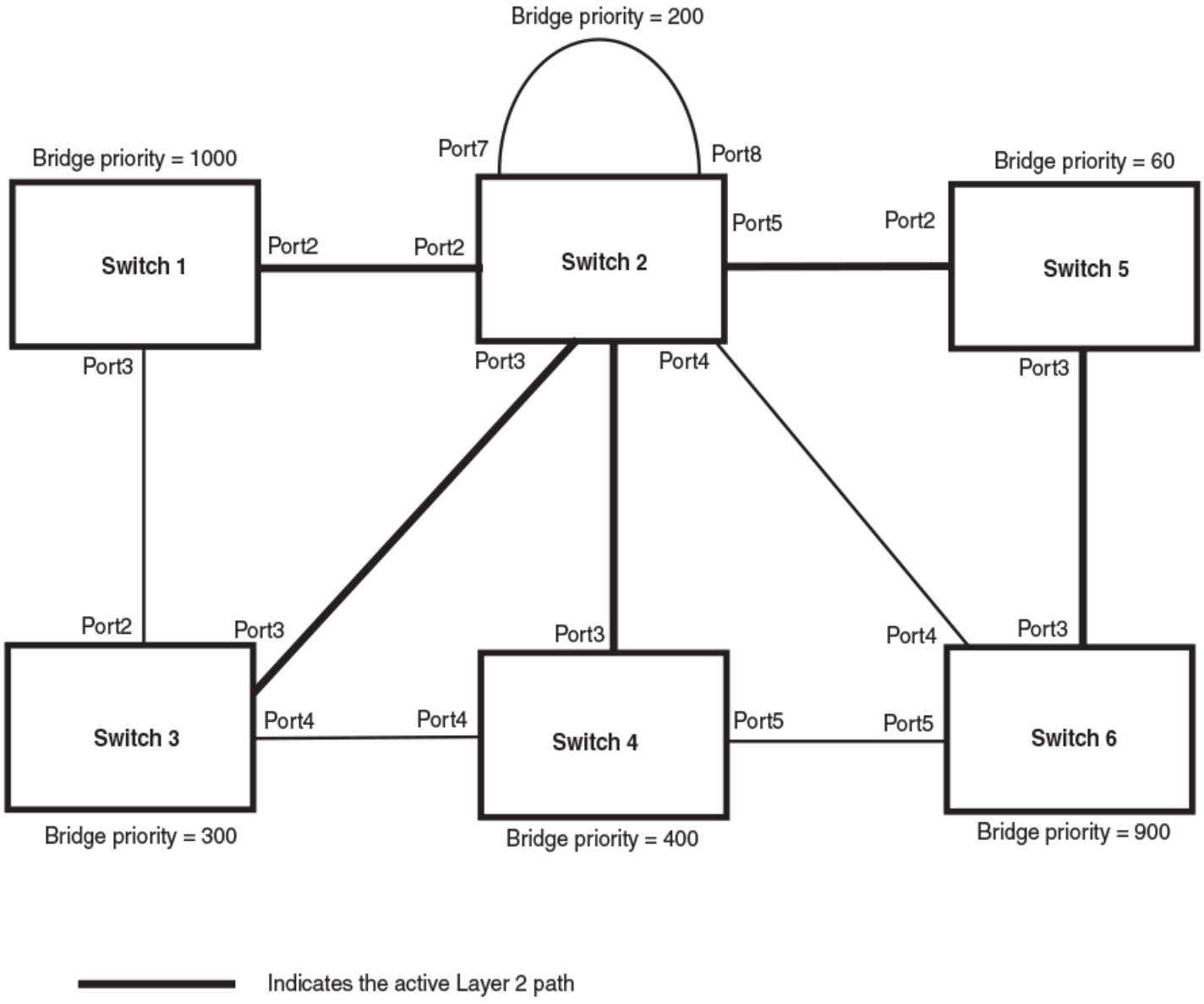
Port4/Switch 3 receives an RST BPDU that is inferior to what it can transmit; therefore, the port retains its Designated port role.

Ports on all the bridges in the topology with Designated port roles that received RST BPDUs with agreed flags go into forwarding states instantly. However, Designated ports that did not receive RST BPDUs with agreed flags must wait until the forward delay timer expires twice on those ports. Only then will these ports move into forwarding states.

The entire IEEE 802.1w topology converges in less than 300 msec and the essential connectivity is established between the designated ports and their connected root ports.

After convergence is complete, the following figure shows the active Layer 2 path of the topology.

FIGURE 71 Active Layer 2 Path in Complex Topology



NOTE
Port numbers are simplified.

Propagation of Topology Change

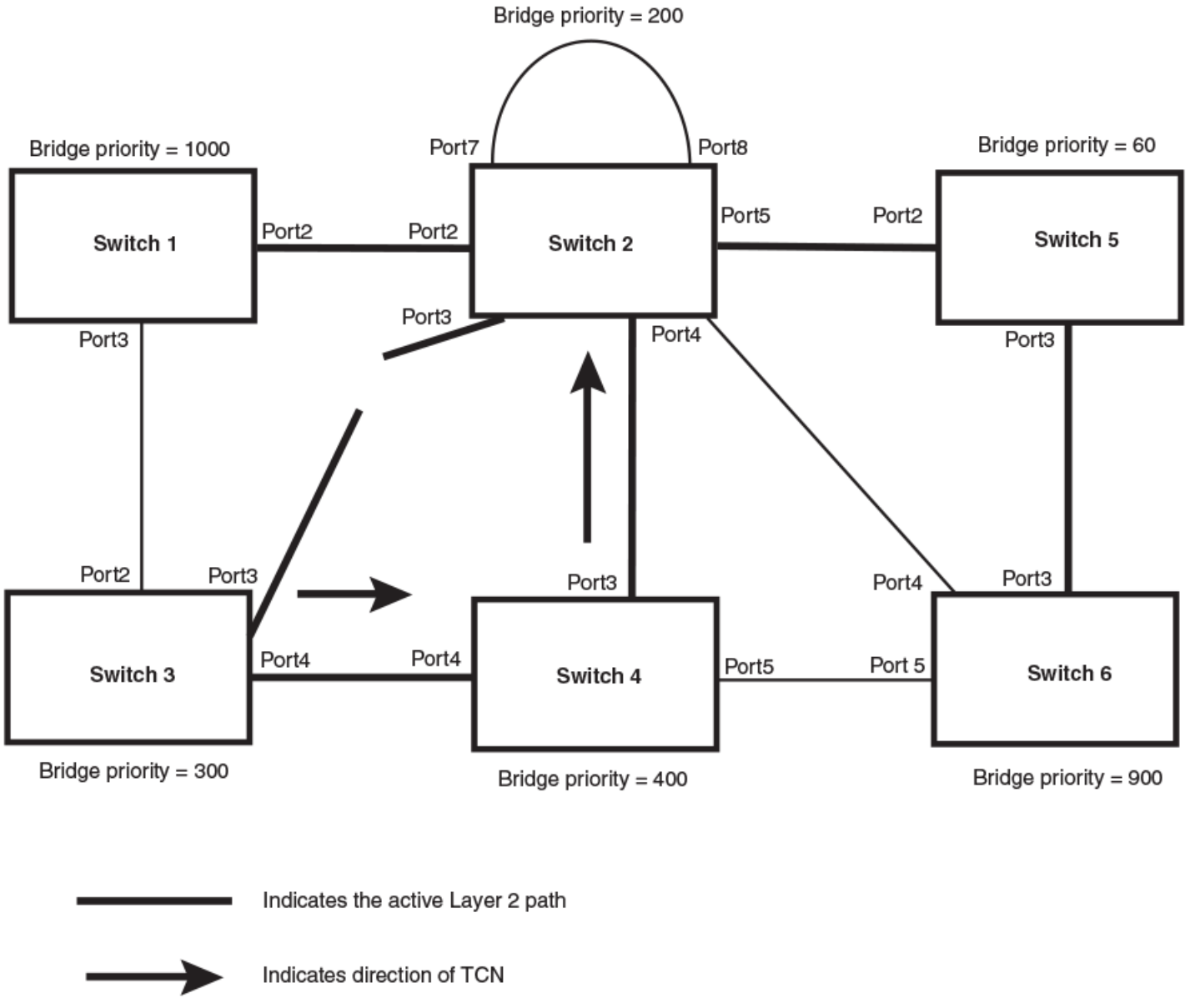
The Topology Change state machine generates and propagates the topology change notification messages on each port. When a Root port or a Designated port goes into a forwarding state, the Topology Change state machine on those ports sends a topology change notice (TCN) to all the bridges in the topology to propagate the topology change.

NOTE
Edge ports, alternate ports, or backup ports do not need to propagate a topology change.

The TCN is sent in the RST BPDU that a port sends. Ports on other bridges in the topology then acknowledge the topology change once they receive the RST BPDU, and send the TCN to other bridges until all the bridges are informed of the topology change.

For example, Port3/Switch 2 in the following figure, Port3/Switch 2 fails. Port4/Switch 3 becomes the new Root port. Port4/Switch 3 sends an RST BPDU with a TCN to Port4/Switch 4. To propagate the topology change, Port4/Switch 4 then starts a TCN timer on itself, on the bridge Root port, and on other ports on that bridge with a Designated role. Then Port3/Switch 4 sends RST BPDU with the TCN to Port4/Switch 2. Note the new active Layer 2 path in the following figure.

FIGURE 72 Beginning of Topology Change Notice

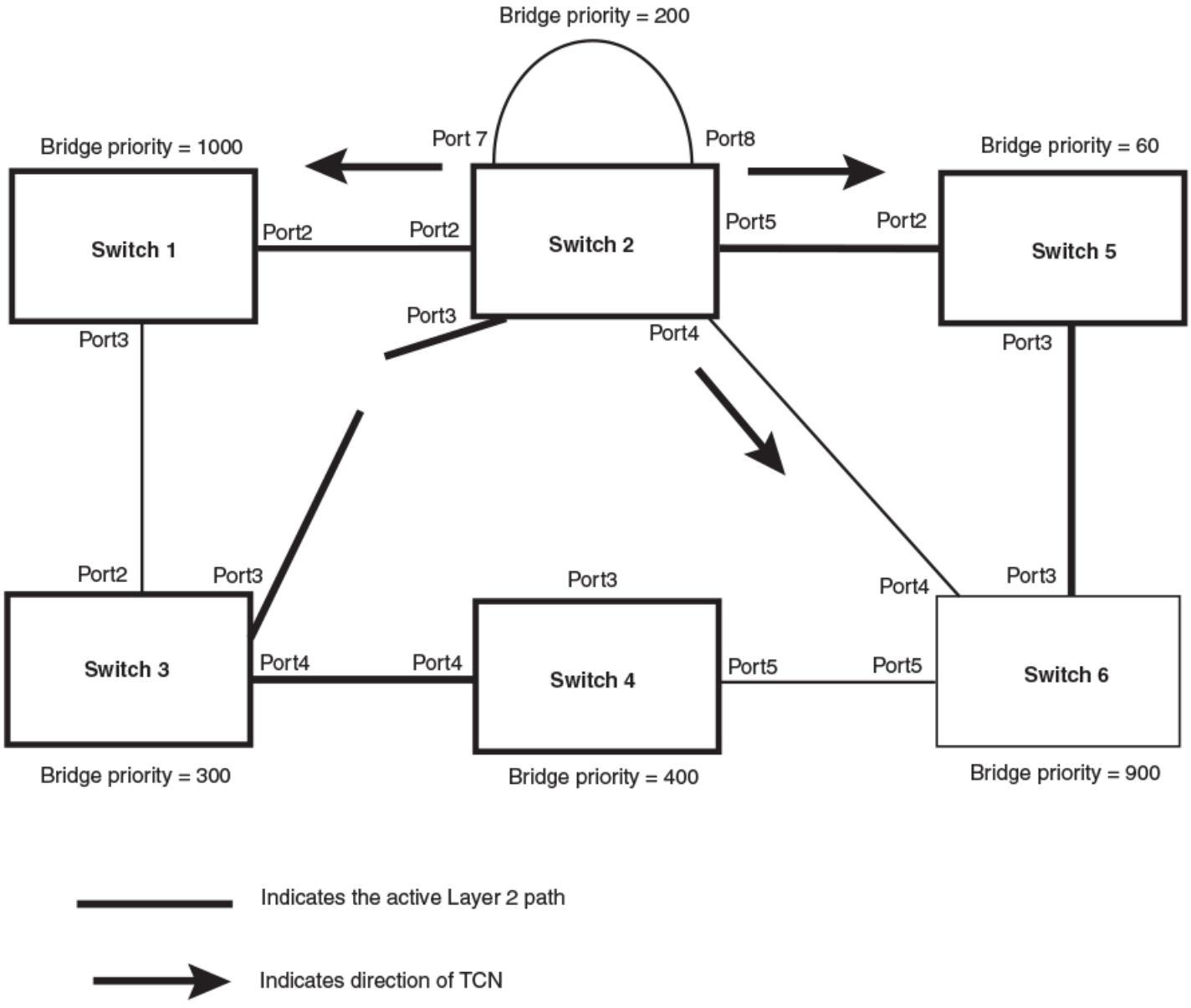


NOTE
Port numbers are simplified.

Switch 2 then starts the TCN timer on the Designated ports and sends RST BPDUs that contain the TCN as follows (Figure 73):

- Port5/Switch 2 sends the TCN to Port2/Switch 5
- Port4/Switch 2 sends the TCN to Port4/Switch 6
- Port2/Switch 2 sends the TCN to Port2/Switch 1

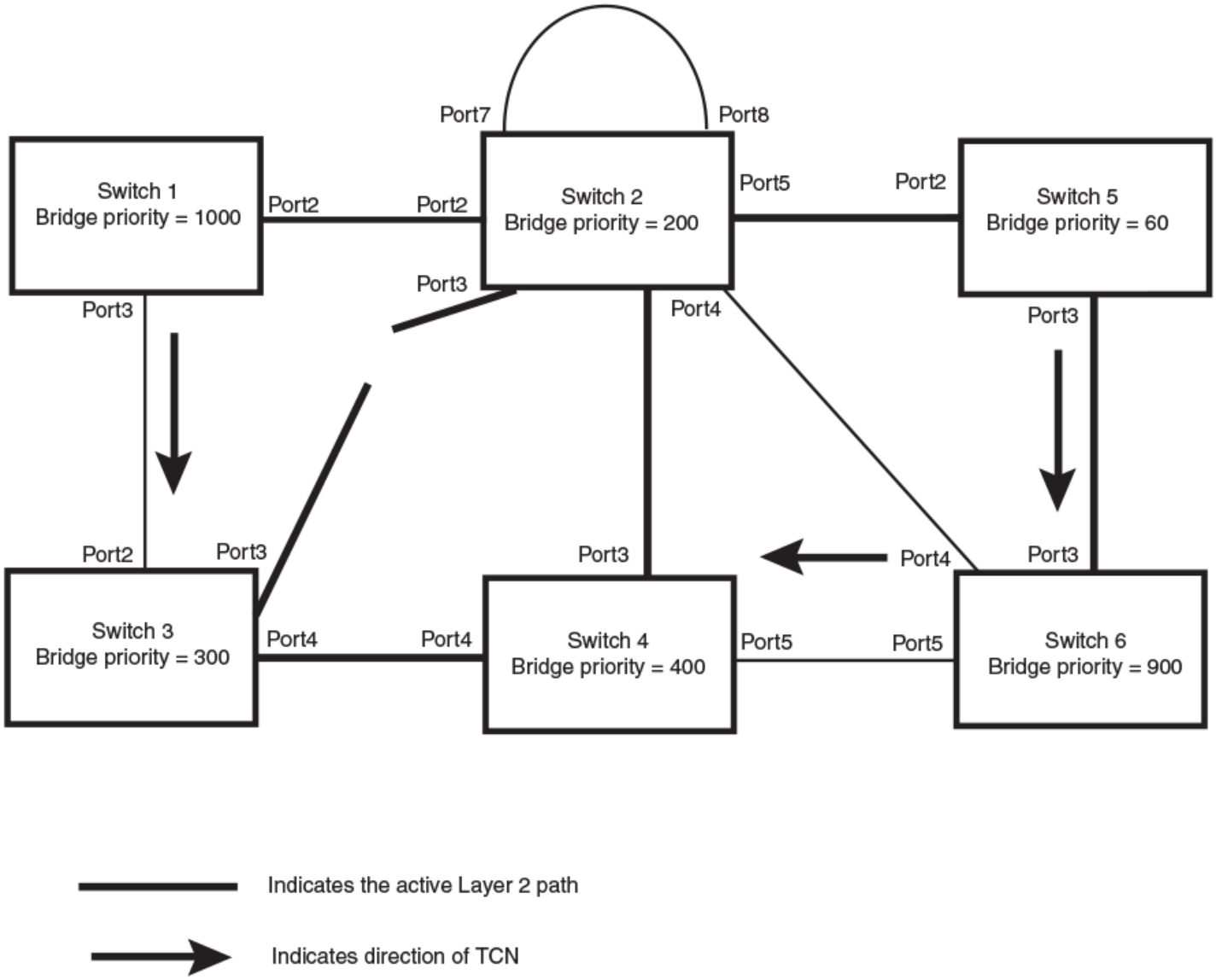
FIGURE 73 Sending TCN to Bridges Connected to Switch 2



NOTE
Port numbers are simplified.

Then Switch 1, Switch 5, and Switch 6 send RST BPDUs that contain the TCN to Switch 3 and Switch 4 to complete the TCN propagation, as shown in the following figure.

FIGURE 74 Completing the TCN Propagation



NOTE
Port numbers are simplified.

PVST/PVST+ Compatibility

The FastIron family of switches support Cisco's Per VLAN Spanning Tree plus (PVST+) protocol, by allowing the device to run multiple spanning trees while also interoperating with IEEE 802.1Q devices[3].

NOTE
RUCKUS device ports automatically detect PVST+ BPDUs and enable support for the BPDUs once detected. You do not need to perform any configuration steps to enable PVST+ support.

Support for Cisco's PVST+ protocol allows a RUCKUS device to run multiple spanning trees while also interoperating with IEEE 802.1Q devices. RUCKUS device ports automatically detect PVST+ BPDUs and enable support for the BPDUs once detected. The enhancement allows a port that is in PVST+ compatibility mode due to auto-detection to revert to the default multiple spanning trees mode when one of the following events occurs:

- The link is disconnected or broken
- The link is administratively disabled
- The link is disabled by interaction with the link-keepalive protocol

This enhancement allows a port that was originally interoperating with PVST+ to revert to multiple spanning trees when connected to a RUCKUS device.

Overview of PVST and PVST+

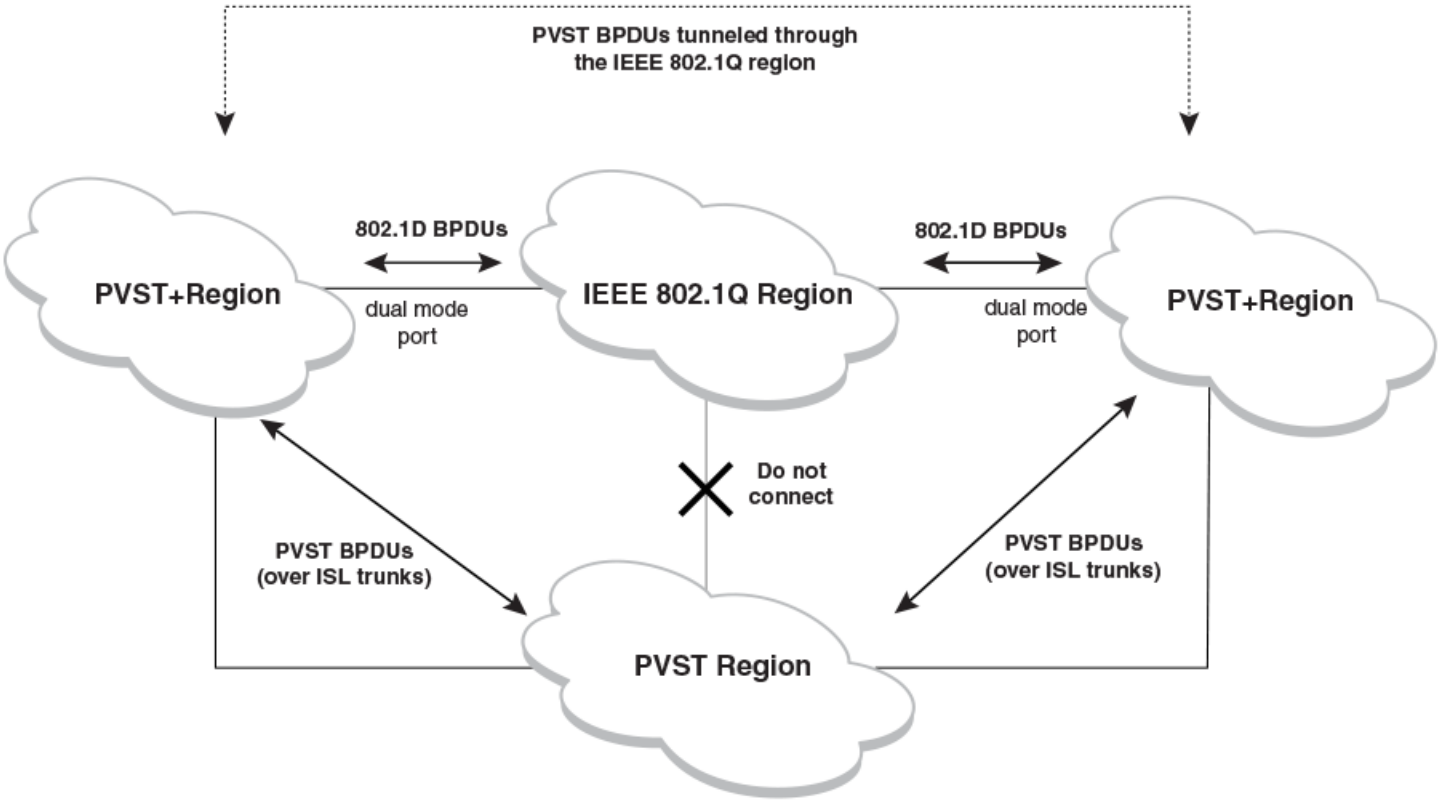
Per VLAN Spanning Tree (PVST) is a Cisco proprietary protocol that allows a Cisco device to have multiple spanning trees. The Cisco device can interoperate with spanning trees on other PVST devices but cannot interoperate with IEEE 802.1Q devices. An IEEE 802.1Q device has all its ports running a single spanning tree. PVST+ is an extension of PVST that allows a Cisco device to also interoperate with devices that are running a single spanning tree (IEEE 802.1Q).

Enhanced PVST+ support allows a RUCKUS device to interoperate with PVST spanning trees and the IEEE 802.1Q spanning tree at the same time.

IEEE 802.1Q and PVST regions cannot interoperate directly but can interoperate indirectly through PVST+ regions. PVST BPDUs are tunneled through IEEE 802.1Q regions, while PVST BPDUs for VLAN 1 (the IEEE 802.1Q VLAN) are processed by PVST+ regions. The following figure shows the interaction of IEEE 802.1Q, PVST, and PVST+ regions.

[3] Cisco user documentation for PVST/PVST+ refers to the IEEE 802.1Q spanning tree as the Common Spanning Tree (CST).

FIGURE 75 Interaction of IEEE 802.1Q, PVST, and PVST+ regions



Configuring PVST+ Support

PVST+ support is automatically enabled when the port receives a PVST BPDU. You can manually enable PVST+ support at any time or disable PVST+ support if desired.

A port that is in PVST+ compatibility mode due to auto-detection reverts to the default multiple spanning tree mode when one of the following events occurs:

- The link is disconnected or broken
- The link is administratively disabled
- The link is disabled by interaction with the link-keepalive protocol

This allows a port that was originally interoperating with PVST+ to revert to multiple spanning tree mode when connected to a RUCKUS device.

Enabling PVST+ Support Manually

To immediately enable PVST+ support on a port, enter the **pvst-mode** command in interface configuration mode

```
device(config)# interface ethernet 1/1/1
device(config-if-1/1/1)# pvst-mode
```

NOTE
If you disable PVST+ support, by entering the **no pvst-mode** command in interface configuration mode, the software still automatically enables PVST+ support if the port receives a BPDU with PVST+ format.

NOTE
If IEEE 802.1w and PVST+ (either by auto-detection or by explicit configuration) are enabled on a tagged VLAN port, IEEE 802.1w will treat the PVST BPDUs as legacy IEEE 802.1D BPDUs.

Displaying PVST+ Support Information

To display PVST+ information for ports on a RUCKUS device, enter the following command at any level of the CLI.

```
device# show span pvst-mode
PVST+ Enabled on:
Port          Method
1/1/1       Set by configuration
1/1/2       Set by configuration
1/2/10      Set by auto-detect
1/3/12      Set by configuration
1/4/24      Set by auto-detect
```

PVST+ Configuration Examples

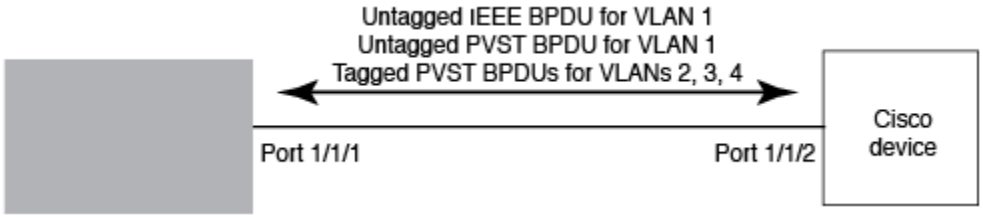
The following sections show examples for two common configurations:

- Tagged IEEE 802.1Q BPDUs on VLAN 1 and untagged BPDUs on another VLAN
- Untagged IEEE 802.1Q BPDUs on VLAN 1 and tagged PVST+ BPDUs on other VLANs

Tagged Port Using Default VLAN 1 as its Port Native VLAN

The following figure shows an example of a PVST+ configuration that uses VLAN 1 as the untagged default VLAN and VLANs 2, 3, and 4 as tagged VLANs.

FIGURE 76 Default VLAN 1 for Untagged BPDU



To implement this configuration, enter the following commands on your RUCKUS device.

```
device(config)# vlan-group 1 vlan 2 to 4
device(config-vlan-group-1)# tagged ethernet 1/1/1
device(config-vlan-group-1)# exit
device(config)# interface ethernet 1/1/1
device(config-if-1/1/1)# pvst-mode
```

These commands configure a VLAN group containing VLANs 2, 3, and 4, add port 1/1/1 as a tagged port to the VLANs, and enable PVST+ support on the port. The implicit dual-mode feature allows the port to send and receive untagged frames for the default VLAN (VLAN 1 in this case) in addition

to tagged frames for VLANs 2, 3, and 4. Enabling PVST+ support ensures that the port is ready to send and receive PVST+ BPDUs. If you do not manually enable PVST+ support, the support is not enabled until the port receives a PVST+ BPDU. The port can send and receive untagged frames for the default VLAN (VLAN 1 in this case) in addition to tagged frames for VLANs 2, 3, and 4.

The configuration leaves the default VLAN and the Port Native VLAN unchanged. The default VLAN is 1 and the Port Native VLAN also is 1.

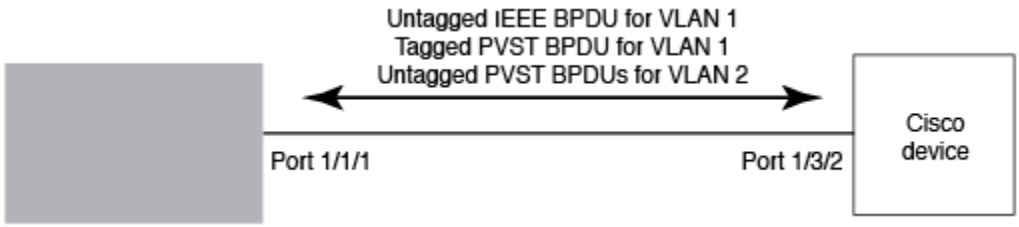
Port 1/1/1 will process BPDUs as follows:

- Process IEEE 802.1Q BPDUs for VLAN 1.
- Process tagged PVST BPDUs for VLANs 2, 3, and 4.
- Drop untagged PVST BPDUs for VLAN 1.

Untagged Port Using VLAN 2 as Port Native VLAN

The following figure shows an example in which a Port Native VLAN is not VLAN 1. In this case, VLAN 1 uses tagged frames and VLAN 2 uses untagged frames.

FIGURE 77 Port Native VLAN 2 for Untagged BPDUs



To implement this configuration, enter the following commands on your RUCKUS device.

```
device(config)# default-vlan-id 4000
device(config)# vlan 1
device(config-vlan-1)# tagged ethernet 1/1/1
device(config-vlan-1)# exit
device(config)# vlan 2
device(config-vlan-2)# untagged ethernet 1/1/1
device(config-vlan-2)# exit
device(config)# interface ethernet 1/1/1
device(config-if-1/1/1)# vlan 2
device(config-vlan-2)# untagged ethernet 1/1/1
device(config-if-1/1/1)# pvst-mode
device(config-if-1/1/1)# exit
```

These commands change the default VLAN ID, configure port 1/1/1 as a tagged member of VLAN 1 and as untagged member of VLAN 2, and enable PVST+ support on port 1/1/1. Since VLAN 1 is tagged in this configuration, the default VLAN ID must be changed from VLAN 1 to another VLAN ID. Changing the default VLAN ID from 1 to 2 allows the port to process tagged frames for VLAN 1 and to process untagged frames and untagged PVST BPDUs on VLAN 2.

Port 1/1/1 will process BPDUs as follows:

- Process IEEE 802.1Q BPDUs for VLAN 1.
- Process untagged PVST BPDUs for VLAN 2.
- Drop tagged PVST BPDUs for VLAN 1.

The following configuration is incorrect.

```
device(config)# default-vlan-id 1000
device(config)# vlan 1
device(config-vlan-1)# tagged ethernet 1/1/1 to 1/1/2
```

```
device(config-vlan-1)# exit
device(config)# interface ethernet 1/1/1
device(config-if-1/1/1)# pvst-mode
device(config-if-1/1/1)# exit
device(config)# interface ethernet 1/1/2
device(config-if-1/1/2)# pvst-mode
device(config-if-1/1/2)# exit
```

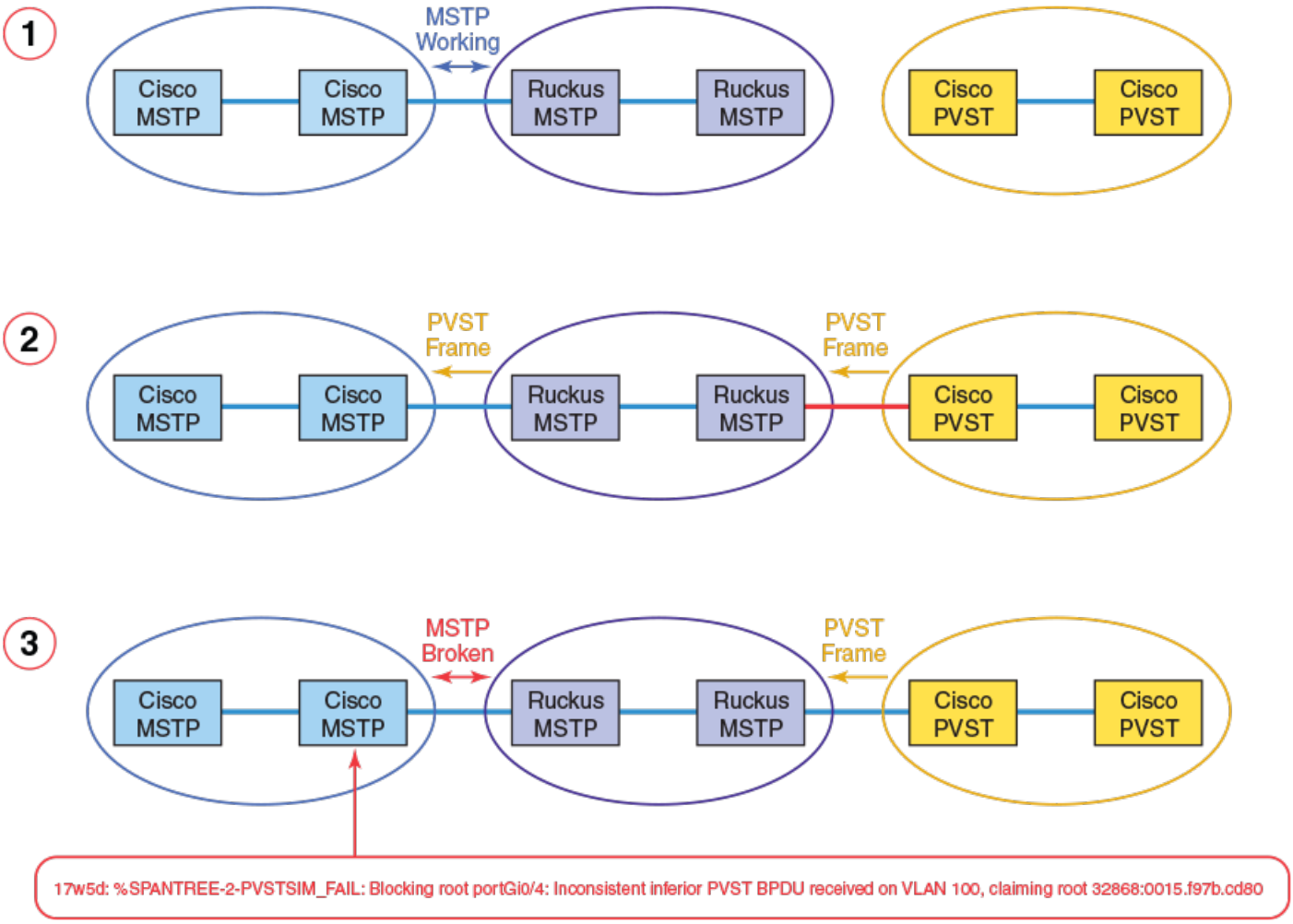
In the configuration above, all PVST BPDUs associated with VLAN 1 would be discarded. Since IEEE BPDUs associated with VLAN 1 are untagged, they are discarded because the ports in VLAN 1 are tagged. Effectively, the BPDUs are never processed by the Spanning Tree Protocol. STP assumes that there is no better bridge on the network and sets the ports to forwarding. This could cause a Layer 2 loop.

PVST+ Protect

If a PVST+ packet is received on a port configured for Multiple Spanning Tree Protocol (MSTP), the RUCKUS device floods it to all its ports in the VLAN so that it reaches other PVST+ devices across the VLAN. This flooding can sometimes cause a port to be blocked on the Cisco side. Use the PVST+ Protect feature to prevent this flooding, blocking the PVST+ BPDU and marking the port as ERR-DISABLED.

The following figure illustrates how a Cisco device running MSTP puts the port in a blocking state.

FIGURE 78 A Cisco Device Running MSTP Putting the Port in a Blocking State



The processes are summarized as follows:

1. RUCKUS and Cisco MSTP work correctly together, without any PVST devices in the topology.
2. A PVST device is connected. RUCKUS MSTP devices flood PVST frames across topology.
3. MSTP between RUCKUS and Cisco no longer works correctly, because Cisco assumes legacy PVST device is connected.

To configure PVST+ Protect, complete the following steps in any order:

- In global configuration mode, enter the **errdisable recovery cause** command and specify **pvstplus-protect** as the cause. If you do not enable automatic recovery, blocked ports will remain blocked.
- Optionally, in global configuration mode, enter the **errdisable recovery interval** command and specify a non-default recovery interval. (The default is 300 seconds.)
- In interface configuration mode, enter the **pvstplus-protect** command on an interface to be protected.

  NOTE
  The **pvstplus-protect** command cannot be issued concurrently with the **pvst-mode** command. The following error message appears:

  ```
  PVST mode not allowed on a PVST+ Protect mode
  ```

To enable error recovery globally:

```
device(config)# errdisable recovery cause pvstplus-protect
```

To change the recovery interval from the default, use the **errdisable recovery interval** command.

```
device(config)# errdisable recovery interval 150
```

To confirm the error recovery status, use the **show errdisable recovery** command.

```
device# show errdisable recovery
ErrDisable Reason                                                    Timer Status
-------------------------------------------------------------------------------
all reason                                                           Disabled
bpduguard                                                            Disabled
loopDetection                                                        Disabled
invalid license                                                      Disabled
packet-inerror                                                       Disabled
loam-critical-event                                                  Disabled
Reload the switch or stack to enable this port in 10G speed          Disabled
stack-port-resiliency                                                Disabled
broadcast traffic threshold exceeded                                 Disabled
multicast traffic threshold exceeded                                 Disabled
unknown unicast traffic threshold exceeded                           Disabled
pvstplus-protect                                                     Enabled
Timeout Value: 60 seconds
Interface that will be enabled at the next timeout:
Interface         Errdisable reason   Time left (sec)
--------------    -----------------   ---------------
Port 1/1/1         pvstplus-protect         31
```

To enable PVST+ Protect on a single port, use the **pvstplus-protect** command.

```
device(config)# interface ethernet 1/1/1
device(config-if-1/1/1)# pvstplus-protect
```

To confirm the running configuration on a specified Ethernet interface, use the **show running-config interface ethernet** command.

```
device# show running-config interface ethernet 1/1/1
interface ethernet 1/1/1
 port-name ToCisco1
 pvstplus-protect
```

To display the status of PVST+ Protect on the Ethernet interface, including the number of dropped PVST+ BPDUs:, use the **show pvstplus-protect-ports** command.

```
device# show pvstplus-protect-ports ethernet 1/1/1
Port    PVST Drop Count
        1/1/1  2
```

To enable PVST+ Protect on a range of ports in interface configuration mode, use the **pvstplus-protect** command.

```
device(config)# interface ethernet 1/1/1 to 1/1/4
device(config-mif-1/1/1-1/1/4)# pvstplus-protect
```

To confirm the configuration on a specified Ethernet interface, use the **show interface ethernet** command.

```
device# show interface ethernet 1/1/1
GigabitEthernet1/1/1 is ERR-DISABLED (pvstplus-protect), line protocol is down
  Port down for 3 second(s)
  Hardware is GigabitEthernet, address is cc4e.2407.affe (bia cc4e.2407.affe)
  Configured speed auto, actual unknown, configured duplex fdx, actual unknown
  Configured mdi mode AUTO, actual unknown
  Tagged member of 7 L2 VLANs, untagged in VLAN 1, port state is DISABLED
  BPDU guard is Disabled, ROOT protect is Disabled, Designated protect is Disabl
ed
  Link Error Dampening is Disabled
  STP configured to ON, priority is level0, mac-learning is enabled
  Flow Control is config enabled, oper disabled, negotiation disabled
  Mirror disabled, Monitor disabled
  Mac-notification is disabled
  Not member of any active trunks
  Not member of any configured trunks
  Port name is ToCisco1
  Inter-Packet Gap (IPG) is 96 bit times
  MTU 1500 bytes
  300 second input rate: 0 bits/sec, 0 packets/sec, 0.00% utilization
  300 second output rate: 0 bits/sec, 0 packets/sec, 0.00% utilization
  8027 packets input, 561171 bytes, 0 no buffer
  Received 0 broadcasts, 8022 multicasts, 5 unicasts
  0 input errors, 0 CRC, 0 frame, 0 ignored
  0 runts, 0 giants
  2487 packets output, 420635 bytes, 0 underruns
  Transmitted 0 broadcasts, 2487 multicasts, 0 unicasts
  0 output errors, 0 collisions
  Relay Agent Information option: Disabled
Egress queues:
Queue counters    Queued packets    Dropped Packets
    0                   0                   0
    1                   0                   0
    2                   0                   0
    3                   0                   0
    4                   0                   0
    5                   0                   0
    6                   0                   0
    7                   0                   0
```

To view the logging status, use the **show logging** command.

```
device# show logging
Syslog logging: enabled ( 0 messages dropped, 0 flushes, 226 overruns)
    Buffer logging: level ACDMEINW, 50 messages logged
    level code: A=alert C=critical D=debugging M=emergency E=error
                I=informational N=notification W=warning
Static Log Buffer:
Dec 31 18:00:40:I:System: Stack unit 1 POE  PS 1, Internal Power supply  with 68
000 mwatts capacity is up
Dynamic Log Buffer (50 lines):
Jan  4 13:49:49:I:System: Interface ethernet 1/1/1, state down
Jan  4 13:49:49:I:MSTP: MST 0 Port 1/1/1 - DISCARDING
Jan  4 13:49:49:I:MSTP: MST 2 Port 1/1/1 - DISCARDING
Jan  4 13:49:49:I:MSTP: MST 1 Port 1/1/1 - DISCARDING
Jan  4 13:49:49:I:PVST: Received PVST+ BPDU on PVST+ Protect enabled Port 1/1/1
, Vlan 100. Error Disabling

<---output omitted--->
```

To clear the PVST+ Protect statistics for one or more specified Ethernet ports:

```
device# clear pvstplus-protect-statistics ethernet 1/1/1
```

To clear the PVST+ Protect statistics on a range of Ethernet interfaces:

```
device# clear pvstplus-protect-statistics ethernet 1/1/1 to 1/1/4
```

PVRST Compatibility

PVRST, the "rapid" version of per-VLAN spanning tree (PVST), is a Cisco proprietary protocol. PVRST corresponds to the RUCKUS full implementation of IEEE 802.1w (RSTP). Likewise, PVST, also a Cisco proprietary protocol, corresponds to the RUCKUS implementation of IEEE 802.1D (STP). When a RUCKUS device receives PVRST BPDUs on a port configured to run IEEE 802.1w, it recognizes and processes these BPDUs and continues to operate in IEEE 802.1w mode.

PVRST compatibility is automatically enabled when a port receives a PVRST BPDU.

BPDU Guard

In an STP environment, switches, end stations, and other Layer 2 devices use Bridge Protocol Data Units (BPDUs) to exchange information that STP will use to determine the best path for data flow.

The BPDU guard, an enhancement to STP, removes a node that reflects BPDUs back in the network. It enforces the STP domain borders and keeps the active topology predictable by not allowing any network devices behind a BPDU guard-enabled port to participate in STP.

In some instances, it is unnecessary for a connected device, such as an end station, to initiate or participate in an STP topology change. In this case, you can enable the STP BPDU guard feature on the RUCKUS port to which the end station is connected. STP BPDU guard shuts down the port and puts it into an errdisable state. This disables the connected device's ability to initiate or participate in an STP topology. A log message is then generated for a BPDU guard violation, and a CLI message is displayed to warn the network administrator of a severe invalid configuration. The BPDU guard feature provides a secure response to invalid configurations because the administrator must manually put the interface back in service if errdisable recovery is not enabled.

NOTE
BPDU guard is supported on tagged ports as long as it is tagged on both sides to the same VLAN.

Enabling STP BPDU Guard

STP BPDU Guard can be enabled on an individual port or on a set of ports.

When a BPDU guard-enabled port is disabled by BPDU guard, the RUCKUS device will place the port in **errdisable** state and display a message on the console indicating that the port is errdisabled (refer to BPDU Guard Status Example Console Messages on page 225).

NOTE
STP BPDU Guard is disabled by default.

1. Enter the global configuration mode.

```
device# configure terminal
```

2. Enable STP BPDU guard on a specific port or on a set of ports.

```
device(config)# interface ethernet 1/1/1
device(config-if-e1000-1/1/1)# stp-bpdu-guard

device(config)# interface ethernet 1/1/1 to 1/1/9
device(config-mif-1/1/1-1/1/9)# stp-bpdu-guard
```

NOTE
Spanning tree must be enabled on the corresponding VLAN.

3. Re-enable an error-disabled port using one of the following methods:
   - Re-enable an error-disabled port manually.

     ```
     device(config)# interface ethernet 1/1/2
     device(config-if-e1000-1/1/2)# disable
     device(config-if-e1000-1/1/2)# enable
     ```

     To re-enable an error-disabled port manually, first use the **disable** command and then the **enable** command from the interface configuration mode.
   - Re-enable an error-disabled port automatically.

     ```
     device(config)# errdisable recovery cause bpduguard
     ```

     To re-enable a port to recover automatically from the error-disabled state, use the **errdisable recovery cause** command in global configuration mode.
4. Check port status using the **show interface** command.

   ```
   device# show interface ethernet 1/1/2
   Gigabit Ethernet1/1/2 is ERR-DISABLED (bpduguard), line protocol is down
   ```

   The **show interface** command output indicates that the port is in errdisable state.

The following example shows how to enable STP BPDU guard on a specific port.

```
device# configure terminal
device(config)# interface ethernet 1/1/1
device(config-if-e1000-1/1/1)# stp-bpdu-guard
```

The following example shows how to enable STP BPDU guard on a set of ports.

```
device# configure terminal
device(config)# interface ethernet 1/1/1 to 1/1/9
device(config-mif-1/1/1-1/1/9)# stp-bpdu-guard
```

Displaying the BPDU Guard Status

To display the BPDU guard state, enter the **show running configuration** or the **show stp-bpdu-guard** command.

For the BPDU status enter the **stp-bpdu-guard** command.

```
device# show stp-bpdu-guard
BPDU Guard Enabled on:
Ports: (U1/M1)    2    3    5    6   37   38
Ports: (U1/M2)    2
Ports: (U2/M1)    6    7   37   38
Ports: (U3/M1)    9   11   31   33   37   38
Ports: (U3/M2)    2
Ports: (LAG)      1    2
```

BPDU Guard Status Example Configurations

The following example shows how to configure BPDU guard at the interface level and to verify the configuration by issuing the **show stp-bpdu-guard** and the **show interface** commands.

```
device(config)# interface ethernet 1/1/1
device(config-if-e1000-1/1/1)# stp-bpdu-guard
device(config-if-e1000-1/1/1)#
device(config-if-e1000-1/1/1)# show stp-bpdu-guard
BPDU Guard Enabled on:
Ports: (U1/M1)    1
```

```
device(config-if-e1000-1/1/1)#
device(config-if-e1000-1/1/1)# show interfaces ethernet 1/1/1
GigabitEthernet1/1/1 is up, line protocol is up
Port up for 40 seconds
Hardware is GigabitEthernet, address is 0000.00a0.7100 (bia 0000.00a0.7100)
Configured speed auto, actual 100Mbit, configured duplex fdx, actual fdx
Configured mdi mode AUTO, actual MDI
Member of L2 VLAN ID 2, port is untagged, port state is FORWARDING
BPDU guard is Enabled
, ROOT protect is Disabled
STP configured to ON, priority is level0, flow control enabled
mirror disabled, monitor disabled
Not member of any active trunks
Not member of any configured trunks
No port name
Mac-notification is Enabled
IPG MII 96 bits-time, IPG GMII 96 bits-time
IP MTU 1500 bytes
300 second input rate: 8 bits/sec, 0 packets/sec, 0.00% utilization
300 second output rate: 256 bits/sec, 0 packets/sec, 0.00% utilization
88 packets input, 15256 bytes, 0 no buffer
Received 75 broadcasts, 13 multicasts, 0 unicasts
1 input errors, 0 CRC, 0 frame, 0 ignored
0 runts, 0 giants
4799 packets output, 313268 bytes, 0 underruns
Transmitted 90 broadcasts, 4709
```

NOTE

The port up/down time is required only for physical ports and not for loopback/ ve/ tunnel ports.

BPDU Guard Status Example Console Messages

A console message such as the following is generated after a BPDU guard violation occurs on a system that is running MSTP.

```
device(config-if-e1000-1/2/3)# MSTP: Received BPDU on BPDU guard enabled Port 1/2/3,errdisable Port 1/2/3
```

A console message such as the following is generated after a BPDU guard violation occurs on a system that is running STP.

```
device(config)# STP: Received BPDU on BPDU guard enabled Port 1/2/3 (vlan=1), errdisable Port 1/2/3
```

A console message such as the following is generated after a BPDU guard violation occurs on a system that is running RSTP.

```
device(config-vlan-1)# RSTP: Received BPDU on BPDU guard enabled Port 1/2/3 (vlan=1),errdisable Port 1/2/3
```

Root Guard

The standard STP (802.1D), RSTP (802.1W) or 802.1S does not provide any way for a network administrator to securely enforce the topology of a switched Layer 2 network. The forwarding topology of a switched network is calculated on the basis of the root bridge position, along with other parameters. This means any switch can be the root bridge in a network as long as it has the lowest bridge ID. The administrator cannot enforce the position of the root bridge. A better forwarding topology comes with the requirement to place the root bridge at a specific predetermined location. Root guard can be used to predetermine a root bridge location and prevent rogue or unwanted switches from becoming the root bridge.

When root guard is enabled on a port, it keeps the port in a designated role. If the port receives a superior STP Bridge Protocol Data Units (BPDU), it puts the port into a root-inconsistent state and triggers a syslog message and an SNMP trap. The root-inconsistent state is equivalent to the blocking state in 802.1D and to the discarding state in 802.1W. No further traffic is forwarded on this port. This allows the bridge to prevent traffic from being forwarded on ports connected to rogue or misconfigured STP bridges.

Once the port stops receiving superior BPDUs, root guard automatically sets the port back to the learning state, and eventually to a forwarding state through the spanning-tree algorithm.

Configure root guard on all ports where the root bridge should not appear. This establishes a protective network perimeter around the core bridged network, cutting it off from the user network.

NOTE
Root guard may prevent network connectivity if it is improperly configured. Root guard must be configured on the perimeter of the network rather than the core.

NOTE
For the details of MSTP root guard, see "MSTP Root Guard on page 247".

Enabling STP Root Guard

An STP root guard is configured on an interface by entering commands similar to the following.

```
device(config)# interface ethernet 1/1/5
device(config-if-e10000-1/1/5)# spanning-tree root-protect
```

Enter the **no** form of the command to disable STP root guard on the port.

Displaying the STP Root Guard

To display the STP root guard state, enter the **show running configuration** or the **show span root-protect** command.

```
device# show span root-protect
Root Protection Enabled on:
Ports: (U1/M1)   2
```

Displaying the Root Guard by VLAN

You can display root guard information for all VLANs or for a specific VLAN. For example, to display root guard violation information for VLAN 7.

If you do not specify a *vlan-id* , information for all VLANs is displayed.

To display root guard violation information for VLAN 7.

```
device# show spanning-tree vlan 7
STP instance owned by VLAN 7
Global STP (IEEE 802.1D) Parameters:
VLAN Root Root Root Prio Max He- Ho- Fwd Last Chg Bridge
ID ID Cost Port rity Age llo ld dly Chang cnt Address
Hex sec sec sec sec sec
7 a000000011112220 0 Root a000 20 2 1 15 4 4 000011112220
Port STP Parameters:
Port Prio Path State Fwd Design Designated Designated
Num rity Cost Trans Cost Root Bridge
Hex
1 80 19 ROOT-INCONS 2 0 a000000011112220 a000000011112220
```

Designated Protection

Designated Protection ensures that a port cannot go to the designated forwarding state in STP 802.1d or RSTP 802.1w.

You can enable Designated Protection on the port to ensure that it does not go to the designated forwarding state. For example, a fast uplink port should never become a designated port to avoid loops in a network topology. It should either be a root port in any STP state or a non-root port in a blocking state. If STP tries to put this port into the designated forwarding state, the device puts this port into a designated inconsistent STP state. This is effectively equivalent to the listening state in STP in which a port cannot transfer any user traffic. When STP no longer marks this port as a designated port, the port is automatically removed from the designated inconsistent state.

Designation Protection is a port-level feature, while the designated inconsistent state is a per-STP-instance, per-port state. In PVST, a port can belong to several VLANs where each VLAN runs a separate spanning tree instance. The designated inconsistent state in one spanning tree instance does not affect the traffic in other spanning tree instances.

For example, consider an interface eth 1 that is in VLAN 20 and VLAN 50. VLAN 20 runs one instance of STP and VLAN 50 runs another instance. Interface eth1 can be in the designated inconsistent state for VLAN 50 and block the VLAN 50 traffic while it is in root forwarding state for VLAN 20 and allow VLAN 20 traffic.

You can view the status of the Designated Protection feature on a port with the **show interface ethernet** command for that port.

NOTE
You cannot enable Designated Protection and Root Guard on the same port.
Designated Protection does not work with Multiple Spanning Tree Protocol (MSTP) 802.1s.

Enabling Designated Protection on a Port

To disallow the designated forwarding state on a port in STP (802.1d or 802.1w), run the **spanning-tree designated-protect** command in interface configuration mode for that port.

The following example shows that the designated forwarding state is disallowed on Ethernet interface 1/1/1.

```
device(config)# ethernet interface 1/1/1
device(config-if-e1000-1/1/1)# spanning-tree designated-protect
```

Syslog Message for a Port in Designated Inconsistent State

The following syslog message is generated when a port is put in the designated inconsistent state.

```
5d19h00m12s:I:STP: VLAN 100 Designated-protect port 2/1/7, inconsistent, Put into Listening state
```

Error Disable Recovery

If a BPDU Guard violation or loop detection violation occurs, or the number of inError packets exceeds the configured threshold, or if an EFM-OAM enabled interface receives a critical event from the remote device, a port is placed into an error-disabled state, which is functionally equivalent to a disable state. Once in an error-disabled state, the port remains in that state until it is enabled either automatically or manually.

Enabling an Error-Disabled Port Automatically

To enable a port to recover automatically from the error-disabled state after the expiry of a configured error recovery timer, run the **errdisable recovery cause** command in global configuration mode.

For example, to enable error-disable recovery for BPDU guard, enter the following command:

```
device(config)# errdisable recovery cause bpduguard
```

NOTE

When automatic recovery re-enables the port, the port is not in the error-disabled state, but it can remain down for other reasons, such as the Tx/Rx of the fiber optic not being seated properly. Thus, the port is not able to receive the signal from the other side. In this case, after the optic is inserted correctly, you should manually disable the port and then enable it.

Enabling an Error-Disabled Port Manually

To enable an error-disabled port manually, you must first run the **disable** command and then the **enable** command in interface configuration mode to disable the port and then enable the port respectively.

Setting the Recovery Interval

The **errdisable recovery interval** command allows you to configure a timeout for ports in the error-disabled state, after which the ports are re-enabled automatically. To set the error-disabled recovery timeout interval, enter the following command:

```
device(config)# errdisable recovery interval 20
```

Displaying the Error Disable Recovery State by Interface

The port status of errdisabled displays in the output of the **show interface** and the **show interface brief** commands. In this example, errdisable is enabled on interface ethernet 1 and errdisable is enabled because of a BPDU guard violation.

```
device# show interfaces ethernet 1/1/1
GigabitEthernet1/1/1 is ERR-DISABLED (bpduguard),
```

```
line protocol is down
   BPDU guard is Enabled, ROOT protect is Disabled
   Port down for 2 hours 45 minutes 10 seconds
   Hardware is GigabitEthernet, address is 0000.00a0.7100 (bia 0000.00a0.7100)
   Configured speed auto, actual unknown, configured duplex fdx, actual unknown
   Configured mdi mode AUTO, actual unknown
   Member of L2 VLAN ID 2, port is untagged, port state is DISABLED
   STP configured to ON, priority is level0, flow control enabled
   mirror disabled, monitor disabled
   Not member of any active trunks
   Not member of any configured trunks
   No port name
   IPG MII 96 bits-time, IPG GMII 96 bits-time
   IP MTU 1500 bytes
   300 second input rate: 0 bits/sec, 0 packets/sec, 0.00% utilization
   300 second output rate: 0 bits/sec, 0 packets/sec, 0.00% utilization
   145 packets input, 23561 bytes, 0 no buffer
   Received 124 broadcasts, 21 multicasts, 0 unicasts
   1 input errors, 0 CRC, 0 frame, 0 ignored
   0 runts, 0 giants
   5067 packets output, 330420 bytes, 0 underruns
   Transmitted 90 broadcasts, 4977 multicasts, 0 unicasts
   0 output errors, 0 collisions
```

Displaying the Recovery State for All Conditions

Use the **show errdisable recovery** command to display all the default error disable recovery state for all possible conditions. In this example, port 6 is undergoing a recovery.

```
device# show errdisable recovery
ErrDisable Reason Timer Status
--------------------------------------
all reason Disabled
bpduguard Enabled
Timeout Value: 300 seconds
Interface that will be enabled at the next timeout:
Interface Errdisable reason Time left (sec)
-------------- ----------------- ---------------
Port 1/2/3 bpduguard 297
```

Displaying the Recovery State by Port Number and Cause

To see which ports are under an errdisabled state, use the **show errdisable summary** command. This command not only shows the port number, but also displays the reason why the port is in an errdisable state and the method used to recover the port. In this example, port 1/2/6 is errdisabled for a BPDU guard violation.

```
device# show errdisable summary
Port 1/2/6 ERR_DiSABLED for bpduguard
```

Errdisable Syslog Messages

When the system places a port into an errdisabled state for BPDU guard, a log message is generated. When the errdisable recovery timer expires, a log message is also generated.

A Syslog message such as the following is generated after a port is placed into an errdisable state for BPDU guard.

```
STP: VLAN 50 BPDU-guard port 1/6/3 detect (Received BPDU), putting into err-disable state
```

A Syslog message such as the following is generated after the recovery timer expires.

```
ERR_DISABLE: Interface ethernet 1/6/3, err-disable recovery timeout
```

802.1s Multiple Spanning Tree Protocol

Multiple Spanning Tree Protocol (MSTP), as defined in IEEE 802.1s, allows multiple VLANs to be managed by a single STP instance and supports per-VLAN STP. As a result, several VLANs can be mapped to a reduced number of spanning-tree instances. This ensures loop-free topology for one or more VLANs that have the similar layer-2 topology. The RUCKUS implementation supports up to 16 spanning tree instances in an MSTP enabled bridge which means that it can support up to 16 different Layer 2 topologies. The spanning tree algorithm used by MSTP is RSTP which provides quick convergence.

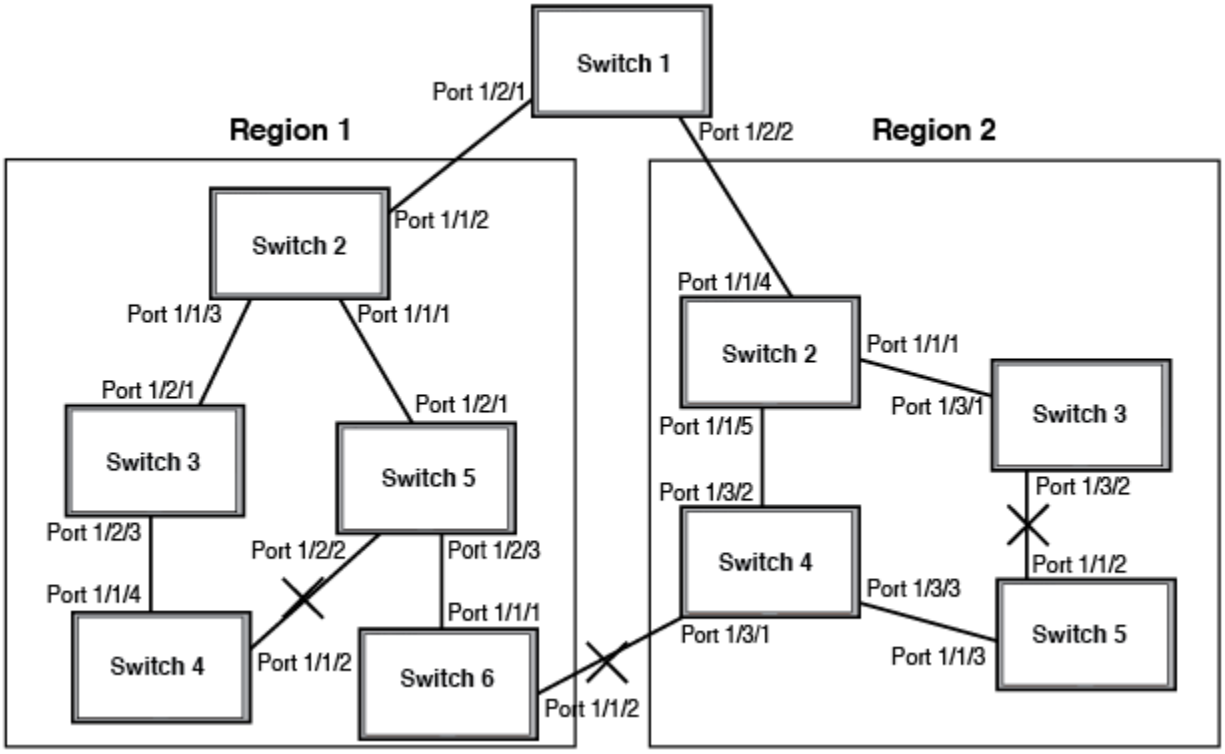
Multiple Spanning Tree Regions

Using MSTP, the entire network runs a common instance of RSTP. Within that common instance, one or more VLANs can be individually configured into distinct regions. The entire network runs the common spanning tree instance (CST) and the regions run a local instance. The local instance is known as Internal Spanning Tree (IST). The CST treats each instance of IST as a single bridge. Consequently, ports are blocked to prevent loops that might occur within an IST and also throughout the CST. With the exception of the provisions for multiple instances, MSTP operates exactly like RSTP.

For example, in Figure 79 a network is configured with two regions: Region 1 and Region 2. The entire network is running an instance of CST. Each of the regions is running an instance of IST. In addition, this network contains Switch 1 running MSTP that is not configured in a region and consequently is running in the CIST instance. In this configuration, the regions are each regarded as a single bridge to the rest of the network, as is Switch 1. The CST prevents loops from occurring across the network. Consequently, a port is blocked at port 1/1/2 of switch 4.

Additionally, loops must be prevented in each of the IST instances. Within the IST Region 1, a port is blocked at port 1/1/2 of switch 4 to prevent a loop in that region. Within Region 2, a port is blocked at port 1/3/2 of switch 3 to prevent a loop in that region.

FIGURE 79 MSTP Configured Network



The following definitions describe the STP instances that define an MSTP configuration.

Common Spanning (CST): CST is defined in 802.1q and assumes one spanning-tree instance for the entire bridged network regardless of the number of VLANs. In MSTP, an MSTP region appears as a virtual bridge that runs CST.

Internal Spanning Tree (IST): IST is a new terminology introduced in 802.1s. An MSTP bridge must handle at least these two instances: one IST and one or more MSTIs (Multiple Spanning Tree Instances). Within each MST region, the MSTP maintains multiple spanning-tree instances. Instance 0 is a special instance known as IST, which extends CST inside the MST region. IST always exists if the switch runs MSTP. Besides IST, this implementation supports up to 15 MSTIs, numbered from 1 to 4094.

Common and Internal Spanning Trees (CIST): CIST is a collection of the ISTs in each MST region and the CST that interconnects the MST regions and single spanning trees.

Multiple Spanning Tree Instance (MSTI): The MSTI is identified by an MST identifier (MSTid) value between 1 and 4094.

MSTP Region: These are clusters of bridges that run multiple instances of the MSTP protocol. Multiple bridges detect that they are in the same region by exchanging their configuration (instance to VLAN mapping), name, and revision-level. Therefore, if you need to have two bridges in the same region, the two bridges must have identical configurations, names, and revision-levels. Also, one or more VLANs can be mapped to one MSTP instance (IST or MSTI) but a VLAN cannot be mapped to multiple MSTP instances.

NOTE
One or more VLANs can be mapped to one MSTP instance (IST or MSTI) but a VLAN cannot be mapped to multiple MSTP instances.

Configuration Notes

When configuring MSTP, note the following:

- With MSTP running, enabling static trunk on ports that are members of many VLANs (4000 or more VLANs) will keep the system busy for 20 to 25 seconds.
- PVST BPDUs are tunneled through 802.1s regions.

Configuring MSTP Mode and Scope

With the introduction of MSTP, a system can be either under MSTP mode or not under MSTP mode. The default state is to not be under MSTP mode. MSTP configuration can only be performed in a system under MSTP mode.

With a system configured under MSTP mode, there is a concept called MSTP scope. MSTP scope defines the VLANs that are under direct MSTP control. You cannot run 802.1D or 802.1w on any VLAN (even outside of MSTP scope) and you cannot create topology groups when a system is under MSTP mode. While a VLAN group will still be supported when a system is under MSTP mode, the member VLAN should either be all in the MSTP scope or all out of the MSTP scope.

When a system is configured from non-MSTP mode to MSTP mode, the following changes are made to the system configuration:

- All 802.1D and 802.1w STP instances are deleted regardless of whether the VLAN is inside the MSTP scope or not.
- All topology groups are deleted.
- Any VSRP configuration is deleted.
- Single-span (if configured) is deleted.
- MRP running on a VLAN inside MSTP scope is deleted.
- The common and internal spanning trees (CIST) collection is created and all VLANS inside the MSTP scope are attached with the CIST.

Make sure that no physical Layer 2 loops exist prior to switching from non-MSTP mode to MSTP mode. If, for example, you have a Layer 2 loop topology configured as a redundancy mechanism before you perform the switch, a Layer 2 storm should be expected.

To configure a system into MSTP mode, use the following command at the Global Configuration level.

```
device(config)# mstp scope all
```

NOTE
When the **mstp scope all** command is issued, the ports associated to VLANs in which any STP is configured transition to the blocking state. You must enter the **mstp start** command to re-converge the ports, VLANs, and VEs.

Once the system is configured into MSTP mode, CIST (sometimes referred to as “instance 0”) is created and all existing VLANs inside the MSTP scope are controlled by CIST. In addition, whenever you create a new VLAN inside MSTP scope, it is put under CIST control by default. In the RUCKUS ICX MSTP implementation however, a VLAN ID can be pre-mapped to another MSTI as described in the “Configuring an MSTP instance” section. A VLAN for which the ID is pre-mapped, will attach to the specified MSTI instead of to the CIST when created.

NOTE
Once under MSTP mode, CIST always controls all ports in the system. If you do not want a port to run MSTP, configure the **no spanning-tree** command under the specified interface configuration.

Configuring **no spanning-tree** command on a system that is configured for MSTP mode changes the system to non-MSTP mode. When this switch is made, all MSTP instances are deleted together with all MSTP configurations. ALL VLANs inside the original MSTP scope will not run any Layer 2 protocols after the switch.

Reduced Occurrences of MSTP Reconvergence

When a VLAN is deleted, the RUCKUS device retains the associated VLAN to MSTI mapping instead of deleting it from the configuration. This way, a VLAN can be pre-mapped to an MSTI and MSTP reconvergence may not be necessary when a VLAN is added to or deleted from the configuration. As long as the VLAN being created or deleted is pre-mapped to an MSTI, and the VLAN to MSTI mapping has not changed, MSTP reconvergence will not occur.

NOTE
MSTP reconvergence occurs when the VLAN to MSTI mapping is changed using the **mstp instance** command.

You can optionally remove VLAN to MSTI mappings from the configuration. Refer to Deleting a VLAN to MSTI Mapping on page 239.

The following shows an example application.

Example Application of MSTP Reconvergence

The following example shows the running configuration file before and after deleting a VLAN from the configuration. The VLAN to MSTI mapping is retained in the running configuration, even after the VLAN is deleted.

```
device(config-vlan-20)# show run

Current configuration:
!
ver 04.2.00bT3e1
!
!
vlan 1 name DEFAULT-VLAN by port
 no spanning-tree
!
vlan 10 by port
 tagged ethe 1/1/1 to 1/1/2
 no spanning tree
!
vlan 20 by port                        <----- VLAN 20 configuration

 tagged ethe 1/1/1 to 1/1/2
 no spanning-tree
!
mstp scope all
mstp instance 0 vlan 1
mstp instance 1 vlan 20
mstp start
some lines omitted for brevity... device(config-vlan-20)#no vlan 20   <----- VLAN 20 deleted
device(config-vlan-20)#show run

Current configuration:
!
ver 04.2.00bT3e1
!
!
vlan 1 name DEFAULT-VLAN by port
 no spanning-tree
!
vlan 10 by port
 tagged ethe 1/1/1 to 1/1/2
 no spanning-tree
!
mstp scope all
mstp instance 0 vlan 1
mstp instance 1 vlan 10
mstp instance 1 vlan 20                <----- VLAN to MSTI mapping kept in
mstp start                                    running configuration, even though
                                              VLAN 20 was deleted
```

```
some lines omitted for brevity...
```

Deleting a VLAN to MSTI Mapping

You can optionally remove a VLAN to MSTI mapping using the **no mstp instance** command. To do so, enter a command such as the following.

```
device(config)# no mstp instance 7 vlan 4 to 7
```

This command deletes the VLAN to MSTI mapping from the running configuration and triggers an MSTP reconvergence.

Viewing the MSTP Configuration Digest

The MSTP Configuration Digest indicates the occurrence of an MSTP reconvergence. The Configuration Digest is recalculated whenever an MSTP reconvergence occurs. To view the Configuration Digest, use the **show mstp config** command. The following shows an example output.

```
device(config-vlan-20)# show mstp config
MSTP CONFIGURATION
------------------
Scope         : all system
Name          :
Revision      : 0
Version       : 3 (MSTP mode)
Config Digest: 0x9bbda9c70d91f633e1e145fbcbf8d321
Status    : Started
Instance VLANs
-------- ----------------------------------------------------
0        1
1        10 20
```

Configuring Multiple Spanning Tree Protocol

Multiple Spanning Tree Protocol can be enabled globally and MSTP bridge and global parameters can be configured.

1. Enter the global configuration mode.

   ```
   device# configure terminal
   ```

2. Enable MSTP scope.

   ```
   device(config)# mstp scope all
   ```

3. Enable MSTP.

   ```
   device(config)# mstp start
   ```

4. Set the MSTP name and revision number.

   ```
   device(config)# mstp name RUCKUS

   device(config)# mstp revision 4
   ```

   Each switch that is running MSTP is configured with a name and revision number. It applies to the switch which can have many different VLANs that can belong to many different MSTP regions.

5. Configure an MSTP instance.

```
device(config)# mstp instance 1 vlan 4 to 7
```

   In the example, an MSTP instance is configured with VLANs 4, 5, 6 and 7 mapped onto it.

   An MSTP instance is configured with an MSTP ID with one or more VLANs. The RUCKUS implementation of MSTP allows you to assign VLANS or ranges of VLANs to an MSTP instance before or after they have been defined.
   The system does not allow a Multiple Spanning Tree Instances (MSTI) without any VLANs mapped to it. Consequently, removing all VLANs from an MSTI, deletes the MSTI from the system.

6. Change bridge priority for the MSTP instance.

```
device(config)# mstp instance 1 priority 0
```

   You can set a **priority** to the instance that gives it forwarding preference over lower priority instances within a VLAN or on the switch. A higher number for the priority variable means a lower forwarding priority.

7. Change the MSTP global parameters using **force-version**, **forward-delay**, **hello-time**, **max-age**, and **max-hop** keywords.

```
device(config)# mstp force-version 0 forward-delay 10 hello-time 4 max-age 12 max-hops 9
```

8. Configure a port as an operational edge port.

```
device(config)# mstp admin-edge-port ethernet 1/1/1
```

   You can define specific ports as edge ports for the region in which they are configured to connect to devices (such as a host) that are not running STP, RSTP, or MSTP.

   You can also configure a Layer 3 switch to automatically set a port as an operational edge port if the port does not receive any BPDUs since link-up. A port can be set as an operational port globally to apply to all the ports on a router where it is configured.

```
device(config)# mstp edge-port-auto-detect
```

9. Set point-to-point link between ports to increase the speed of convergence.

```
device(config)# mstp admin-pt2pt-mac ethernet 1/2/5 ethernet 1/4/5
```

The following example shows how to enable MSTP and configure bridge and MSTP global parameters.

```
device# configure terminal
device(config)# mstp scope all
device(config)# mstp start
device(config)# mstp name RUCKUS
device(config)# mstp revision 4
device(config)# mstp instance 1 vlan 4 to 7
device(config)# mstp instance 1 priority 0
device(config)# mstp force-version 0 forward-delay 10 hello-time 4 max-age 12 max-hops 9
device(config)# mstp admin-edge-port ethernet 1/1/1
device(config)# mstp admin-pt2pt-mac ethernet 1/2/5 ethernet 1/4/5
```

MSTP+ Overview

The MSTP+ feature allows you to selectively include VLANs in the MSTP scope.

In the standard IEEE 802.1s MSTP all VLANs are automatically placed under CIST control so that the entire switch is controlled by the MSTP. The MSTP+ feature is an enhancement that allows you to exclude one or more VLANs from the MSTP scope and configure them in a non-MSTP topology. These VLANs are considered free VLANs and can run any Layer 2 protocols such as PVST/PVRST, MRP, VSRP, or any pure Layer 3 protocols.

You must ensure all the connected devices are properly configured, create the MSTP instances, and assign the VLANs to those instances. These instances must be configured on all devices that interoperate with the same VLAN assignments.

NOTE
In a system running MSTP+, the point-to-point links in the VLAN that wants to run per-vlan STP/RSTP should always be 'tagged'. Only edge-ports can be 'untagged' in that VLAN.

The following table lists the protocols that can run under free VLANs along with the MSTP+.

TABLE 18 Protocols Compatible with MSTP+

Protocols	Compatible with MSTP+
Rapid spanning tree single (IEEE 802.1w)	No
MCT	No
Per-VLAN spanning tree (RSTP)	Yes
VSRP	Yes
MRP	Yes
All Layer 3 protocols (pure Layer 3 network)	Yes

This means that you can create an independent Layer 3 topology even when on a switch that is configured with MSTP. The MSTP convergence does not affect the Layer 3 topology.

You can switch between non-MSTP, MSTP, and MSTP+ modes.

NOTE
Systems configured with MSTP+ may not interoperate properly with the systems on which standard MSTP is configured. It is recommended that you configure MSTP+ on both sides.

NOTE
Free VLANs must have their own means to break Layer 2 loops; MSTP+ cannot be relied on to do so.

Configuring MSTP+

Use the **mstp scope** command with the **pvst** keyword to configure MSTP+.

MSTP+ is not operational until you configure at least one MSTP+ instance and configure the **mstp start** command. You can create MSTP+ instances the same way you configure MSTP instances. See the "Configuring an MSTP instance" section for information on configuring MSTP.

1. Configure MSTP+.

```
Device(config)# mstp scope pvst
Enabling MSTP+ scope. MSTP instances need to be configured and 'mstp start'
need to be entered in order to activate this MSTP+ feature
```

   Configures MSTP+. CIST is not automatically created and VLANs are not under MSTP+ scope unless you explicitly configure the MSTP+ instances and attach the VLANs to them.

2. Create an MSTP+ instance.

```
Device(config)# mstp instance 1 vlan 4 to 7
```

   Creates an MSTP+ instance on VLANs 4 to 7.

3. Start the MSTP+ protocol.

```
Device(config)# mstp start
```

4. Remove the MSTP+ configuration.

```
Device(config)# no mstp scope pvst
```

   Removes the MSTP+ configuration. The VLANs that were attached to MSTP+ are out of MSTP+ scope and there is no PVST under those VLANs. The non-MSTP VLANs are not affected.

Switching Between Non-MSTP, MSTP, and MSTP+ Modes

Use the **mstp scope** command to switch between non-MSTP, MSTP, and MSTP+ modes. This allows you to move between modes without explicitly removing the current mode and reconfiguring the new mode.

When an MSTP instance is enabled, you can configure the **pvst** and **all** keywords to switch between modes.

1. When the **mstp scope all** command is configured and MSTP mode is active, change to MSTP+ mode.

```
Device(config)# mstp scope pvst
```

   The mode is changed to MSTP+. You can remove the VLANs from MSTP+ instances. VLANs that are removed from MSTP+ scope become free and other supported protocols can be configured.

2. When the **mstp scope pvst** command is configured and MSTP+ mode is active, change to MSTP mode.

```
Device(config)# mstp scope all
```

   The mode is changed to MSTP. The VLANs that are already attached to MSTP+ are kept as is and all the free VLANs are attached to a CIST instance. Any protocols configured under the free VLANs are removed.

Disabling MSTP on a Port

To disable MSTP on a specific port, use a command such as the following at the Global Configuration level.

```
device(config)# mstp disable ethernet 2/1/1
```

When a port is disabled for MSTP, it behaves as blocking for all the VLAN traffic that is controlled by MSTIs and the CIST.

Changing MSTP Port Parameters

The path cost for a port comes from the speed of that port if there is no path cost configuration.

If you do not want the link speed to influence the path cost and affect the network topology, then configure the path cost for the port. You can configure a single port, multiple port, or a port range.

1. Enter the following command in global configuration mode level to configure a system into MSTP mode.

```
device(config)# mstp scope all
```

2. Enter a command such as the following to configure the path cost configuration and the priority together.

```
device(config)# mstp instance 0 ethernet 1/1/4 ethernet 1/1/5 ethernet 1/1/15 to 1/1/17 path-cost
20000 priority 192
```

   You can configure path cost and priority separately.

3. Enter the following command to enable MSTP in global configuration mode level.

```
device(config)# mstp start
```

4. Verify the running configuration.

```
device(config)# show run | include mstp

mstp scope all
mstp instance 0 vlan 1
mstp instance 0 ethe 1/1/4 to 1/1/5 ethe 1/1/15 to 1/1/17 path-cost 20000
mstp instance 0 ethe 1/1/4 to 1/1/5 ethe 1/1/15 to 1/1/17 priority 64
mstp start
```

Forcing Ports to Transmit an MSTP BPDU

To force a port to transmit an MSTP BPDU, use a command such as the following at the Global configuration mode.

```
device(config)# mstp force-migration-check ethernet 3/1/1
```

Enabling MSTP on a Device

You must enable MSTP on the device.

MSTP scope must be enabled on the switch as described in Configuring MSTP Mode and Scope on page 237 before MSTP can be enabled.

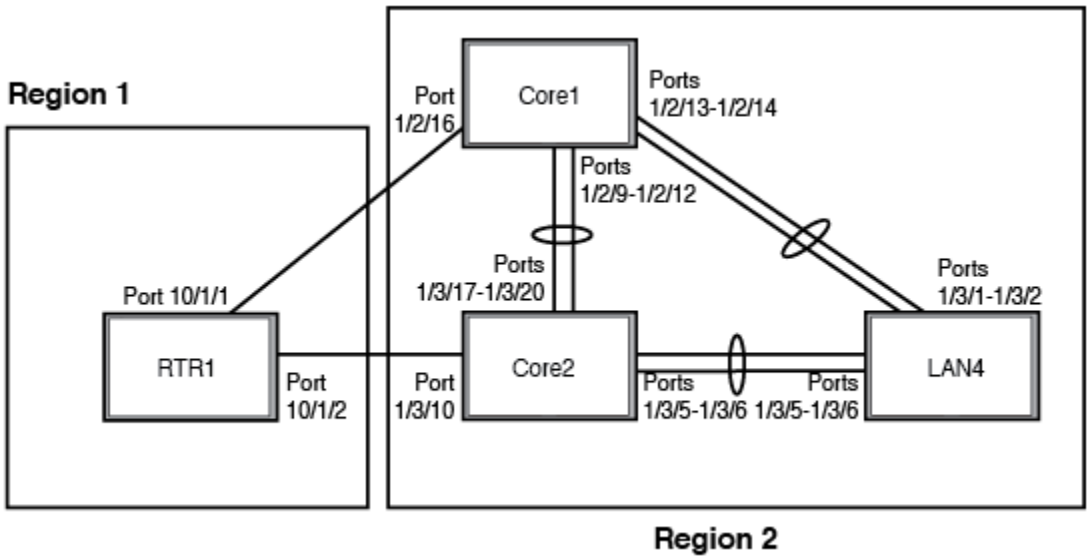
To enable MSTP on your switch, use the following at the Global Configuration level.

```
device(config)# mstp start
```

Examples of an MSTP configuration

In the following figure, four RUCKUS device routers are configured in two regions. There are four VLANs in four instances in Region 2. Region 1 is in the CIST.

FIGURE 80 Sample MSTP Configuration



RTR1 on MSTP Configuration

```
device(config-vlan-4093)# tagged ethernet 10/1/1 to 10/1/2
device(config-vlan-4093)# exit
device(config)# mstp scope all
device(config)# mstp name Reg1
device(config)# mstp revision 1
device(config)# mstp admin-pt2pt-mac ethernet 10/1/1 to 10/1/2
device(config)# mstp start
device(config)# hostname RTR1
```

Core 1 on MSTP Configuration

```
device(config)# trunk ethernet 1/2/9 to 1/2/12 ethernet 1/2/13 to 1/2/14
device(config-vlan-1)# name DEFAULT-VLAN by port
device(config-vlan-1)# exit
device(config)# vlan 20 by port
device(config-vlan-20)# tagged ethernet 1/2/9 to 1/2/14 ethernet 1/2/16
device(config-vlan-20)# exit
device(config)# vlan 21 by port
device(config-vlan-21)# tagged ethernet 1/2/9 to 1/2/14 ethernet 1/2/16
device(config-vlan-21)# exit
device(config)# vlan 22 by port
device(config-vlan-22)# tagged ethernet 1/2/9 to 1/2/14 ethernet 1/2/16
device(config-vlan-22)# exit
device(config)# vlan 23 by port
device(config)# mstp scope all
device(config)# mstp name HR
device(config)# mstp revision 2
device(config)# mstp instance 20 vlan 20
device(config)# mstp instance 21 vlan 21
device(config)# mstp instance 22 vlan 22
device(config)# mstp instance 0 priority 8192
device(config)# mstp admin-pt2pt-mac ethernet 1/2/9 to 1/2/14
device(config)# mstp admin-pt2pt-mac ethernet 1/2/16
device(config)# mstp disable ethernet 2/240.
device(config)# mstp start
device(config)# hostname CORE1
```

Core2 on MSTP Configuration

```
device(config)# trunk ethernet 1/3/5 to 1/3/6 ethernet 1/3/17 to 1/3/20
device(config)# vlan 1 name DEFAULT-VLAN by port
device(config-vlan-1)# exit
device(config)# vlan 20 by port
device(config-vlan-20)# tagged ethernet 1/3/5 to 1/3/6 ethernet 1/3/17 to 1/3/20
device(config-vlan-20)# exit
device(config)# vlan 21 by port
device(config-vlan-21)# tagged ethernet 1/3/5 to 1/3/6 ethernet 1/3/17 to 1/3/20
device(config-vlan-21)# exit
device(config)# vlan 22 by port
device(config-vlan-22)# tagged ethernet 1/3/5 to 1/3/6 ethernet 1/3/17 to 1/3/20
device(config-vlan-22)# exit
device(config)# mstp scope all
device(config)# mstp name HR
device(config)# mstp revision 2
device(config)# mstp instance 20 vlan 20
device(config)# mstp instance 21 vlan 21
device(config)# mstp instance 22 vlan 22
device(config)# mstp admin-pt2pt-mac ethernet 1/3/17 to 1/3/20 ethernet 1/3/5 to 1/3/6
device(config)# mstp admin-pt2pt-mac ethernet 1/3/10
device(config)# mstp disable ethernet 1/3/7 ethernet 1/3/24
device(config)# mstp start
device(config)# hostname CORE2
```

LAN 4 on MSTP Configuration

```
device(config)# trunk ethernet 1/3/5 to 1/3/6 ethernet 3/1/1 to 3/1/2
device(config)# vlan 1 name DEFAULT-VLAN by port
device(config-vlan-1)# exit
device(config)# vlan 20 by port
device(config-vlan-20)# tagged ethernet 3/1/1 to 3/1/2 ethernet 3/1/5 to 3/1/6
device(config)# exit
device(config)# vlan 21 by port
device(config-vlan-21)# tagged ethernet 3/1/1 to 3/1/2 ethernet 3/1/5 to 3/1/6
device(config-vlan-21)# exit
device(config)# vlan 22 by port
device(config-vlan-22)# tagged ethernet 3/1/1 to 3/1/2 ethernet 3/1/5 to 3/1/6
device(config)# mstp scope all
device(config)# mstp config name HR
device(config)# mstp revision 2
device(config)# mstp instance 20 vlan 20
device(config)# mstp instance 21 vlan 21
device(config)# mstp instance 22 vlan 22
device(config)# mstp admin-pt2pt-mac ethernet 3/1/5 to 3/1/6 ethernet 3/1/1 to 3/1/2
device(config)# mstp start
device(config)# hostname LAN4
```

Displaying MSTP Statistics

MSTP statistics can be displayed using the commands shown below.

To display all general MSTP information, enter the following command.

```
device# show mstp
MSTP Instance 0 (CIST) - VLANs: 1
-------------------------------------------------------------------------------
Bridge           Bridge Bridge Bridge Bridge Root   Root  Root   Root
Identifier       MaxAge Hello  FwdDly Hop    MaxAge Hello FwdDly Hop
hex              sec    sec    sec    cnt    sec    sec   sec    cnt
8000000cdb80af01 20     2      15     20     20     2     15     19
Root             ExtPath   RegionalRoot     IntPath   Designated       Root
Bridge           Cost      Bridge           Cost      Bridge           Port
hex                        hex                        hex
```

```
8000000480bb9876 2000      8000000cdb80af01 0         8000000480bb9876 3/1/1
Port  Pri PortPath  P2P Edge Role       State      Designa-  Designated
Num       Cost      Mac Port                      ted cost  bridge
3/1/1   128 2000      T   F    ROOT       FORWARDING 0         8000000480bb9876
MSTP Instance 1 - VLANs: 2
------------------------------------------------------------------------------
Bridge           Max RegionalRoot     IntPath   Designated       Root  Root
Identifier       Hop Bridge           Cost      Bridge           Port  Hop
hex              cnt hex                        hex                    cnt
8001000cdb80af01 20  8001000cdb80af01 0         8001000cdb80af01 Root  20
Port  Pri PortPath  Role       State      Designa-  Designated
Num       Cost                          ted cost  bridge
3/1/1   128 2000      MASTER     FORWARDING 0         8001000cdb80af01
```

Displaying MSTP Information for a Specified Instance

The following example displays MSTP information specified for an MSTP instance.

```
device# show mstp 1
MSTP Instance 1 - VLANs: 2
------------------------------------------------------------------------------
Bridge           Max RegionalRoot     IntPath   Designated       Root  Root
Identifier       Hop Bridge           Cost      Bridge           Port  Hop
hex              cnt hex                        hex                    cnt
8001000cdb80af01 20  8001000cdb80af01 0         8001000cdb80af01 Root  20
Port    Pri    PortPath  Role       State       Designa-    Designated
Num            Cost                           ted cost    bridge
1/3/1   128    2000      MASTER     FORWARDING  0           8001000cdb80af01
```

Displaying MSTP Information for CIST Instance 0

Instance 0 is the Common and Internal Spanning Tree (CIST) Instance. When you display information for this instance, there are some differences from the information displayed for other instances.. The following example displays MSTP information for CIST instance 0.

```
device# show mstp 0
MSTP Instance 0 (CIST) - VLANs: 1
------------------------------------------------------------------------------
Bridge           Bridge Bridge Bridge Bridge Root   Root  Root   Root
Identifier       MaxAge Hello  FwdDly Hop    MaxAge Hello FwdDly Hop
hex              sec    sec    sec    cnt    sec    sec   sec    cnt
8000000cdb80af01 20     2      15     20     20     2     15     19
Root             ExtPath   RegionalRoot     IntPath   Designated       Root
Bridge           Cost      Bridge           Cost      Bridge           Port
hex                        hex                        hex
8000000480bb9876 2000      8000000cdb80af01 0         8000000480bb9876 3/1/1
Port    Pri PortPath  P2P Edge Role       State      Designa-  Designated
Num         Cost      Mac Port                      ted cost  bridge
3/1/1   128 2000      T   F    ROOT       FORWARDING 0         8000000480bb9876
```

To display details about the MSTP configuration, enter the **show mstp configuration** command.

```
device# show mstp configuration
MSTP CONFIGURATION
------------------
Name     : Reg1
Revision : 1
Version  : 3 (MSTP mode)
Status   : Started
Instance VLANs
-------- ----------------------------------------------------
0        4093
```

To display details about the MSTP configuration on the device, enter the **show mstp detail** command.

```
device# show mstp detail
MSTP Instance 0 (CIST) - VLANs:1
--------------------------------------------------------------------------------
  Bridge: 800000a0c9c002a0 [Priority 32768, SysId 0, Mac 00a0c9c002a0]
  FwdDelay 15, HelloTime 2, MaxHops 20, TxHoldCount 6, ForceVersion 3
Number of topology changes: 20
Last topology change occured 6 second(s)  ago on lg40

Port 1/1/1 - Role: DESIGNATED - State: FORWARDING
  PathCost 20000, Priority 128, OperEdge F, OperPt2PtMac F, rcvdInternal F, Boundary F
  Designated   - Root 800000a0c9c002a0, RegionalRoot 800000a0c9c002a0,
                 Bridge 800000a0c9c002a0, ExtCost 0, IntCost 0
  ActiveTimers - helloWhen 2
  MachineState - PRX-DISCARD, PTX-IDLE, PPM-SENDING_RSTP, PIM-CURRENT
                 PRT-ACTIVE_PORT, PST-FORWARDING, TCM-ACTIVE
  BPDUs      - Rcvd MST 0, RST 0, Config 0, TCN 0
               Sent MST 129, RST 0, Config 0, TCN 0
```

MSTP Root Guard

Root guard is supported for MSTP at the port level, to ensure that the port is in the designated state all the time. The timeout value for root protection is user-configurable.

> **NOTE**
> For details of root guard, refer to Root Guard on page 227.

The MSTP root guard feature ensures that the port on which root guard is enabled is the designated port. A MSTP BPDU normally contains multiple instances of information, including CIST and any MSTIs of which the sending port is member. If an MSTP BPDU is received on a root-guard-enabled port with either CIST or any MSTI information considered to be "superior," the switch puts that port in a "root inconsistent" STP state in that CIST or MSTI instance, which is effectively equal to a discarding state in 802.1S, to make sure that no traffic is forwarded across this port in that CIST or MSTI instance.

For example, root-guard-configured port 1/1/5 belongs to CIST, MSTI 1, 3, 5, 6. If the MSTP BPDU received by port 1/1/5 has superior information for CIST and MSTI 5, 6 but inferior information for MSTI 1, 3, the port 1/1/5 is put into "root inconsistent" state in CIST and MSTI 5, 6.

The recovery from the root-inconsistent state is made automatic through the MSTP root guard timer, which is a per-port per MSTP instance timer. The timeout value for this timer can be configured globally. If the configured MSTP root guard timeout is 60 seconds, any superior information received on the port for an MSTP instance ensures that the port stays in the root-inconsistent state and resets the timer back to 60 seconds. If no superior information is received on that port during that interval, the port is put into the root-consistent state (the normal state) for that MSTP instance. This triggers the entire port to re-initialize (the port is re-initialize in all MSTP instances of which that port is member, including CIST and any MSTIs of which the port is a member).

Regarding previous root guard support for 802.1D/802.1W, when root guard is configured on a root port/alternate/backup port, the port is put into the designated blocking state immediately. The MSTP root guard operates differently. It depends on the next superior BPDU to bring the port into the root-inconsistent state. Until that time, the port may still be in the alternative/root/backup role.

When the system moves a port into or out of the root-inconsistent state, a syslog message is generated as in the following example. The log message format is the same for both 802.1D and 802.1W.

```
 0d00h14m50s:I:MSTP: Root-protect port 3/1/5, MSTP Index 16 (CIST) inconsistent (Received superior BPDU)

0d00h14m35s:I:MSTP: Root-protect port 3/1/5, MSTP index 16 (CIST) consistent (Timeout)
```

NOTE
This feature works on the switch that locally enabled MSTP. If only have 802.1D/802.1W is enabled on the device, even sending MSTP BPDUs with bridge ID 0 does not trigger anything.

Do the following to enable MSTP root guard.

- Use the **spanning-tree root-protect** command to enable MSTP root guard on an interface. This command is used to set the port on root guard for all spanning tree protocols.
- Use the global **mstp root-protect timeout** command to configure root protection timeout value for MSTP root guard.
- Use the **show mstp root-protect** command to verify the configuration.

Configuring MSTP Root Guard

MSTP root guard can be enabled on an individual port of an interface.

1. Enter the global configuration mode.

```
device# configure terminal
```

2. Enable MSTP root guard on a port by using the **spanning-tree root-protect** command at the interface configuration level.

```
device(config)# interface ethernet 1/1/1
device(config-if-e10000-1/1/1)# spanning-tree root-protect
```

3. Exit the interface configuration mode and enter the global configuration mode.

```
device(config-if-e10000-1/1/1)# exit

device# configure terminal
```

4. Setting MSTP root guard timeout value by using the **mstp root-protect timeout** command.

```
device(config)# mstp root-protect timeout 60
```

The following example shows how to configure MSTP root guard.

```
device# configure terminal
device(config)# interface ethernet 1/1/1
device(config-if-e10000-1/1/1)# spanning-tree root-protect
device(config-if-e10000-1/1/1)# exit
device# configure terminal
device(config)# mstp root-protect timeout 60
```

Displaying MSTP Root Guard Configuration

You can display the MSTP root guard configuration information.

Use the following **show** commands from privileged EXEC mode to display MSTP information.

The following example displays the MSTP root guard configuration.

```
device# show mstp

MSTP Instance 0 (CIST) - VLANs:
1
--------------------------------------------------------------------------------
Bridge Bridge Bridge Bridge Bridge Root Root Root Root
Identifier MaxAge Hello FwdDly Hop MaxAge Hello FwdDly Hop
hex sec sec sec cnt sec sec sec cnt
2000000000000000a 20 2 15 20 20 2 15 20
```

```
Root ExtPath RegionalRoot IntPath Designated Root
Bridge Cost Bridge Cost Bridge Port
hex hex hex
200000000000000a 0 200000000000000a 0 200000000000000a Root
Port Pri Port PortPath P2P Edge Role State Designa- Designated
Num Id Cost Mac Port ted cost bridge
1/1/5 128 193 1400 F F DESIGNATE RT-INCO 0 200000000000000a
MSTP Instance 1 - VLANs:
20
------------------------------------------------------------------------------
Bridge Max RegionalRoot IntPath Designated Root Root
Identifier Hop Bridge Cost Bridge Port Hop
hex cnt hex hex cnt
200100000000000a 20 200100000000000a 0 200100000000000a Root 20
Port Pri Port PortPath Role State Designa- Designated
Num Id Cost ted cost bridge
1/1/5 128 193 1400 DESIGNATE FORWARD 0 200100000000000a
```

A root-inconsistent port is indicated by RTINCO.

The following example verifies whether MSTP instances are in consistent or inconsistent state by using the **show mstp root-protect** command.

```
device# show mstp root-protect

Port MSTI Current State
1/1/5 MSTI 1 Consistent state
1/1/5 CIST Inconsistent state (59 seconds left on timer)
```

Packet InError Detection

Packet InError Detection identifies links that receive a higher number of bad frames than the configured threshold and disables them to avoid instability in the network. For instance, if a network has redundant uplinks, usually only one link is in forwarding state and the rest are redundant and blocked. If one of the redundant links becomes faulty, it may drop the PDUs and become a forwarding link. This can cause loops in the network. Packet InError Detection detects the faults in the link and disables the link to prevent loops in the network.

Packet InError Detection counts an ingress frame that has one or more of the following errors as an inError packet:

- Alignment error
- CRC error
- Oversized frame error
- Internal received MAC address error (Errors that do not fall in the above 3 types)
- Symbol error (includes the fragmented, short, or undersized frames)

You can configure the number of inError packets allowed per port in a specified sampling interval. If the port receives more than the configured number of inError packets in two consecutive sampling intervals, then the port becomes error-disabled. The output of the **show interface ethernet** command for the affected port will show the status of the port as "ERR-DISABLED (packet-inerror)".

NOTE
It is recommended to use Packet InError Detection only on required ports. If you enable this on a large number of ports in a device and use a very short sampling interval, it may lead to heavy CPU usage.

NOTE
The inError count configured on the LAG virtual interface of a LAG is inherited by other member ports of the LAG. However, the LAG ports are individually sampled for inError packets. Therefore, inError packets on a port disable only that port and not the entire LAG.

NOTE
Executing commands that clear the packet counters, such as the **clear statistics** command may interfere with the proper functioning of Packet InError Detection because these commands reset the inError packet count.

Configuring Packet InError Detection

Perform the following steps to configure Packet InError Detection:

1. Run the **errdisable packet-inerror-detect** command in global configuration mode to enable the feature and to define the sampling time interval.
2. Run the **packet-inerror-detect** command in interface configuration mode of the port that you want to monitor for inError packets.
3. *(Optional)* If you want the ports to automatically recover from the error-disabled state after the expiry of a configured recovery timer, run the **errdisable recovery cause** and **errdisable recovery interval** commands in global configuration mode. For more details, see "Enabling an error-disabled port automatically" on page 15 on page 231 and Setting the Recovery Interval on page 231.

The following example shows the configuration of Packet InError Detection on a device and its Ethernet interface 1/1/1.

```
device(config)# errdisable packet-inerror-detect interval 3
device(config)# errdisable recovery cause packet-inerror-detect
device(config)# errdisable recovery interval 20
device(config)# interface ethernet 1/1/1
device(config-if-e1000-1/1/1)# packet-inerror-detect 10
```

The ethernet interface 1/1/1 becomes disabled if more than 10 inError packets are received in each of two consecutive 3-second intervals. After the interface is disabled, it automatically recovers to the enabled state after 20 seconds.

Syslog Message for Error-Disabled Port Due to inError Packets

The following syslog message is generated when a port is error-disabled because of inError packets.

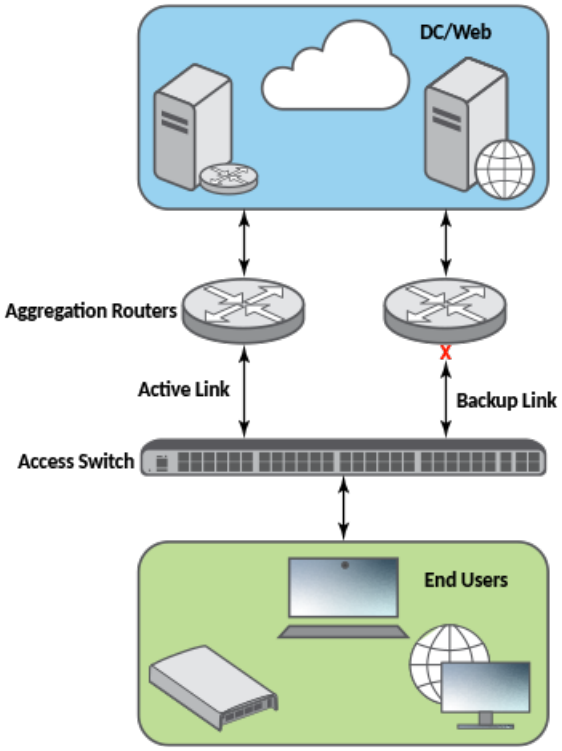
```
0d01h38m44s:I:PORT: 1/1/37 is ERR-DISABLED due to number of packet inErrors exceeded the threshold
```

Flexlink

Flexlink Overview

Flexlink is a link redundancy feature that provides Layer 2 resilience. Flexlink provides link redundancy at a faster network convergence rate than Spanning Tree Protocol (STP) and its variants (RSTP and MSTP). Generally, it is deployed at the access layer where the switch has dual uplinks to the distribution layer. Flexlink re-aligns the forwarding path on the local switch where it is configured and has no role in adjusting the forwarding path on other switches in the network.

FIGURE 81 Flexlink Deployment



Flexlink is enabled on a pair of interfaces by configuring an interface as active link and other interface as backup link. The interfaces can be physical interfaces or LAG interfaces. When Flexlink instance is configured, spanning support is disabled on the interfaces configured as flexlinks and the no span configuration will show up as explicit interface configuration under those interfaces.

The interface under which Flexlink is enabled is automatically considered the active link. The active link forwards the network traffic and the backup link remains dormant. When the active link goes down, the backup link transitions to the active link state and begins forwarding the traffic. When the originally configured active link comes back up, it goes into backup mode and does not forward traffic. This default failover forwarding behavior can be changed by configuring a preemption mechanism, which allows you to specify the preferred port for forwarding traffic.

In a Flexlink configuration, traffic flows through the active link and all dynamic MAC entries are learned through the active link. Upon link failure, the MAC addresses learned on the active link are moved to the backup link for network packet forwarding because the backup link is set to the

Forwarding state. At any given point of time, only the active link forwards the traffic. The Flexlink interfaces (both the active link and backup link) must be physically up to ensure expected failover forwarding behavior. No configuration is required on the peer switch or backup interface for Flexlink to work.

Preemption

Flexlink preemption is a mechanism that allows you to specify the preferred interface for forwarding traffic. That is, you can define the criteria for preempting the current active interface by the other interface when it comes back up after a failover. This is done by configuring the following preemption modes:

By default, the preemption mode is "off", that is no preemption will take place. When active link fails, backup will take over and forward the traffic.

- Forced: The preemption forced mode always provides precedence to the configured active interface to forward the traffic. When the link status of the configured active interface comes up after the failover, it preempts the current active link (configured backup interface) and takes over traffic forwarding.
- Bandwidth: The preemption bandwidth mode preempts the Flexlink link that has lesser bandwidth and chooses the interface with higher bandwidth as the active link.

Preemption Delay

A preemption delay can be set to postpone the preemption of the current active link by the backup link according to the configured preemption mode. That is, you can set a time delay (in seconds) before allowing the backup link to preempt the current active interface and take over traffic again. The preemption delay helps to avoid frequent active-to-backup switchover. Preemption delay configuration is applicable only for **forced** and **bandwidth** preemption modes. Preemption delay configuration and the preemption mode **off** are mutually exclusive.

Flexlink Configuration Notes and Considerations

- A maximum of 32 Flexlink instances can be configured.
- Flexlink is supported on LAG interfaces.
- Flexlink is supported in a stacking configuration.

  Flexlink interfaces must be part of the same VLAN domain. The addition and deletion of Flexlink interfaces on the VLAN must be organized as a port pair.
- The forwarding state of the Flexlink interfaces are controlled by Flexlink feature, so other Layer2 features cannot co-exist on those interfaces. Following features cannot be enabled on Flexlink interfaces:
  - Spanning Tree protocol (STP)
  - Per-VLAN protocols (Metro Ring Protocol [MRP] and Virtual Switch Redundancy Protocol [VSRP])
  - Multiple VLAN Registration Protocol (MVRP)
  - Private VLAN (PVLAN)
  - Port MAC security (PMS)
  - Protected ports
  - Flexible Authentication
  - Loop detection
- ACL configurations can be applied, but ingress or egress traffic on the backup link will not take effect.

Configuring Flexlink

Complete the following steps to configure a Flexlink instance.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Specify the interface that you want to configure as the active link of the Flexlink pair and enter the interface configuration mode.

   The interface can be a physical interface or LAG interface.

```
device(config)# interface ethernet 1/1/9
```

3. Configure a Flexlink instance by specifying the backup interface (physical interface or LAG interface) to form a pair of Flexlink interfaces.

   The interface under which the **flexlink backup** command is run is automatically considered as the active link.

```
device(config-if-e1000-1/1/9)# flexlink backup interface ethernet 1/1/10
```

4. Configure the preemption scheme to specify the preferred interface for forwarding traffic by configuring one of the following modes:

   By default, no preemption will take place. When the active link fails, backup will take over and forward the traffic.

   - Configure the preemption mechanism to provide precedence to the configured active link and always preempt the backup link.

```
device(config-if-e1000-1/1/9)# flexlink preemption mode forced
```

   - Configure the preemption mechanism to preempt the Flexlink interface with lesser bandwidth and choose the interface with higher bandwidth as the active link.

```
device(config-if-e1000-1/1/9)# flexlink preemption mode bandwidth
```

5. Configure the time delay in seconds, so that link is preempted after that delay. Otherwise, link will get preempted immediately. Default value for the delay is 0.

```
device(config-if-e1000-1/1/9)# flexlink preemption delay 100
```

The following example configures a Flexlink instance with Ethernet interfaces as the Flexlink pair ports.

```
device# configure terminal
device(config)# interface ethernet 1/1/9
device(config-if-e1000-1/1/9)# flexlink backup interface ethernet 1/1/10
device(config-if-e1000-1/1/9)# flexlink preemption mode forced
device(config-if-e1000-1/1/9)# flexlink preemption delay 100
```

The following example configures a Flexlink instance with LAG interfaces as the Flexlink pair ports.

```
device# configure terminal
device(config)# interface lag 10
device(config-lag-if-lg10)# flexlink backup interface lag 11
device(config-lag-if-lg10)# flexlink preemption mode bandwidth
device(config-lag-if-lg10)# flexlink preemption delay 100
```

Displaying Flexlink Information

Use the **show flexlink all** command to display brief information about all Flexlink instances configured on the device.

```
device(config)# show flexlink all
    Active Interface        Backup Interface        State
    ------------------------------------------------------------
    ethernet 1/1/9            ethernet 1/1/10       Active Up/Backup FL-Backup

    lag 10                    lag 11                Active Up/Backup FL-Backup
```

Use the **show flexlink all detail** command to display detailed information about all Flexlink instances configured on the device.

```
device(config)# show flexlink all detail
    Active Interface        Backup Interface        State
    ------------------------------------------------------------
    ethernet 1/1/9         ethernet 1/1/10          Active Up/Backup FL-Backup
    Preemption Mode: forced
    Preemption delay: 100
    Bandwidth : 1/1/9 (1Gbit), 1/1/10 (1Gbit)

    lag 10              lag 11                      Active Up/Backup FL-Backup
    Preemption Mode: Bandwidth
    Preemption delay: 100
    Bandwidth : lg10 (20G), lg11 (10G)
```

Use the **show flexlink interface** command to display brief information about a specific Flexlink interface (physical interface).

```
device(config)# show flexlink interface ethernet 1/1/9
    Active Interface        Backup Interface        State
    ------------------------------------------------------------
    ethernet 1/1/9         ethernet 1/1/10          Active Up/Backup FL-Backup
```

Use the **show flexlink interface** command with the **detail** keyword to display detailed information about a specific Flexlink interface (physical interface).

```
device(config)# show flexlink interface ethernet 1/1/9 detail
    Active Interface        Backup Interface        State
    ------------------------------------------------------------
    ethernet 1/1/9         ethernet 1/1/10             Active Up/Backup FL-Backup
    Preemption Mode: forced
    Preemption delay: 100
    Bandwidth : 1/1/9 (1Gbit), 1/1/10 (1Gbit)
```

Use the **show flexlink interface** command to display brief information about a specific Flexlink interface (LAG interface).

```
device(config)# show flexlink interface lag 10
    Active Interface        Backup Interface        State
    ------------------------------------------------------------
    lag 10                  lag 11                Active Up/Backup FL-Backup
```

Use the **show flexlink interface** command with the **detail** keyword to display detailed information about a specific Flexlink interface (LAG interface).

```
device(config)# show flexlink interface lag 10 detail
    Active Interface        Backup Interface        State
    ------------------------------------------------------------
    lag 10              lag 11                Active Up/Backup FL-Backup
    Preemption Mode: bandwidth
    Preemption delay: 100
    Bandwidth : lg10 (20G), lg11 (10G)
```

The following example displays the Flexlink running configuration under the active link interface. The backup link does not generate any configuration.

```
device(config)# show running-config interface ethernet 1/1/9 ethernet 1/1/10
     interface ethernet 1/1/9
         no spanning-tree
       flexlink backup interface ethernet 1/1/10
          flexlink preemption mode forced
          flexlink preemption delay 100

        interface ethernet 1/1/10
         no spanning-tree
         !
```

The following example displays the Flexlink status and the active link or backup link status under a specific interface.

```
device(config)# show interface ethernet 1/1/9 ethernet 1/1/10
    GigabitEthernet1/1/9 is up, line protocol is up
```

```
      Port up for 21 minute(s) 37 second(s)
      Hardware is GigabitEthernet, address is 609c.9f08.7c90 (bia 609c.9f08.7c90)
      Configured speed auto, actual 1Gbit, configured duplex fdx, actual fdx
      Configured mdi mode AUTO, actual MDIX
      EEE Feature Disabled
      Untagged member of L2 VLAN 1, port state is FORWARDING
      BPDU guard is Disabled, ROOT protect is Disabled, Designated protect is Disabled
      Link Error Dampening is Disabled
      STP configured to OFF, priority is level0, mac-learning is enabled
      MACsec is Disabled
      Openflow is Disabled, Openflow Hybrid mode is Disabled, Flow Control is config enabled, oper enabled,
negotiation disabled
      Mirror disabled, Monitor disabled
      Mac-notification is disabled
      VLAN-Mapping is disabled
      Not member of any active trunks
      Not member of any configured trunks
      No port name
      IPG MII 96 bits-time, IPG GMII 96 bits-time
      MTU 1500 bytes, encapsulation ethernet
      MMU Mode is Store-and-forward
      427423738 packets input, 147888438354 bytes, 0 no buffer
      Received 427423113 broadcasts, 623 multicasts, 0 unicasts
      0 input errors, 0 CRC, 0 frame, 0 ignored
      0 runts, 0 giants
      426927885 packets output, 147717044454 bytes, 0 underruns
      Transmitted 426927868 broadcasts, 17 multicasts, 0 unicasts
      0 output errors, 0 collisions
      Relay Agent Information option: Disabled
      Protected: No
      MAC Port Security: Disabled
      Flexlink: Enabled (Active link)

    UC Egress queues:
    Queue counters    Queued packets    Dropped Packets
             0                 0                 0
             1                 0                 0
             2                 0                 0
             3                 0                 0
             4                 0                 0
             5                 0                 0
             6                 0                 0
             7                43                 0


    MC Egress queues:
    Queue counters    Queued packets    Dropped Packets
             0            426961097                 0
             1                 0                 0
             2                 0                 0
             3                 0                 0

device(config)# show interface ethernet 1/1/10
    GigabitEthernet1/1/10 is up, line protocol is down (FL-BACKUP)
      Port down (FL-BACKUP) for 21 minute(s) 47 second(s)
      Hardware is GigabitEthernet, address is 609c.9f08.7c91 (bia 609c.9f08.7c91)
      Configured speed auto, actual 1Gbit, configured duplex fdx, actual fdx
      Configured mdi mode AUTO, actual MDIX
      EEE Feature Disabled
      Untagged member of L2 VLAN 1, port state is BLOCKING
      BPDU guard is Disabled, ROOT protect is Disabled, Designated protect is Disabled
      Link Error Dampening is Disabled
      STP configured to OFF, priority is level0, mac-learning is enabled
      MACsec is Disabled
      Openflow is Disabled, Openflow Hybrid mode is Disabled, Flow Control is config enabled, oper enabled,
negotiation disabled
      Mirror disabled, Monitor disabled
      Mac-notification is disabled
      VLAN-Mapping is disabled
      Not member of any active trunks
      Not member of any configured trunks
      No port name
```

```
  IPG MII 96 bits-time, IPG GMII 96 bits-time
  MTU 1500 bytes, encapsulation ethernet
  MMU Mode is Store-and-forward
  418345415 packets input, 144747337532 bytes, 0 no buffer
  Received 418344785 broadcasts, 628 multicasts, 0 unicasts
  0 input errors, 0 CRC, 0 frame, 0 ignored
  0 runts, 0 giants
  77723 packets output, 26887364 bytes, 0 underruns
  Transmitted 77706 broadcasts, 17 multicasts, 0 unicasts
  0 output errors, 0 collisions
  Relay Agent Information option: Disabled
  Protected: No
  MAC Port Security: Disabled
  Flexlink: Enabled (Backup link)

UC Egress queues:
Queue counters    Queued packets    Dropped Packets
         0                  0                  0
         1                  0                  0
         2                  0                  0
         3                  0                  0
         4                  0                  0
         5                  0                  0
         6                  0                  0
         7                 42                  0


MC Egress queues:
Queue counters    Queued packets    Dropped Packets
         0              77681                  0
         1                  0                  0
         2                  0                  0
         3                  0                  0
```

Unknown Unicast Flood Block

Unknown Unicast Flood Block Overview

Unknown unicast traffic consists of unicast packets without known destinations. The unknown unicast traffic floods across switches causing unnecessary loops and decreasing network throughput. Unknown Unicast Flood Block (UUFB) blocks the flood of unknown unicast traffic and allows only traffic with known MAC addresses to exit the port. UUFB can be applied on physical ports and Link Aggregation Group (LAG) ports.

FIGURE 82 Unknown Unicast Flooding Example

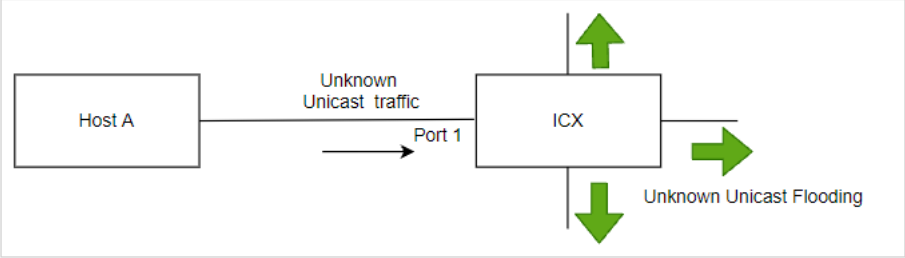
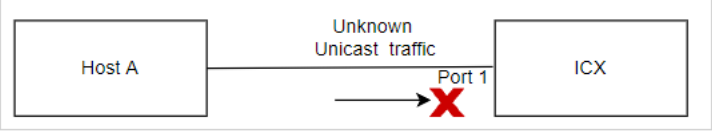


FIGURE 83 Unknown Unicast Flooding Block



Configuring UUFB

Complete the following steps to configure UUFB.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter interface configuration mode and specify the interface that you want to configure for UUFB.

   The interface can be a physical interface or LAG interface.

```
device(config)# interface ethernet 1/1/1
```

3. Enter the **block unknown-unicast** command to block unknown unicast traffic.

```
device(config-interface-eth-1/1/1)# block unknown-unicast
```

The following example configures UUFB on an Ethernet interface.

```
device# configure terminal
device(config)# interface ethernet 1/1/1
device(config-interface-eth-1/1/1)# block unknown-unicast
device(config-interface-eth-1/1/1)# end
```

The following example configures UUFB on a LAG interface.

```
device# configure terminal
device(config)# interface lag 10
device(config-lag-if-lg10)# block unknown-unicast
device(config-lag-if-lg10)# end
```

Displaying UUFB Information

Use the **show uufb enabled-ports** command to display the ports that are enabled for Unknown Unicast Flood Block (UUFB).

```
device# show uufb enabled-ports
UUFB Enabled Ports
==================
eth 1/1/1
eth 1/1/2
```

VLANs

VLAN Overview

A virtual LAN (VLAN) is used to partition a computer network at Layer 2 into a broadcast domain, which is uniquely identified by a VLAN ID. VLAN helps to split a network that acts as a separate network, but are connected to a physical network. Using VLAN tagging, the network administrators can virtually group networks together even if they are not physically connected to the same network switch.

Based on various parameters like users, groups, or roles, VLAN can be used to segment traffic on a network. Based on network requirements, the network administrators are able to handle network traffic by upscaling VLANs securely and manage network efficiently.

The following sections provide details about the VLAN types and features supported on the RUCKUS family of switches.

VLAN Support on RUCKUS Devices

You can configure the following type of VLAN on RUCKUS devices:

- Layer 2 port-based VLAN - a set of physical ports that share a common, exclusive Layer 2 broadcast domain.

When a FastIron device receives a packet on a port that is a member of a VLAN, the device forwards the packet based on the following VLAN hierarchy:

- When the packet can be forwarded at Layer 2, the device forwards the packet on all the ports within the receiving port-based VLAN.

Layer 2 Port-Based VLANs

On all RUCKUS devices, you can configure port-based VLANs. A port-based VLAN is a subset of ports on a RUCKUS device that constitutes a Layer 2 broadcast domain.

By default, all the ports on a RUCKUS device are members of the default VLAN. Thus, all the ports on the device constitute a single Layer 2 broadcast domain.

You can configure multiple port-based VLANs. You can configure up to 4094 port-based VLANs on a Layer 2 Switch or Layer 3 Switch. On both device types, valid VLAN IDs are 1 - 4095. You can configure up to the maximum number of VLANs within that ID range.

NOTE
VLAN IDs 4087 is reserved for RUCKUS internal use only. VLAN 4094 is reserved for use by Single STP. If you want to use VLANs 4090, 4091 and 4092 as configurable VLANs, you can assign them to different VLAN IDs. For more information, refer to Assigning a Different VLAN ID to Default and Reserved VLANs on page 280.

The following behaviors apply to VLANs:

- All interfaces will be untagged members of the default VLAN or any non-default VLAN, unless you explicitly remove this untagged interface from the default VLAN.
- When you configure an interface as untagged member of a non-default VLAN, that interface will be moved from the default VLAN to the configured VLAN.
- When you configure an interface as a tagged member of a non-default VLAN, the untagged VLAN membership of the interface will not be modified, be it default-VLAN or non-default VLAN. You can configure default VLAN port membership.
- You can remove untagged membership of an interface using the **no untagged ethernet** command within the default VLAN node.
- An interface should be a member of at least one VLAN at any given time.
- An interface will be moved to the default VLAN when the last non-default VLAN is removed on that interface (tagged or untagged).
- When an untagged membership of an interface is removed from a non-default VLAN, the interface will be added back to the default VLAN as an untagged interface.

  NOTE
  An untagged interface can be configured as tagged in other user VLANs, and a tagged interface can be configured as untagged in a default VLAN or in non-default VLANs.

NOTE
In RUCKUS 08.0.90 and later releases, there will no longer be any link flap when a port is being added as a tagged member to a VLAN for the first time or when a port is removed from the last tagged VLAN. External devices that may have relied on this link flap in the past for any kind of renegotiation must be reconfigured appropriately, or the user must manually flap the interface.

Because each port-based VLAN is a separate Layer 2 broadcast domain, each VLAN can be configured to run a separate instance of the Spanning Tree Protocol (STP). Layer 2 traffic is bridged within a port-based VLAN, and Layer 2 broadcasts are sent to all the ports within the VLAN.

Configuring Port-Based VLANs

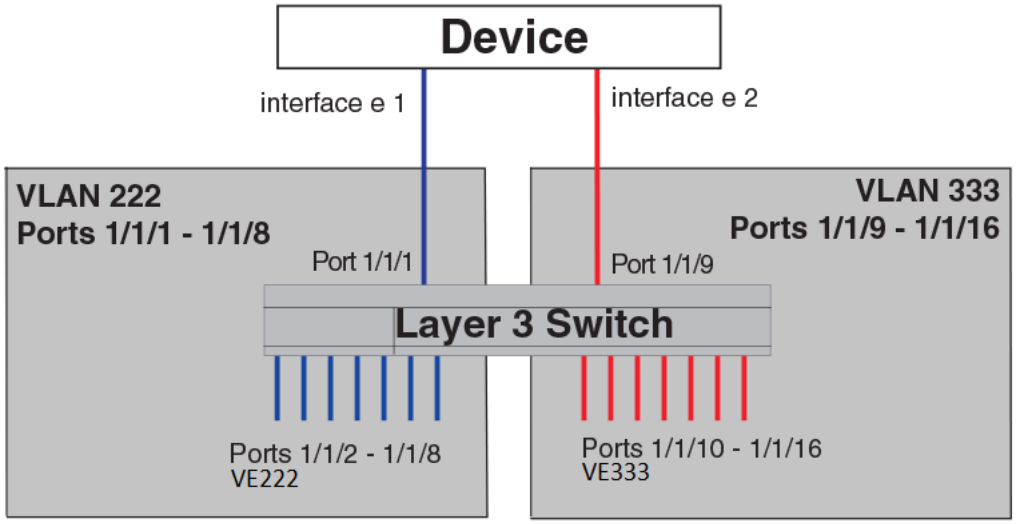
Port-based VLANs allow you to provide separate spanning tree protocol (STP) domains or broadcast domains on a port-by-port basis.

This section describes how to perform the following tasks for port-based VLANs using the CLI:

- Create a VLAN
- Delete a VLAN
- Modify a VLAN
- Change a VLAN priority
- Enable or disable STP on the VLAN

The following figure shows a simple port-based VLAN configuration using a single RUCKUS Layer 2 switch. All ports within each VLAN are untagged. One untagged port within each VLAN is used to connect the Layer 2 switch to a Layer 3 switch for Layer 3 connectivity between the two port-based VLANs.

FIGURE 84 Port-Based VLANs 222 and 333

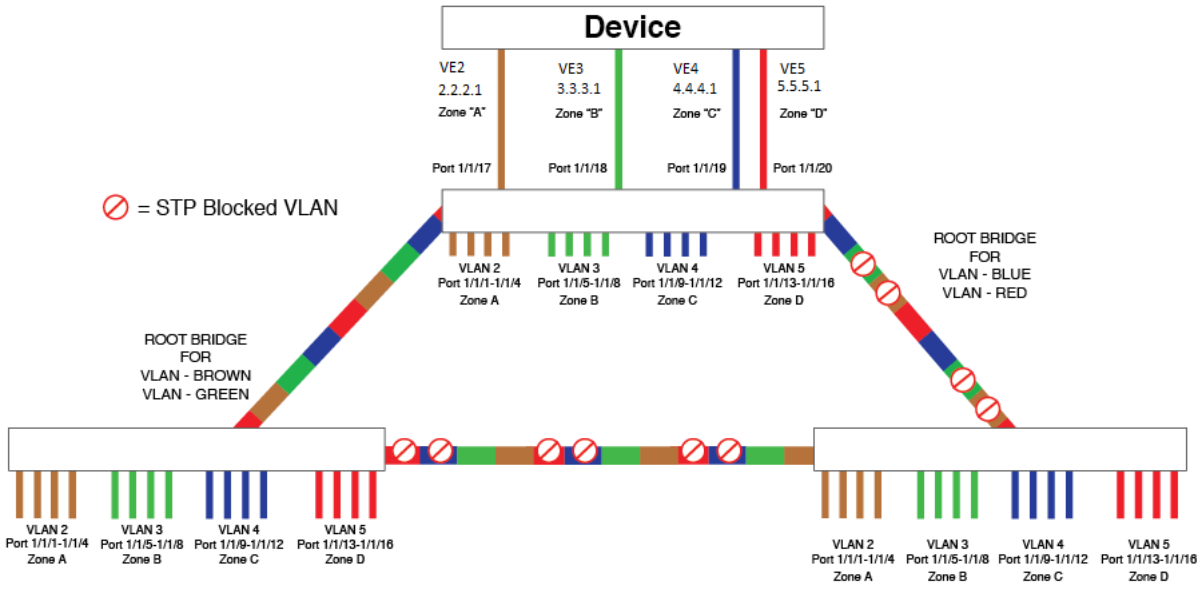


To create the two port-based VLANs shown in the above figure, enter the following commands.

```
device(config)# vlan 222 by port
device(config-vlan-222)# untagged ethernet 1/1/1 to 1/1/8
device(config-vlan-222)# vlan 333 by port
device(config-vlan-333)# untagged ethernet 1/1/9 to 1/1/16
```

The following figure shows a more complex port-based VLAN configuration using multiple Layer 2 switches and IEEE 802.1Q VLAN tagging. The backbone link connecting the three Layer 2 switches remains untagged. One untagged port within each port-based VLAN on Device-A connects each separate network wide Layer 2 broadcast domain to the router for Layer 3 forwarding between broadcast domains. The STP priority is configured to force Device-A to be the root bridge for VLANs RED and BLUE. The STP priority on Device-B is configured so that Device-B is the root bridge for VLANs GREEN and BROWN.

FIGURE 85 More Complex Port-Based VLAN



To configure the Port-based VLANs on the Layer 2 switches in the previous figure, use the following method.

Configuring Port-Based VLANs on Device-A

Enter the following commands to configure Device-A.

1. Enable the privileged EXEC mode and enter the global configuration command.

```
device# enable
device# configure terminal
```

2. Enter the device name.

```
device(config)# hostname device-A
```

3. Configure a VLAN by assigning an ID to the VLAN and name it.

```
device-A(config)# vlan 2 name BROWN
```

4. Configure the following ports as untagged member of VLAN 2.

```
device-A(config-vlan-2)# untagged ethernet 1/1/1 to 1/1/4 ethernet 1/1/17
```

5. If ports are not a member of the above specified VLAN, then add the ports as a tagged member of that VLAN.

```
device-A(config-vlan-2)# tagged ethernet 1/1/25 to 1/1/26
```

6. Enable STP using the **spanning-tree** command.

```
device-A(config-vlan-2)# spanning-tree
```

7. Follow the same steps for other VLANs with the addition of changing the port priority for VLAN 4 and VLAN 5.

```
device-A(config-vlan-2)# vlan 3 name GREEN
device-A(config-vlan-3)# untagged ethernet 1/1/5 to 1/1/8 ethernet 1/1/18
device-A(config-vlan-3)# tagged ethernet 1/1/25 to 1/1/26
device-A(config-vlan-3)# spanning-tree
device-A(config-vlan-3)# vlan 4 name BLUE
device-A(config-vlan-4)# untagged ethernet 1/1/9 to 1/1/12 ethernet 1/1/19
device-A(config-vlan-4)# tagged ethernet 1/1/25 to 1/1/26
device-A(config-vlan-4)# spanning-tree
device-A(config-vlan-4)# spanning-tree priority 500
device-A(config-vlan-4)# vlan 5 name RED
device-A(config-vlan-5)# untagged ethernet 1/1/13 to 1/1/16 ethernet 1/1/20
device-A(config-vlan-5)# tagged ethernet 1/1/25 to 1/1/26
device-A(config-vlan-5)# spanning-tree
device-A(config-vlan-5)# spanning-tree priority 500
device-A(config-vlan-5)# end
device-A# write memory
```

Follow the above same method for Device-B and Device-C.

Configuring Port-Based VLANs on Device-B

Enter the following commands to configure Device-B.

```
device# enable
device# configure terminal
device(config)# hostname Device-B
device-B(config)# vlan 2 name BROWN
device-B(config-vlan-2)# untagged ethernet 1/1/1 to 1/1/4
device-B(config-vlan-2)# tagged ethernet 1/1/25 to 1/1/26
device-B(config-vlan-2)# spanning-tree
device-B(config-vlan-2)# spanning-tree priority 500
device-B(config-vlan-2)# vlan 3 name GREEN
device-B(config-vlan-3)# untagged ethernet 1/1/5 to 1/1/8
device-B(config-vlan-3)# tagged ethernet 1/1/25 to 1/1/26
device-B(config-vlan-3)# spanning-tree
device-B(config-vlan-3)# spanning-tree priority 500
device-B(config-vlan-3)# vlan 4 name BLUE
device-B(config-vlan-4)# untagged ethernet 1/1/9 to 1/1/12
device-B(config-vlan-4)# tagged ethernet 1/1/25 to 1/1/26
device-B(config-vlan-4)# vlan 5 name RED
device-B(config-vlan-5)# untagged ethernet 1/1/13 to 1/1/16
device-B(config-vlan-5)# tagged ethernet 1/1/25 to 1/1/26
device-B(config-vlan-5)# end
device-B# write memory
```

Configuring Port-Based VLANs on Device-C

Enter the following commands to configure Device-C.

```
device# enable
device# configure terminal
device(config)# hostname Device-C
device-C(config)# vlan 2 name BROWN
device-C(config-vlan-2)# untagged ethernet 1/1/1 to 1/1/4
device-C(config-vlan-2)# tagged ethernet 1/1/25 to 1/1/26
device-C(config-vlan-2)# vlan 3 name GREEN
device-C(config-vlan-3)# untagged ethernet 1/1/5 to 1/1/8
device-C(config-vlan-3)# tagged ethernet 1/1/25 to 1/1/26
device-C(config-vlan-3)# vlan 4 name BLUE
device-C(config-vlan-4)# untagged ethernet 1/1/9 to 1/1/12
device-C(config-vlan-4)# tagged ethernet 1/1/25 to 1/1/26
device-C(config-vlan-4)# vlan 5 name RED
device-C(config-vlan-5)# untagged ethernet 1/1/13 to 1/1/16
device-C(config-vlan-5)# tagged ethernet 1/1/25 to 1/1/26
```

```
device-C(config-vlan-5)# end
device-C# write memory
```

Modifying a Port-Based VLAN

You can make the following modifications to a port-based VLAN:

- Add or delete a VLAN port
- Enable or disable STP

Removing a Port-Based VLAN

Suppose you want to remove VLAN 5 from the example in Figure 85 on page 264. To do so, use the following procedure.

1. Access the global configuration mode on Device-A by entering the following commands.

```
device-A# enable
No password has been assigned yet...
device-A# configure terminal
device-A(config)#
```

2. Enter the following command to remove the port VLAN 5.

```
device-A(config)# no vlan 5
```

3. Enter the following commands to exit the global configuration mode and save the configuration to the system-config file on flash memory.

```
device-A(config)# end
device-A# write memory
```

4. Repeat step 1 through step 3 on Device-B and Device-C.

Moving Untagged Port Membership from One Non-Default VLAN to Another

You can move untagged ports from one VLAN to another, using the wrapper CLI **vlan-config move untagged**. This CLI can be used to modify untagged VLAN membership of a port.

NOTE
This wrapper CLI performs several CLI calls internally. If any internal CLI fails to execute, corresponding message will be displayed.

1. Access the global configuration mode on entering the following commands.

```
device# enable
No password has been assigned yet...
device# configure terminal
device(config)#
```

2. Access the level of the CLI for configuring port-based VLAN, by entering the following commands.

```
device(config)# vlan 20
device(config-vlan-20)# untagg e 1/1/2
Added untagged port(s) ethe 1/1/2 to port-vlan 20.
device(config-vlan-20)#
device(config-vlan-20)#
device(config-vlan-20)# inter e 1/1/2
device(config-if-e1000-1/1/2)#
```

3. Enter the following command to remove the untagged port and add it to another VLAN.

```
device(config-if-e40000-1/1/2)# vlan-config move untagged 100
```

4. Enter the following commands to exit the VLAN configuration mode and save the configuration to the system-config file on flash memory.

```
device(config-if-e40000-1/1/2)# end
device# write memory
```

If the new VLAN is not yet configured, the command will create it and the port will be added to it.

NOTE
The **vlan-config move untagged** is a non-savable command. It can be applicable to multiple interfaces. This port should belong to a non-default VLAN before moving to another VLAN. The command does not work if the port belongs to a member of default vlan only.

Removing VLANs from Physical Ports

You can remove VLANs from an Ethernet port, and the port will be added back to the default VLAN as an untagged member port.

The **vlan-config remove all** feature is used to remove all VLANs from physical ports (except the default VLAN and reserved VLANs). The untagged ports will be moved to the default VLAN after removing the ports from this VLAN and the tagged ports will be moved to the default VLAN if they are not members of any other VLAN.

NOTE
The **vlan-config remove all** feature is supported on all ICX devices.

Removing VLANs from a Physical Port

To remove VLANs from a physical port, complete the following steps.

1. From global configuration mode, enter interface Ethernet configuration mode.

```
device(config)# interface ethernet 1/1/1
```

2. Enter the **vlan-config remove all** command to remove all VLANs from the Ethernet port.
   - Enter the **vlan-config remove all** command to remove the VLANs from the Ethernet port.

   The following examples indicate how the command can be used.

```
device(config-if-e40000-1/1/1)# vlan-config remove all
(port(s) will be removed from 300 VLANs in single execution)
```

To remove all VLANs from a physical port, enter commands such as the following.

```
device(config)# interface ethernet 1/1/1
device(config-if-e40000-1/1/1)# vlan-config remove all
```

NOTE

VLAN groups cannot be removed from the ports using this command.

The following is an example to show that vlan-groups cannot be removed from the ports.

```
device(config)# vlan-group 1 vlan 1001 to 1005
device(config-vlan-group-1)# tag ethernet 1/1/1
Added tagged port(s) ethernet 1/1/1 to vlan-group 1.
device(config-vlan-group-1)# exit
device(config)#
device(config)# interface ethernet 1/1/1
device(config-if-e4000-1/1/1)# vlan-config remove all
Port(s) ethernet 1/1/1 cannot be removed from VLANs 1 1001 to 1005
```

To remove all VLANs from more than one physical port, enter commands such as the following.

```
device(config)# interface ethernet 1/1/1 ethernet 1/1/5
device(config-mif-1/1/1,1/1/5)# vlan-config remove all
```

Add and Remove All Tagged VLANs

A new wrapper is introduced to add and remove selective VLAN at the interface level.

To enable addition and deletion of a tagged port selectively to VLAN, the existing commands **vlan-config add** and **vlan-config remove** is upgraded to support selective VLANs and VLAN range. The **vlan-config add** command is modified to accept optional parameter of VLAN or VLAN range. The **vlan-config remove** command is modified to accept optional parameter of VLAN.

Adding Selective VLAN

The command will create a new VLAN and add the interface to it, if interface being added is the first interface. The command can add port to non-active non-configured VLAN. The command is available in MIF mode. The maximum VLAN or VLAN range supported in a single input is 1024.

```
device(config-if-e1000-1/1/1)# vlan-config add
  all-tagged    Add interface to all VLAN(s)
  tagged-vlan   Add interface to VLAN(s)
device(config-if-e1000-1/1/1)# vlan-config add tagged-vlan 2 3 4
Added tagged port(s) ethe 1/1/1 to port-vlan 2.
Added tagged port(s) ethe 1/1/1 to port-vlan 3.
Added tagged port(s) ethe 1/1/1 to port-vlan 4.
device(config-if-e1000-1/1/1)#
device(config-if-e1000-1/1/1)# vlan-config add tagged-vlan 1001 to 1005
Added tagged port(s) ethe 1/1/1 to port-vlan 1001.
Added tagged port(s) ethe 1/1/1 to port-vlan 1002.
Added tagged port(s) ethe 1/1/1 to port-vlan 1003.
Added tagged port(s) ethe 1/1/1 to port-vlan 1004.
Added tagged port(s) ethe 1/1/1 to port-vlan 1005.
device(config-if-e1000-1/1/1)#
```

NOTE

The CLI prompt will not be available for next command until port is added to all VLANs in the system

Removing Selective VLAN

The max VLAN or VLAN Range supported in single input will be 1024.

```
device(config-if-e40000-1/1/1)# vlan-config remove 2000 to 2005
(port(s) will be removed from 300 VLANs using this command)
```

NOTE
The command is a non-savable command, which removes the interface as a tagged member of all configured and active VLAN. The command is available in MIF mode. The command line interface prompt will not be available for next command until the port is added to all the VLANs in the system.

Multi-Range VLANs

The multi-range VLAN feature allows users to use a single command to create and configure multiple VLANs. These VLANs can be continuous, for example, from 2 to 7, or discontinuous, for example, 2 4 7.

Creating and Configuring a Multi-Range VLAN

You can create and configure multi-range VLANs using various VLAN parameters. These VLANs can be continuous or discontinuous or a combination of continuous and discontinuous VLANs.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Create a multi-range VLAN using one of the following methods.
   - Create a continuous range of VLANs.

```
device(config)# vlan 2 to 7
```

   - Create discontinuous VLANs.

```
device(config)# vlan 2 4 7
```

   - Create continuous and discontinuous VLANS.

```
device(config)# vlan 16 17 20 to 24
```

     NOTE
     You can create or configure a maximum number of 1024 VLANs with a single command.

3. Add tagged port or a range of ports to multi-range VLANs.

```
device(config-mvlan-2-7)# tagged ethernet 1/1/3 to 1/1/4
device(config-mvlan-16*24)# tagged ethernet 1/1/1
```

   In the first example, the tagged Ethernet ports 1/1/3 and 1/1/4 are added to VLANs, 2 through 7. In the second example, the tagged Ethernet port 1/1/1 is added to VLANs 16, 17, 20, 21, 22, 23, and 24.

4. Configure a static MAC address on a range of Ethernet ports for multi-range VLANs.

```
device(config-mvlan-2-7)# static-mac-address 0000.0063.67ff ethernet 1/1/3 to 1/1/4
```

5. Monitor a mirrored port on multi-range VLANs.

```
device(config-mvlan-2-7)# monitor ethernet 1/1/1
```

6. Set IGMP snooping for multi-range VLANs.

```
device(config-mvlan-2-7)# multicast active
device(config-mvlan-2-7)# multicast6 passive
```

7. Enable IP source guard for multi-range VLANs.
   - Enable IP source guard on all ports of multiple VLANs.

     ```
     device(config-mvlan-2-7)# source-guard enable
     ```

   - Enable IP source guard on a range of ports of multiple VLANs.

     ```
     device(config-mvlan-2-7)# source-guard enable ethernet 1/1/3 to 1/1/4
     ```

8. Enable 802.1W spanning tree on multi-range VLANs.

   ```
   device(config-mvlan-2-7)# spanning-tree 802-1w
   ```

9. Exit MVLAN configuration mode.

   ```
   device(config-mvlan-2-7)# exit
   ```

10. (Optional) Delete a multi-range VLAN using one of the following methods.
    - Delete a continuous range of VLANs.

      ```
      device(config)# no vlan 2 to 7
      ```

    - Delete discontinuous VLANs.

      ```
      device(config)# no vlan 2 4 7
      ```

    - Delete continuous and discontinuous VLANS.

      ```
      device(config)# no vlan 16 17 20 to 24
      ```

The following example shows how to configure multi-range VLANs using various VLAN parameters.

```
device# configure terminal
device(config)# vlan 2 to 7
device(config-mvlan-2-7)# tagged ethernet 1/1/3 to 1/1/4
device(config-mvlan-2-7)# static-mac-address 0000.0063.67ff ethernet 1/1/3 to 1/1/4
device(config-mvlan-2-7)# monitor ethernet 1/1/1
device(config-mvlan-2-7)# multicast active
device(config-mvlan-2-7)# source-guard enable
device(config-mvlan-2-7)# spanning-tree 802-1w
device(config-mvlan-2-7)# exit
device(config)# no vlan 2 to 7
```

Displaying Multi-Range VLAN Information

You can display the multi-range VLAN information from the multi-range VLAN configuration mode.

The following example displays the STP information for multiple VLANs (VLANs 4, 5, and 6).

```
device(config)# vlan 4 to 6
device(config-mvlan-4-6)# show 802-1w

--- VLAN 4 [ STP Instance owned by VLAN 4 ] ----------------------------
Bridge IEEE 802.1W Parameters:
Bridge Bridge Bridge Bridge Force tx
Identifier MaxAge Hello FwdDly Version Hold
hex sec sec sec cnt
8000002022227700 20 2 15 Default 3
RootBridge RootPath DesignatedBri- Root Max Fwd Hel
Identifier Cost dge Identifier Port Age Dly lo
hex hex sec sec sec
8000002022227700 0 8000002022227700 Root 20 15 2
Port IEEE 802.1W Parameters:
```

```
<--- Config Params --><-------------- Current state ----------------->
Port Pri PortPath P2P Edge Role State Designa- Designated
Num Cost Mac Port ted cost bridge
1/1/1 128 20000 F F DESIGNATED FORWARDING 0 8000002022227700
--- VLAN 5 [ STP Instance owned by VLAN 5 ] ----------------------------
Bridge IEEE 802.1W Parameters:
Bridge Bridge Bridge Bridge Force tx
Identifier MaxAge Hello FwdDly Version Hold
hex sec sec sec cnt
8000002022227700 20 2 15 Default 3
RootBridge RootPath DesignatedBri- Root Max Fwd Hel
Identifier Cost dge Identifier Port Age Dly lo
hex hex sec sec sec
8000002022227700 0 8000002022227700 Root 20 15 2
Port IEEE 802.1W Parameters:
<--- Config Params --><-------------- Current state ----------------->
Port Pri PortPath P2P Edge Role State Designa- Designated
Num Cost Mac Port ted cost bridge
1/1/1 128 20000 F F DESIGNATED FORWARDING 0 8000002022227700
--- VLAN 6 [ STP Instance owned by VLAN 6 ] ----------------------------
Bridge IEEE 802.1W Parameters:
Bridge Bridge Bridge Bridge Force tx
Identifier MaxAge Hello FwdDly Version Hold
hex sec sec sec cnt
8000002022227700 20 2 15 Default 3
RootBridge RootPath DesignatedBri- Root Max Fwd Hel
Identifier Cost dge Identifier Port Age Dly lo
hex hex sec sec sec
8000002022227700 0 8000002022227700 Root 20 15 2
Port IEEE 802.1W Parameters:
<--- Config Params --><-------------- Current state ----------------->
Port Pri PortPath P2P Edge Role State Designa- Designated
Num Cost Mac Port ted cost bridge
1/1/1 128 20000 F F DESIGNATED FORWARDING 0 8000002022227700
```

You can use various **show** parameters for the specified VLAN range from the multi-range VLAN configuration mode. The output of these **show** commands display information about the specified VLANs only.

The following example displays the MAC address table for a specific VLAN.

```
device# show mac-address vlan 1 0000.0000.0001

Total active entries from all ports = 16
MAC-Address Port Type Index
0000.0000.0001 1/1/1 Dynamic NA
```

The following example displays the STP information for a specific VLAN.

```
device# show span vlan 100

STP instance owned by VLAN 100
Global STP (IEEE 802.1D) Parameters:
VLAN Root Root Root Prio Max He- Ho- Fwd Last Chg Bridge
ID ID Cost Port rity Age llo ld dly Chang cnt Address
Hex sec sec sec sec sec
100 8000cc4e24b46fcc 0 Root 8000 20 2 1 15 11 1 cc4e24b46fcc
Port STP Parameters:
Port Prio Path State Fwd Design Designated Designated
Num rity Cost Trans Cost Root Bridge
Hex
1/1/1 80 4 FORWARDING 1 0 8000cc4e24b46fcc 8000cc4e24b46fcc
1/1/2 80 4 FORWARDING 1 0 8000cc4e24b46fcc 8000cc4e24b46fcc
1/1/3 80 4 FORWARDING 1 0 8000cc4e24b46fcc 8000cc4e24b46fcc
1/1/4 80 4 FORWARDING 1 0 8000cc4e24b46fcc 8000cc4e24b46fcc
1/1/5 80 4 FORWARDING 1 0 8000cc4e24b46fcc 8000cc4e24b46fcc
1/1/6 80 4 FORWARDING 1 0 8000cc4e24b46fcc 8000cc4e24b46fcc
1/1/7 80 4 FORWARDING 1 0 8000cc4e24b46fcc 8000cc4e24b46fcc
1/1/8 80 4 FORWARDING 1 0 8000cc4e24b46fcc 8000cc4e24b46fcc
lg1 80 4 FORWARDING 1 0 8000cc4e24b46fcc 8000cc4e24b46fcc
lg256 80 4 FORWARDING 1 0 8000cc4e24b46fcc 8000cc4e24b46fcc
```

The following example displays the VLAN information.

```
device# show vlan
Total PORT-VLAN entries: 4
Maximum PORT-VLAN entries: 1024

Legend: [Stk=Stack-Id, S=Slot]

Total PORT-VLAN entries: 4
Maximum PORT-VLAN entries: 1024

Legend: [Stk=Stack-Id, S=Slot]

PORT-VLAN 1, Name DEFAULT-VLAN, Priority level0, On
 Untagged Ports: (U1/M1)   1   2   3   4   5   6   7   8   9  10  11  12
 Untagged Ports: (U1/M1)  13  14  15  16  17  18  19  20  21  22  23  24
 Untagged Ports: (U1/M1)  25  26  27  28  29  30  31  32  33  34  35  36
 Untagged Ports: (U1/M1)  37  38  39  40  41  42  43  44  45  46  47  48

 Untagged Ports: (U1/M2)   1   2   3   4
   Tagged Ports: None
 Mac-Vlan Ports: None
     Monitoring: Disabled
Total PORT-VLAN entries: 4
Maximum PORT-VLAN entries: 1024

Legend: [Stk=Stack-Id, S=Slot]

PORT-VLAN 4, Name [None], Priority level0, On
 Untagged Ports: None
   Tagged Ports: None
 Mac-Vlan Ports: None
     Monitoring: Disabled
Total PORT-VLAN entries: 4
Maximum PORT-VLAN entries: 1024

Legend: [Stk=Stack-Id, S=Slot]

PORT-VLAN 5, Name [None], Priority level0, On
 Untagged Ports: None
   Tagged Ports: None
 Mac-Vlan Ports: None
     Monitoring: Disabled
Total PORT-VLAN entries: 4
Maximum PORT-VLAN entries: 1024

Legend: [Stk=Stack-Id, S=Slot]

PORT-VLAN 6, Name [None], Priority level0, On
 Untagged Ports: None
   Tagged Ports: None
 Mac-Vlan Ports: None
     Monitoring: Disabled
```

The following example displays the VSRP commands for a specific VLAN.

```
device# show vsrp vlan 100 100

VLAN 100
auth-type no authentication
VRID 100
========
State Administrative-status Advertise-backup Preempt-mode save-current
master enabled disabled true false
Parameter Configured Current Unit/Formula
priority 100 50 (100-0)*(2.0/4.0)
hello-interval 1 1 sec/1
dead-interval 3 3 sec/1
hold-interval 3 3 sec/1
initial-ttl 2 2 hops
next hello sent in 00:00:00.3
```

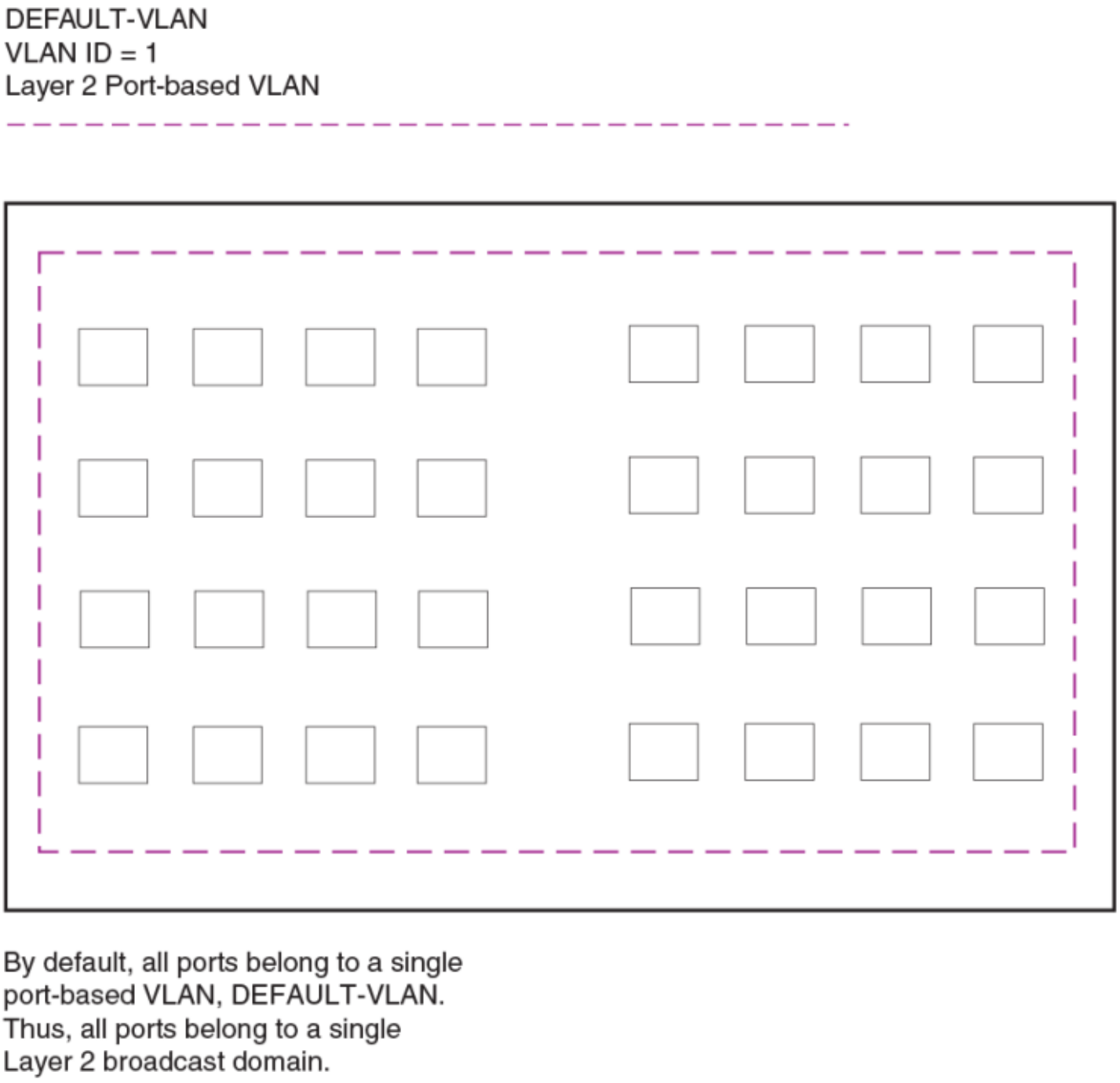
```
Member ports: ethe 1/2/5 to 1/2/8
Operational ports: ethe 1/2/5 ethe 1/2/8
Forwarding ports: ethe 1/2/5 ethe 1/2/8
Restart ports: 1/2/5(1) 1/2/6(1) 1/2/7(1) 1/2/8(1)
```

Default VLAN

By default, all the ports on a RUCKUS device are in a single port-based VLAN. This VLAN is called the DEFAULT-VLAN and is VLAN number 1.

The following figure shows an example of the default Layer 2 port-based VLAN.

FIGURE 86 Default Layer 2 Port-Based VLAN



When you configure a port-based VLAN, one of the configuration items you provide is the ports that are in the VLAN.

NOTE
Information for the default VLAN is available only after you define another VLAN.

Some network configurations may require that a port be able to reside in two or more Layer 2 broadcast domains (port-based VLANs). In this case, you can enable a port to reside in multiple port-based VLANs by tagging the port. Refer to the following section.

If your network requires that you use VLAN ID 1 for a user-configured VLAN, you can reassign the default VLAN to another valid VLAN ID. Refer to Assigning a Different VLAN ID to Default and Reserved VLANs on page 280.

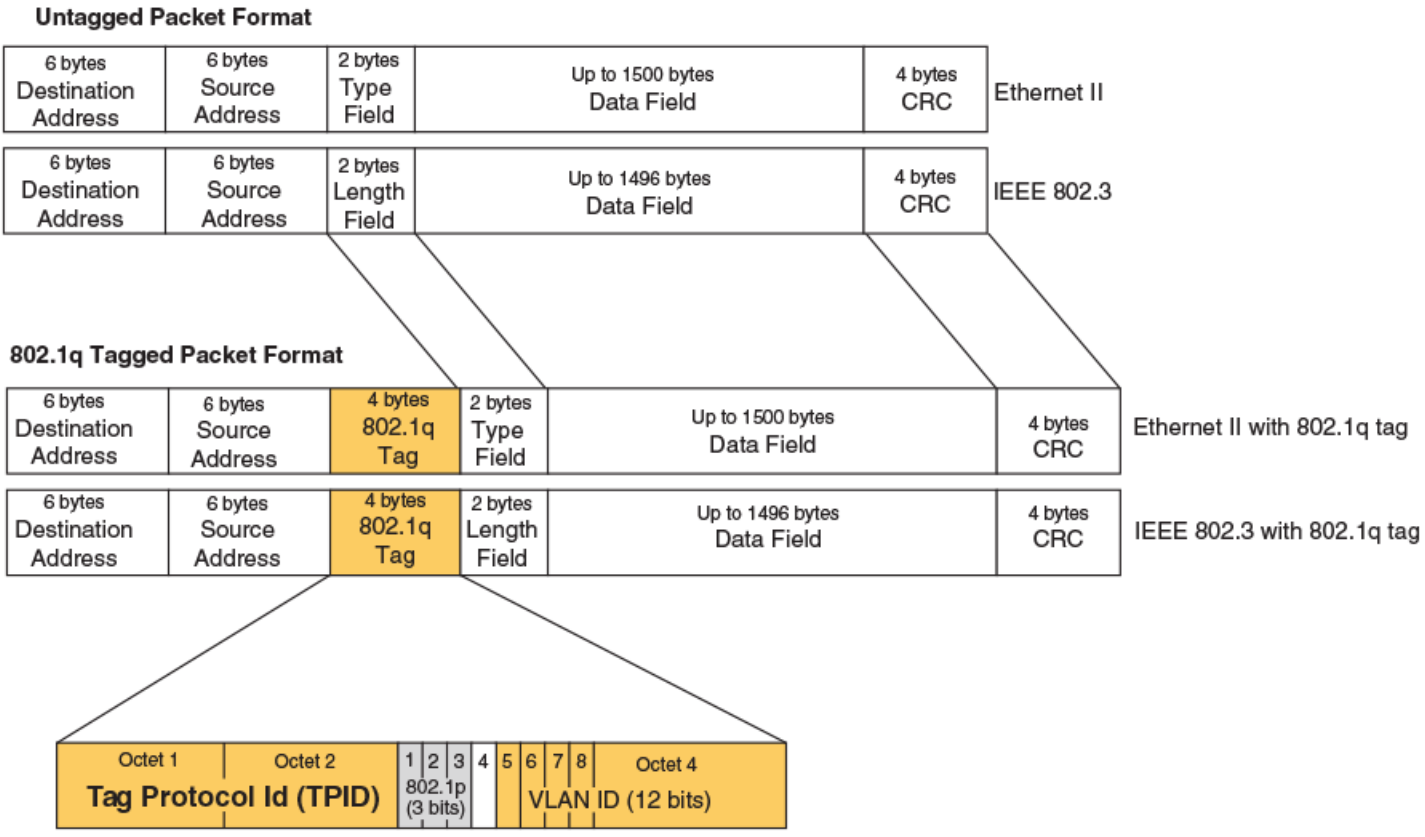
802.1Q Tagging

802.1Q tagging is an IEEE standard that allows a networking device to add information to a Layer 2 packet in order to identify the VLAN membership of the packet. RUCKUS devices tag a packet by adding a four-byte tag to the packet. The tag contains the tag value, which identifies the data as a tag, and also contains the VLAN ID of the VLAN from which the packet is sent.

- The default tag value is 8100 (hexadecimal). This value comes from the 802.1Q specification. You can change this tag value on a global basis on RUCKUS devices if needed to be compatible with other vendors' equipment.
- The VLAN ID is determined by the VLAN on which the packet is being forwarded.

The following figure shows the format of packets with and without the 802.1Q tag. The tag format is vendor-specific. To use the tag for VLANs configured across multiple devices, make sure all the devices support the same tag format.

FIGURE 87 Packet Containing a RUCKUS 802.1Q VLAN Tag

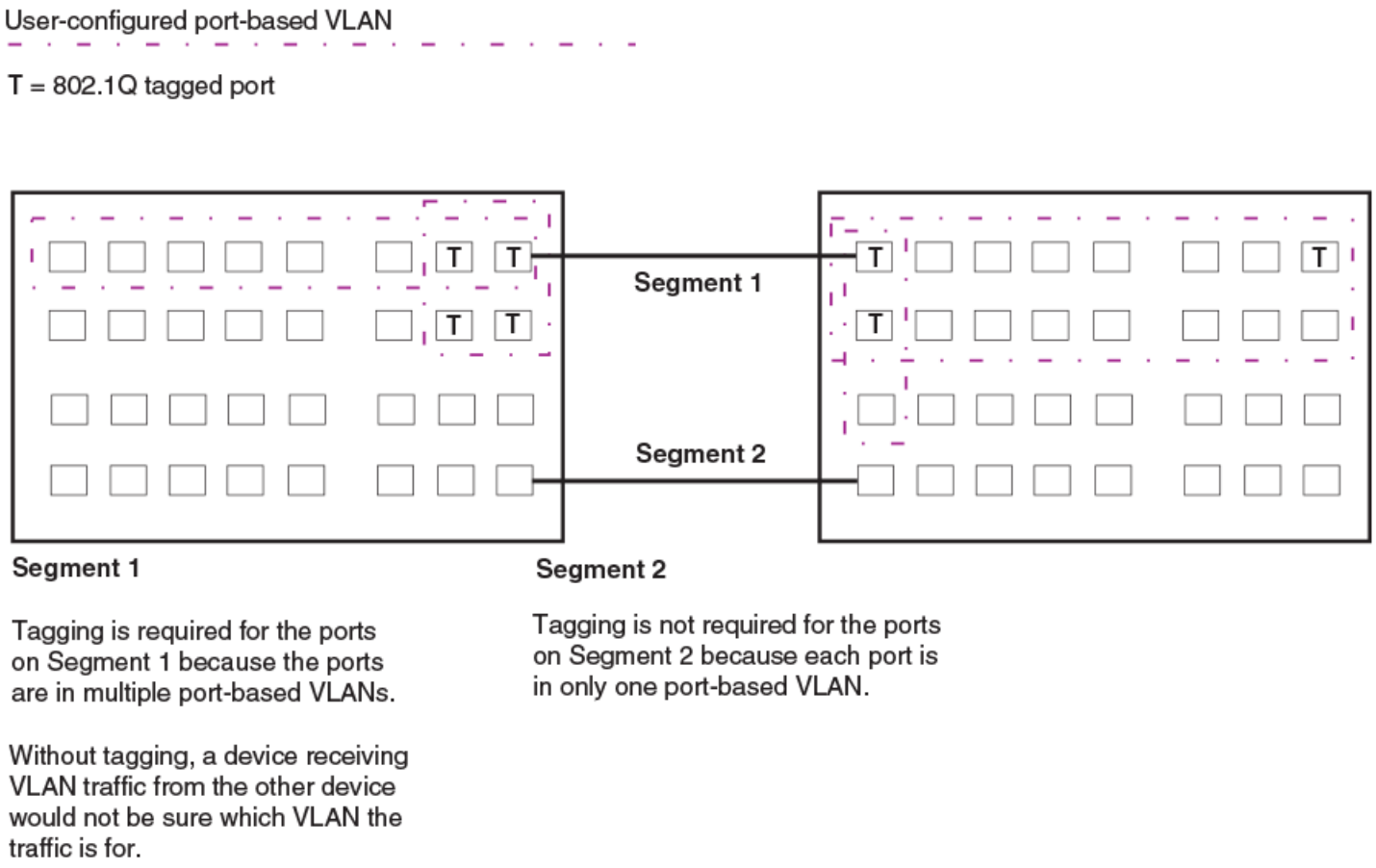


If you configure a VLAN that spans multiple devices, you need to use tagging only if a port connecting one of the devices to the other is a member of more than one port-based VLAN. If a port connecting one device to the other is a member of only a single port-based VLAN, tagging is not required.

If you use tagging on multiple devices, each device must be configured for tagging and must use the same tag value. In addition, the implementation of tagging must be compatible on the devices. The tagging on all RUCKUS devices is compatible with other RUCKUS devices.

The following figure shows an example of two devices that have the same Layer 2 port-based VLANs configured across them. Notice that only one of the VLANs requires tagging.

FIGURE 88 VLANs Configured Across Multiple Devices



Support for 802.1ad (Q-in-Q) tagging

RUCKUS devices provide finer granularity for configuring 802.1Q tagging, enabling you to configure 802.1Q tag-profiles on a group of ports, thereby enabling the creation of two identical 802.1Q tags (802.1ad tagging) on a single device. This enhancement improves SAV interoperability between RUCKUS devices and other vendors' devices that support the 802.1Q tag-profiles, but are not very flexible with the tag-profiles they accept.

802.1ad Tagging for ICX Devices

The following enhancement allows the ICX devices to use Q-in-Q and SAV, by allowing the changing of a tag profile for ports:

- Tag profiles on a single port or a group of ports can be configured to point to the global tag profile.

For example applications and configuration details, refer to 802.1ad Tagging Configuration on page 300.

To configure a global tag profile, enter the following command in configuration mode.

```
device(config)# tag-profile 9500
```

To direct individual ports or a range of ports to this tag profile, enter commands similar to the following example.

```
device(config)# interface ethernet 1/1/1
device(config-if-e1000-1/1/1)# tag-profile enable
device(config-mif-1/1/1,1/2/1)# tag-profile enable
```

Spanning Tree Protocol

The default state of Spanning Tree Protocol (STP) depends on the device type:

- STP is enabled by default on RUCKUS Layer 2 switches.
- On RUCKUS Layer 3 switches, STP is enabled on the default VLAN after reload if there is no preconfiguration. If there is a configuration on a Layer 3 switch and if the configuration was generated by a Layer 3 image, then there is no change of STP behavior after reload.

Also by default, each port-based VLAN has a separate instance of STP. Thus, when STP is globally enabled, each port-based VLAN on the device runs a separate spanning tree.

You can enable or disable STP on the following levels:

- Globally: Affects all ports on the device.

  NOTE
  If you configure a port-based VLAN on the device, the VLAN has the same STP state as the default STP state on the device. Therefore, on Layer 2 switches, new VLANs have STP enabled by default. On Layer 3 switches, new VLANs have STP disabled by default. You can enable or disable STP in each VLAN separately. In addition, you can enable or disable STP on individual ports.

- Port-based VLAN: Affects all ports within the specified port-based VLAN.

STP is a Layer 2 protocol. The STP state of a port-based VLAN determines the STP state for all the Layer 2 broadcasts within the port-based VLAN. This is true even though Layer 3 protocol broadcasts are sent on Layer 2 within the VLAN.

If you enable Single STP (SSTP) on the device, the ports in all VLANs on which STP is enabled become members of a single spanning tree. The ports in VLANs on which STP is disabled are excluded from the single spanning tree.

For more information, refer to #unique_310.

Super-Aggregated VLANs

RUCKUS ICX devices support Super Aggregated VLANs. You can aggregate multiple VLANs within another VLAN. This feature allows you to construct Layer 2 paths and channels. This feature is particularly useful for Virtual Private Network (VPN) applications in which you need to provide a private, dedicated Ethernet connection for an individual client to transparently reach its subnet across multiple networks.

For an application example and configuration information, refer to Super-Aggregated VLAN Configuration on page 292.

LAG Interface and VLAN Membership

A LAG is a set of physical ports that are configured to act as a single logical/virtual interface. Each LAG member port's configuration is based on the configuration of the LAG interface.

If you add a LAG interface to a VLAN, all of the ports in the LAG become members of that VLAN.

Multiple VLAN Membership Rules

- A port can belong to multiple, overlapping Layer 2 port-based VLANs only if the port is a tagged port. Packets sent out of a tagged port use an 802.1Q-tagged frame.

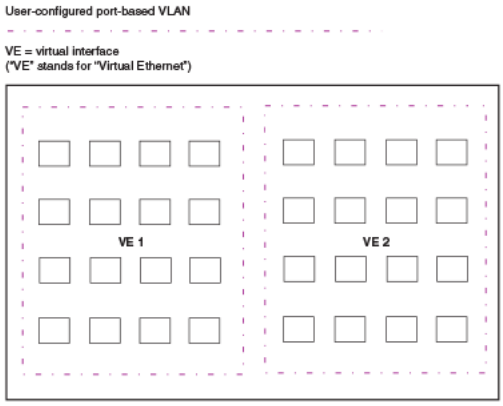
Virtual Routing Interfaces

A virtual routing interface is a logical routing interface that RUCKUS Layer 3 switches use to route Layer 3 protocol traffic between VLANs.

Layer 3 traffic from one VLAN to another must be routed and this is accomplished by virtual routing interfaces. Beginning with FastIron 09.0.10a, the **router-interface ve** command is deprecated. When a VLAN is configured, a virtual routing interface is statically mapped to the VLAN ID. However, only when you use the **interface ve** *ve-num* command does the virtual interface get created. The interface VE cannot be created before its statically mapped VLAN is configured. You must ensure that the *ve-num* corresponds to the VLAN ID of the VLAN under which it is configured. You can further configure other routing parameters on the virtual routing interfaces. For example, to enable a Layer 3 switch to route IP traffic from one VLAN to another, you must create a virtual routing interface for each VLAN and then configure the appropriate IP routing parameters on each of the virtual routing interfaces. If the system default VLAN ID is changed, the *ve-id* associated with it will also change automatically.

The following figure shows an example of VLANs that use virtual routing interfaces for routing.

FIGURE 89 Using Virtual Routing Interfaces for Routing Between VLANs



VLAN and Virtual Routing Interface Groups

RUCKUS ICX FastIron devices support the configuration of VLAN groups. To simplify configuration, you can configure VLAN groups and virtual routing interface groups. When you create a VLAN group, the VLAN parameters you configure for the group apply to all the VLANs within the group. Additionally, you can easily associate the same IP subnet interface with all the VLANs in a group by configuring a virtual routing interface group with the same ID as the VLAN group. You can configure a virtual routing interface group using the **group-router-interface** command and associate the virtual interface routing group with a VLAN group using the **interface group-ve** command.

For configuration information, refer to VLAN Groups and Virtual Routing Interface Group on page 286.

Routing Between VLANs

RUCKUS Layer 3 switches can locally route IP between VLANs defined within a single router. All other routable protocols must be routed by another external router capable of routing the protocol.

Configuring VLAN Virtual Interface

The following configuration steps show basic virtual interface configuration in a network topology that has two devices.

1. Enter the global configuration mode.

```
device# configure terminal
```

2. Create a user VLAN on device 1.

```
device1(config)# vlan 10
```

3. Add an access port to the VLAN.

```
device1(config-vlan-10)# untagged ethernet 1/1/48
```

The port can be a tagged, untagged, or multiple ports.

NOTE
Even without any ports in the VLAN, interface VE can still be configured and ports can be added to the VLAN post configuration of the interface.

4. Configure interface VE.

```
device1(config-vlan-10)# interface ve 10
device1(config-vif-10)#
```

The interface VE is directly mapped to the corresponding VLAN ID.

5. Configure L3 settings.

```
device1(config-vif-10)# ip address 100.0.0.1/24
device1(config-vif-10)# ip ospf area 0
```

NOTE
Router OSPF should already be configured in the global configuration mode as below:

```
device1(config)# router ospf
device1(config-ospf-router)# area 0
```

6. Verify the configuration.

```
device1(config-vif-10)# show run int ve 10
interface ve 10
 ip address 100.0.0.1 255.255.255.0
 ip ospf area 0
!
```

Perform the same configuration on the peer device.

VLAN and VE Pre-Provisioning

Create VLAN without ports feature creates VLAN and puts it in INUSE mode, once you configure the VLAN. This allows applications to configure any feature under VLAN even before adding ports to it.

Some features needs to change VLAN control settings in the hardware when they are configured. So if VLAN is present even before adding ports to it, you should be able to program VLAN control settings when you are configuring the feature under VLAN. You can retain the software configuration and the hardware VLAN control settings even after removing all the ports from the VLAN.

You can remove all the ports from a port-based VLAN without losing the rest of the VLAN configuration. You can configure an IP address even without ports in the VLAN, but the IP address will be effective based on the ports added to the VLAN. If the VLAN has a virtual routing interface, the virtual routing interface IP address is deleted when the ports associated with the interface are deleted. The rest of the VLAN configuration is retained. Ports cannot be moved from or moved to a private VLAN.

Following are few limitations:

- In MRP, ring-interface configuration is not allowed when ports are not part of the VLAN. Ring-interfaces configuration will be lost when all ports removed from VLAN.

- Enabling UDLD for tagged ports requires port to be part of VLAN. Removal of all ports from the VLAN is not allowed when UDLD is enabled on tagged ports.
- Port based IGMP and MLD are not supported.
- In MCT, you can configure keep-alive and session VLANs configuration under cluster configuration, without having any ports in VLAN. However, ICL configuration mandates to have ports in the session VLAN.
- Once MCT cluster is deployed, you are not allowed to remove ports from either session or keep-alive VLANs. You must un-deploy the cluster if there is any change of ports in session or keep-alive VLANs.
- You are allowed to remove all ports from MCT-VLAN but have to deploy the client again after adding ports back to MCT-VLAN.

Sample Configuration

Following is a sample configuration for creating VLAN without ports.

```
device(config)# vlan 500
device(config-vlan-500)# interface ve 500
device(config-vif-500)# ip address 3.3.3.3/8
device(config-vif-500)# sh run vlan 500
vlan 500 by port
!
device(config-vif-500)# sh run int ve 500
interface ve 500
 ip address 3.3.3.3 255.0.0.0
```

NOTE

When there are no ports in the VLAN with VE interface configured in it, the interface status is show as DOWN.

```
device(config-vif-34)# show int ve 34
ve34 is down, line protocol is down
```

When an active port is added to the related VLAN, the VE interface status is shown as UP.

```
device(config-vlan-34)# show int ve 34
ve34 is up, line protocol is up
```

Assigning a Different VLAN ID to Default and Reserved VLANs

You can assign a different VLAN ID to the default VLAN and to reserved VLANs. When you enable port-based VLANs, all ports in the system are added to the default VLAN. By default, the default VLAN ID is "VLAN 1".

The default VLAN is not configurable. If you want to use the VLAN ID "VLAN 1" as a configurable VLAN, you can assign a different VLAN ID to the default VLAN. You must specify a valid VLAN ID that is not already in use.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Assign a different VLAN ID to the VLAN.
   - Assign a different VLAN ID to the default VLAN.

     ```
     device(config)# default-vlan-id 4095
     ```

     Reassigning a different VLAN ID to the default VLAN does not change the properties of the default VLAN, but allows you to use the VLAN ID "1" as a configurable VLAN.
   - Assign different VLAN IDs to reserved VLANs 4090, 4091 and 4092.

     ```
     device(config)# reserved-vlan-map vlan 4091 new-vlan 10
     device(config)# write memory
     device(config)# exit
     device# reload
     ```

     NOTE
     You must save and reload the configuration.

     In the example, the configuration changes the VLAN ID of 4091 to 10.

The following example shows how to assign VLAN ID to the default VLAN.

```
device# configure terminal
device(config)# default-vlan-id 4095
```

The following example shows how to assign VLAN ID to reserved VLANs.

```
device(config)# reserved-vlan-map vlan 4091 new-vlan 10
device(config)# write memory
device(config)# exit
device# reload
```

Viewing Reassigned VLAN IDs for Reserved VLANs

To view the assigned VLAN IDs for reserved VLANs 4090, 4091 and 4092, use the **show reserved-vlan-map** command. The reassigned VLAN IDs also display in the output of the **show running-config** and **show config** commands.

The following shows example output for the **show reserved-vlan-map** command.

```
device# show reserved-vlan-map
Reserved Purpose           Default   Re-assign   Current
 Tunnel VLAN               4090       4090         4090
 CPU VLAN                  4091       4091         4091
 All Ports VLAN            4092       4092         4092
```

The following table defines the fields in the output of the **show reserved-vlan-map** command.

TABLE 19 Output of the show reserved-vlan-map Command

Field	Description
Reserved Purpose	Describes for what the VLAN is reserved. Note that the description is for RUCKUS internal VLAN management.
Default	The default VLAN ID of the reserved VLAN.
Re-assign	The VLAN ID to which the reserved VLAN was reassigned.[4]
Current	The current VLAN ID for the reserved VLAN.[4]

[4] If you reassign a reserved VLAN without saving the configuration and reloading the software, the reassigned VLAN ID will display in the Re-assign column. However, the previously configured or default VLAN ID will display in the Current column until the configuration is saved and the device reloaded.

Assigning LAG Ports

When a LAG interface is assigned to a VLAN, all member ports of the LAG are automatically added to that VLAN.

Enable Spanning Tree on a VLAN

The spanning tree bridge and port parameters are configurable using one CLI command set at the Global Configuration Level of each Port-based VLAN. Suppose you want to enable the IEEE 802.1D STP across VLAN 3. To do so, use the following method.

NOTE
When port-based VLANs are not operating on the system, STP is set on a system-wide level at the global CONFIG level of the CLI.

1. Access the global CONFIG level of the CLI on Device-A by entering the following commands.

```
device-A# enableNo password has been assigned yet...device-A# configure terminaldevice-A(config)#
```

2. Access the level of the CLI for configuring port-based VLAN 3 by entering the following command.

```
device-A(config)# device-A(config)# vlan 3device-A(config-vlan-3)#
```

3. From VLAN 3 configuration level of the CLI, enter the following command to enable STP on all tagged and untagged ports associated with VLAN 3.

```
device-B(config-vlan-3)# device-B(config-vlan-3)# spanning-treedevice-B(config-vlan-3)#
```

4. Enter the following commands to exit the VLAN configuration mode and save the configuration to the system-config file on flash memory.

```
device-B(config-vlan-3)# device-B(config-vlan-3)# enddevice-B# write memorydevice-B#
```

5. Repeat steps 1 - 4 on Device-B.

NOTE
You do not need to configure values for the STP parameters. All parameters have default values as noted below. Additionally, all values will be globally applied to all ports on the system or on the port-based VLAN for which they are defined.

To configure a specific **path-cost** or **priority** value for a given port, enter those values using the key words in the brackets [ ] shown in the syntax summary below. If you do not want to specify values for any given port, this portion of the command is not required.

To configure STP bridge and port parameters, refer to #unique_323.

Enabling Port-Based VLANs and Tagging a Port to the VLAN

You can enable port-based VLAN at the global configuration mode and can assign 802.1Q tagging to a port that belongs to a VLAN.

You can configure up to 4000 port-based VLANs depending on the device support. Each port-based VLAN can contain either tagged or untagged ports. A port cannot be a member of more than one port-based VLAN unless the port is tagged.

1. Enter the global configuration mode.

```
device# configure terminal
```

2. Configure a port-based VLAN using the **name** and **by port** keywords.

```
device(config)# vlan 222 name Mktg by port
```

Port-based VLAN 222 by the name "Mktg" is created.

3. Tag a port to the VLAN.

```
device(config-vlan-222)# tagged ethernet 1/1/5
```

NOTE
Tagging does not apply to the default VLAN. The ports are defined as either tagged or untagged at the VLAN level.

The following example shows how to configure a port-based VLAN and tag a port to the VLAN.

```
device# configure terminal
device(config)# vlan 222 name Mktg by port
device(config-vlan-222)# tagged ethernet 1/1/5
```

VLAN-Based Static MAC Entries Configuration

You can configure a VLAN to drop packets that have a particular source or destination MAC address.

You can configure a maximum of 2048 static MAC address drop entries on a RUCKUS device.

Use the CLI command **show running-config** to view the static MAC address drop entries currently configured on the device.

Configuring a VLAN to Drop Static MAC Entries

To configure a VLAN to drop packets with a source or destination MAC address of 0000.0063.67FF, enter the following commands.

```
device(config)# vlan 2
device(config-vlan-2)# static-mac-address 0000.0063.67FF drop
```

IP Subnet Address on Multiple Port-Based VLAN Configuration

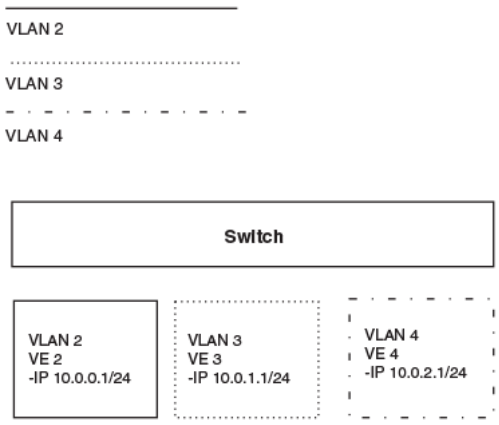
For a RUCKUS device to route between port-based VLANs, you must add a virtual routing interface to each VLAN. Generally, you also configure a unique IP subnet address on each virtual routing interface. For example, if you have three port-based VLANs, you add a virtual routing interface to each VLAN, then add a separate IP subnet address to each virtual routing interface. The IP address on each of the virtual routing interfaces must be in a separate subnet. The RUCKUS device routes Layer 3 traffic between the subnets using the subnet addresses.

NOTE
This feature applies only to Layer 3 switches.

NOTE
Before using the method described in this section, refer to VLAN Groups and Virtual Routing Interface Group on page 286. You might be able to achieve the results you want using the methods in that section instead.

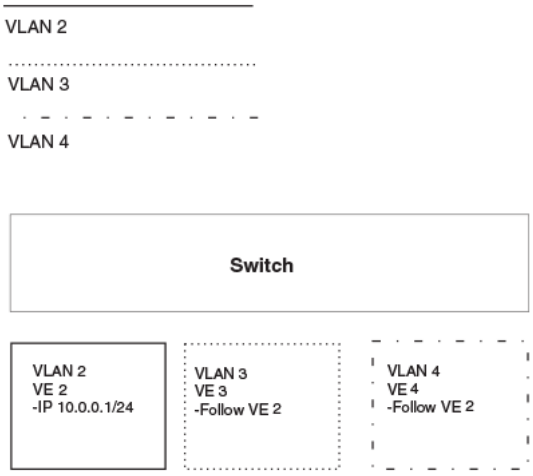
The following figure shows an example of this type of configuration.

FIGURE 90 Multiple Port-Based VLANs with Separate Protocol Addresses



As shown in this example, each VLAN has a separate IP subnet address. If you need to conserve IP subnet addresses, you can configure multiple VLANs with the same IP subnet address, as shown in the following figure.

FIGURE 91 Multiple Port-Based VLANs with the Same Protocol Address



Each VLAN still requires a separate virtual routing interface. However, all three VLANs now use the same IP subnet address.

In addition to conserving IP subnet addresses, this feature allows containment of Layer 2 broadcasts to segments within an IP subnet. For ISP environments where the same IP subnet is allocated to different customers, placing each customer in a separate VLAN allows all customers to share the IP subnet address, while at the same time isolating them from one another Layer 2 broadcasts.

> **NOTE**
> You can provide redundancy to an IP subnet address that contains multiple VLANs using a pair of RUCKUS Layer 3 Switches configured for RUCKUS VRRP (Virtual Router Redundancy Protocol).

The RUCKUS device performs proxy Address Resolution Protocol (ARP) for hosts that want to send IP traffic to hosts in other VLANs that are sharing the same IP subnet address. If the source and destination hosts are in the same VLAN, the RUCKUS device does not need to use ARP:

- If a host attached to one VLAN sends an ARP message for the MAC address of a host in one of the other VLANs using the same IP subnet address, the RUCKUS device performs a proxy ARP on behalf of the other host. The RUCKUS device then replies to the ARP by sending the

virtual routing interface MAC address. The RUCKUS device uses the same MAC address for all virtual routing interfaces. When the host that sent the ARP then sends a unicast packet addressed to the virtual routing interface MAC address, the device switches the packet on Layer 3 to the destination host on the VLAN.

NOTE
If the RUCKUS device ARP table does not contain the requested host, the RUCKUS device forwards the ARP request on Layer 2 to the same VLAN as the one that received the ARP request. Then the device sends an ARP for the destination to the other VLANs that are using the same IP subnet address.

- If the destination is in the same VLAN as the source, the RUCKUS device does not need to perform a proxy ARP.

To configure multiple VLANs to use the same IP subnet address:

- Configure each VLAN, including adding tagged or untagged ports.
- Configure a separate virtual routing interface for each VLAN, but do not add an IP subnet address to more than one of the virtual routing interfaces.
- Configure the virtual routing interfaces that do not have the IP subnet address to "follow" the virtual routing interface that does have the address.

To configure the VLANs shown in Figure 91, you could enter the following commands.

```
device(config)# vlan 1 by port
device(config-vlan-1)# untagged ethernet 1/1/1
device(config-vlan-1)# tagged ethernet 1/1/8
device(config-vlan-1)# interface ve 1
```

The commands above configure port-based VLAN 1. The VLAN has one untagged port (1/1/1) and a tagged port (1/1/8). In this example, all three VLANs contain port 1/1/8 so the port must be tagged to allow the port to be in multiple VLANs. You can configure VLANs to share a Layer 3 protocol interface regardless of tagging. A combination of tagged and untagged ports is shown in this example to demonstrate that sharing the interface does not change other VLAN features.

Notice that each VLAN still requires a unique virtual routing interface.

The following commands configure port-based VLANs 2 and 3.

```
device(config-vlan-1)# vlan 2 by port
device(config-vlan-2)# untagged ethernet 1/1/2
device(config-vlan-2)# tagged ethernet 1/1/8
device(config-vlan-2)# interface ve 2
device(config-vlan-2)# vlan 3 by port
device(config-vlan-3)# untagged ethernet 1/1/5 to 1/1/6
device(config-vlan-3)# tagged ethernet 1/1/8
device(config-vlan-3)# interface ve 3
```

The following commands configure an IP subnet address on virtual routing interface 1.

```
device(config-vlan-3)# interface ve 1
device(config-vif-1)# ip address 10.0.0.1/24
```

The following commands configure virtual routing interfaces 2 and 3 to "follow" the IP subnet address configured on virtual routing interface 1.

```
device(config-vif-1)# interface ve 2
device(config-vif-2)# ip follow ve 1
device(config-vif-2)# interface ve 3
device(config-vif-3)# ip follow ve 1
```

VLAN Groups and Virtual Routing Interface Group

To simplify configuration when you have many VLANs with the same configuration, you can configure VLAN groups and virtual routing interface groups.

> **NOTE**
> VLAN groups are supported on Layer 3 Switches and Layer 2 Switches. Virtual routing interface groups are supported only on Layer 3 Switches.

When you create a VLAN group, the VLAN parameters you configure for the group apply to all the VLANs within the group. Additionally, you can easily associate the same IP subnet interface with all the VLANs in a group by configuring a virtual routing interface group with the same ID as the VLAN group.

- The VLAN group feature allows you to create multiple port-based VLANs with identical port members. Because the member ports are shared by all the VLANs within the group, you must add the ports as tagged ports. This feature not only simplifies VLAN configuration but also allows you to have a large number of identically configured VLANs in a startup-config file on the device flash memory module. Normally, a startup-config file with a large number of VLANs might not fit on the flash memory module. By grouping the identically configured VLANs, you can conserve space in the startup-config file so that it fits on the flash memory module.
- The virtual routing interface group feature is useful when you want to configure the same IP subnet address on all the port-based VLANs within a VLAN group. You can configure a virtual routing interface group only after you configure a VLAN group with the same ID. The virtual routing interface group automatically applies to the VLANs in the VLAN group that has the same ID and cannot be applied to other VLAN groups or to individual VLANs.

You can create up to 32 VLAN groups and 32 virtual routing interface groups. A virtual routing interface group always applies only to the VLANs in the VLAN group with the same ID.

> **NOTE**
> Depending on the size of the VLAN ID range you want to use for the VLAN group, you might need to allocate additional memory for VLANs. On Layer 3 Switches, if you allocate additional memory for VLANs, you also need to allocate the same amount of memory for virtual routing interfaces. This is true regardless of whether you use the virtual routing interface groups. To allocate additional memory, refer to Allocating Memory for More VLANs, More Associated ports, or More Virtual Routing Interfaces on page 288.

Configuring a VLAN Group

You can configure a VLAN group and add or remove individual VLANs or a range of VLANs from the VLAN group.

Beginning with FastIron 08.0.80, configuring dual-mode with untagged VLAN as VLAN group member VLAN on VLAN group member interfaces is not allowed. You can configure the VLAN group member interface as untagged in any other VLAN, but not in VLAN group member VLANs.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Configure a VLAN group.

```
device(config)# vlan-group 1 vlan 2 to 257
```

   In the example, VLAN group 1 is configured and VLANs from 2 through 257 are assigned to the VLAN group.

3. Add individual VLANs or a range of VLANs.

```
device(config)# vlan-group 2 vlan 1000 to 1250
device(config-vlan-group-2)# add-vlan 1251 to 1500
device(config-vlan-group-2)# add-vlan 1501 to 1750
```

   To add more than 256 VLANs, use the **add-vlan** command from the interface configuration mode.

4. (Optional) Remove individual VLANs or a range of VLANs.

```
device(config-vlan-group-2)# remove-vlan 900 to 1000
```

   To remove more than 256 VLANs, use the **remove-vlan** command from the interface configuration mode.

5. Add ports to the VLAN group.

```
device(config-vlan-group-2)# tagged 1/1/1 to 1/1/2
```

   Ports 1/1/1 and 1/1/2 are added as tagged ports for the VLAN group.

The following example shows how to configure a VLAN group.

```
device# configure terminal
device(config)#vlan-group 2 vlan 1000 to 1250
device(config-vlan-group-2)#add-vlan 1251 to 1500
device(config-vlan-group-2)# remove-vlan 900 to 1000
device(config-vlan-group-2)# tagged 1/1/1 to 1/1/2
```

Displaying Information About VLAN Groups

To display VLAN group configuration information, use the **show vlan-group** command.

```
device# show vlan-group
vlan-group 1 vlan 2 to 20
 tagged ethernet 1/1/1 to 1/1/2
!
vlan-group 2 vlan 21 to 40
 tagged ethernet 1/1/1 to 1/1/2
!
```

The *group-id* parameter is not specified, the configuration information for all the configured VLAN groups is displayed.

Configuring a Virtual Routing Interface Group

You can configure a virtual routing interface group using the **group-router-interface** command.

A virtual routing interface group allows you to associate the same IP subnet interface with multiple port-based VLANs. For example, if you associate a virtual routing interface group with a VLAN group, all the VLANs in the group have the IP interface of the virtual routing interface group.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Configure a VLAN group.

```
device(config)# vlan-group 1
```

   The VLAN group must be preconfigured and enabled to use a virtual routing interface group.

3. Enable VLAN group to use a virtual routing interface group.

```
device(config-vlan-group-1)# group-router-interface
```

   The **group-router-interface** command enables a VLAN group to use a virtual routing interface group. The software automatically associates a virtual routing interface group only with the VLAN group that has the same ID.

4. Exit VLAN configuration mode.

```
device(config-vlan-group-1)# exit
```

5. Associate the virtual interface routing group with a VLAN group from the interface configuration mode.

```
device(config)# interface group-ve 1
```

NOTE
IPv6 is not supported with **group-ve**.

NOTE
FastIron devices support group-ve with OSPF, VRRP v2, and VRRP-E v2 protocols only

The following example shows how to configure a virtual routing interface group.

```
device# configure terminal
device(config)# vlan-group 1
device(config-vlan-group-1)# group-router-interface
device(config-vlan-group-1)# exit
device(config)# interface group-ve 1
```

Configuration Notes and Feature Limitations for Virtual Routing Interface Group

- When you configure a virtual routing interface group, all members of the group have the same IP subnet address. This feature is useful in collocation environments where the device has many IP addresses and you want to conserve the IP address space.
- The **group-router-interface** command creates router interfaces for each VLAN in the VLAN group by using the VLAN IDs of each of the VLANs as the corresponding virtual interface number. Therefore, if a VLAN group contains VLAN IDs greater than the maximum virtual interface number allowed, the **group-router-interface** command will be rejected.

Displaying the VLAN Group and Virtual Routing Interface Group Information

To verify configuration of VLAN groups and virtual routing interface groups, display the running-config file. If you have saved the configuration to the startup-config file, you also can verify the configuration by displaying the startup-config file. The following example shows the running-config information for the VLAN group and virtual routing interface group configured in the previous examples. The information appears in the same way in the startup-config file.

```
device# show running-config
lines not related to the VLAN group omitted...
vlan-group 1 vlan 2 to 20
 add-vlan 1001 to 1002
 tagged ethe 1/1/1 to 1/1/2
 router-interface-group
lines not related to the virtual routing interface group omitted...
interface group-ve 1
 ip address 10.10.10.1 255.255.255.0
```

NOTE
If you have enabled display of subnet masks in CIDR notation, the IP address information is shown as follows: 10.10.10.1/24.

Allocating Memory for More VLANs, More Associated ports, or More Virtual Routing Interfaces

Ruckus Layer 2 and Layer 3 Switches support up to 4095 VLANs. In addition, Layer 3 switches can support up to 512 virtual routing interfaces.

The number of VLANs, associated ports, and virtual routing interfaces supported on your product depends on the device. The following table lists the default and configurable maximum numbers of VLANs and virtual routing interfaces for Layer 2 and Layer 3 switches. Unless otherwise noted, the values apply to both types of switches.

NOTE
A minimum value of 5 must be configured when configuring **system-max vlan**.

TABLE 20 VLAN, VPORT, and Virtual Routing Interface Support

VLANs		VPORTs		Virtual routing interfaces	
Default maximum	Configurable maximum	Default maximum	Configurable maximum	Default maximum	Configurable maximum
1024	4094	8192	524032	255	512

NOTE
If many of your VLANs will have an identical configuration, you might want to configure VLAN groups and virtual routing interface groups after you increase the system capacity for VLANs and virtual routing interfaces. Refer to VLAN Groups and Virtual Routing Interface Group on page 286.

Increasing the Number of Configurable VLANs or Port-to-VLAN Associations or Virtual Routing Interfaces

You can increase the maximum number of VLANs or port-to-VLAN associations or virtual routing interfaces that can be configured.

You can specify up to 4095 VLANs, however, you can configure only 4094 VLANs. VLAN ID 4094 is reserved for use by the Single Spanning Tree feature.

VPORT entries are created in the software database for each port association to a VLAN. The default VPORT limit is 64. By default, this limits port-to-VLAN associations to 8,192 (Max VLANs * MAX ports per VLAN, or 64 * 128).

1. Enter global configuration mode.

```
device# configure terminal
```

2. Increase the maximum number of VLANs or port-to-VLAN associations.

```
device (config)# system-max vlan 4094
device(config)# write memory
device(config)# end
device# reload
```

In the example, the available port-to-VLAN associations is increased to more than 524,000 entries. The **num** parameter indicates the maximum number of VLANs. The range of valid values depends on the device you are configuring. Refer to Table 20 on page 289.

NOTE
The **system-max** default VLAN is increased from 64 through 1024. Hence the default VPORT limit is no more equal to the default MAX VLAN setting.

3. Increase the maximum number of virtual routing interfaces.

```
device(config)# system-max virtual-interface 1024
device(config)# write memory
device(config)# end
device# reload
```

The **num** parameter indicates the maximum number of virtual routing interfaces. The range of valid values depends on the device you are configuring. Refer to Table 20 on page 289.

Topology Groups

A topology group is a named set of VLANs that share a Layer 2 topology. Topology groups simplify configuration and enhance scalability of Layer 2 protocols by allowing you to run a single instance of a Layer 2 protocol on multiple VLANs.

You can use topology groups with the following Layer 2 protocols:

- STP/RSTP
- MRP
- VSRP
- 802.1W

Topology groups simplify Layer 2 configuration and provide scalability by enabling you to use the same instance of a Layer 2 protocol for multiple VLANs. For example, if a RUCKUS device is deployed in a Metro network and provides forwarding for two MRP rings that each contain 128 VLANs, you can configure a topology group for each ring. If a link failure in a ring causes a topology change, the change is applied to all the VLANs in the ring topology group. Without topology groups, you would need to configure a separate ring for each VLAN.

Master VLAN and Member VLANs

Each topology group contains a master VLAN and can contain one or more member VLANs and VLAN groups:

- Master VLAN - The master VLAN contains the configuration information for the Layer 2 protocol. For example, if you plan to use the topology group for MRP, the topology group master VLAN contains the ring configuration information.
- Member VLANs - The member VLANs are additional VLANs that share ports with the master VLAN. The Layer 2 protocol settings for the ports in the master VLAN apply to the same ports in the member VLANs. A change to the master VLAN Layer 2 protocol configuration or Layer 2 topology affects all the member VLANs. Member VLANs do not independently run a Layer 2 protocol.
- Member VLAN groups - A VLAN group is a named set of VLANs. The VLANs within a VLAN group have the same ports and use the same values for other VLAN parameters.

When a Layer 2 topology change occurs on a port in the master VLAN, the same change is applied to that port in all the member VLANs that contain the port. For example, if you configure a topology group for which the master VLAN contains ports 1/1/1 and 1/1/2, a Layer 2 state change on port 1/1/1 applies to port 1/1/1 in all the member VLANs that contain that port. However, the state change does not affect port 1/1/1 in VLANs that are not members of the topology group.

Control Ports and Free Ports

A port that is in a topology group can be a control port or a free port:

- Control port - A control port is a port in the master VLAN, and is therefore controlled by the Layer 2 protocol configured in the master VLAN. The same port in all the member VLANs is controlled by the master VLAN Layer 2 protocol. Each member VLAN must contain all of the control ports and can contain additional ports.
- Free port - A free port is not controlled by the master VLAN Layer 2 protocol. The master VLAN can contain free ports. (In this case, the Layer 2 protocol is disabled on those ports.) In addition, any ports in the member VLANs that are not also in the master VLAN are free ports.

  NOTE
  Since free ports are not controlled by the master port Layer 2 protocol, they are assumed to always be in the Forwarding state.

Topology Group Configuration Considerations

- You must configure the master VLAN and member VLANs or member VLAN groups before you configure the topology group.
- You can configure up to 30 topology groups. Each group can control up to 4096 VLANs. A VLAN cannot be controlled by more than one topology group.
- The topology group must contain a master VLAN and can also contain individual member VLANs, VLAN groups, or a combination of individual member VLANs and VLAN groups.

- If you add a new master VLAN to a topology group that already has a master VLAN, the new master VLAN replaces the older master VLAN. All member VLANs and VLAN groups follow the Layer 2 protocol settings of the new master VLAN.
- If you remove the master VLAN (by entering **no master-vlan** *vlan-id*), the software selects the new master VLAN from member VLANs. A new candidate master VLAN will be in configured order to a member VLAN so that the first added member VLAN will be a new candidate master VLAN. Once you save and reload, a member-vlan with the youngest VLAN ID will be the new candidate master. The new master VLAN inherits the Layer 2 protocol settings of the older master VLAN.
- The topology group will be deleted if the master is deleted and there are no member VLANs. This is true even if the topology group has member-groups.
- Once you add a VLAN as a member of a topology group, all the Layer 2 protocol information on the VLAN is deleted.
- A default VLAN cannot be a member of a topology group.
- MRP master node has to be un-configured (**no master-vlan** command) prior to changing the master VLAN of a topology group where this MRP instance is part of. This action prevents MRP BPDU hardware flooding which can result in MRP continuous state flap.

Configuring a Topology Group

You can configure a topology group with a master VLAN. You can add member VLANs and VLAN groups to the master VLAN.

A topology group can have only one master VLAN. The master VLAN must be already configured before it is added to the topology group.

Once you add a VLAN or VLAN group as a member of a topology group, all the Layer 2 protocol configuration information for the VLAN or VLAN group is deleted and uses the Layer 2 protocol settings of the master VLAN instead.

If you remove a member VLAN or VLAN group from a topology group, you will need to reconfigure the Layer 2 protocol information in the VLAN or VLAN group.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Configure a topology group.

```
device(config)# topology-group 2
```

3. Add a master VLAN to the topology group.

```
device(config-topo-group-2)# master-vlan 2
```

4. Add a member VLAN to the topology group.

```
device(config-topo-group-2)# member-vlan 3
```

5. Add a member VLAN group to the topology group.

```
device(config-topo-group-2)# member-group 2
```

The following example shows how to configure a topology group and add member VLANs and VLAN groups to the topology group.

```
device# configure terminal
device(config)# topology-group 2
device(config-topo-group-2)# master-vlan 2
device(config-topo-group-2)# member-vlan 3
device(config-topo-group-2)# member-group 2
```

Displaying STP Information

To display STP information for a VLAN, enter a command such as the following.

```
device# show span vlan 4
VLAN 4 BPDU cam_index is 14344 and the Master DMA Are(HEX) 18 1A
STP instance owned by VLAN 2
```

This example shows STP information for VLAN 4. The line shown in bold type indicates that the VLAN STP configuration is controlled by VLAN 2. This information indicates that VLAN 4 is a member of a topology group and VLAN 2 is the master VLAN in that topology group.

Displaying Topology Group Information

To display topology group information, enter the following command.

```
device# show topology-group
Topology Group 3
================
 master-vlan 2
 member-vlan none
 Common control ports           L2 protocol
 ethernet 1/1/1                 MRP
 ethernet 1/1/2                 MRP
 ethernet 1/1/5                 VSRP
 ethernet 1/2/22                VSRP
 Per vlan free ports
 ethernet 1/2/3                 Vlan 2
 ethernet 1/2/4                 Vlan 2
 ethernet 1/2/11                Vlan 2
 ethernet 1/2/12                Vlan 2
```

This display shows the following information.

TABLE 21 CLI Display of Topology Group Information

Field	Description
master-vlan	The master VLAN for the topology group. The settings for STP, MRP, or VSRP on the control ports in the master VLAN apply to all control ports in the member VLANs within the topology group.
member-vlan	The member VLANs in the topology group.
Common control ports	The master VLAN ports that are configured with Layer 2 protocol information. The Layer 2 protocol configuration and state of these ports in the master VLAN applies to the same port numbers in all the member VLANs.
Layer 2 (L2) protocol	The Layer 2 protocol configured on the control ports. The Layer 2 protocol can be one of the following: • MRP • STP • VSRP
Per VLAN free ports	The ports that are not controlled by the Layer 2 protocol information in the master VLAN.

Super-Aggregated VLAN Configuration

You can aggregate multiple VLANs within another VLAN. This feature allows you to construct Layer 2 paths and channels. This feature is particularly useful for Virtual Private Network (VPN) applications in which you need to provide a private, dedicated Ethernet connection for an individual client to transparently reach its subnet across multiple networks.

Conceptually, the paths and channels are similar to Asynchronous Transfer Mode (ATM) paths and channels. A path contains multiple channels, each of which is a dedicated circuit between two end points. The two devices at the end points of the channel appear to each other to be directly attached. The network that connects them is transparent to the two devices.

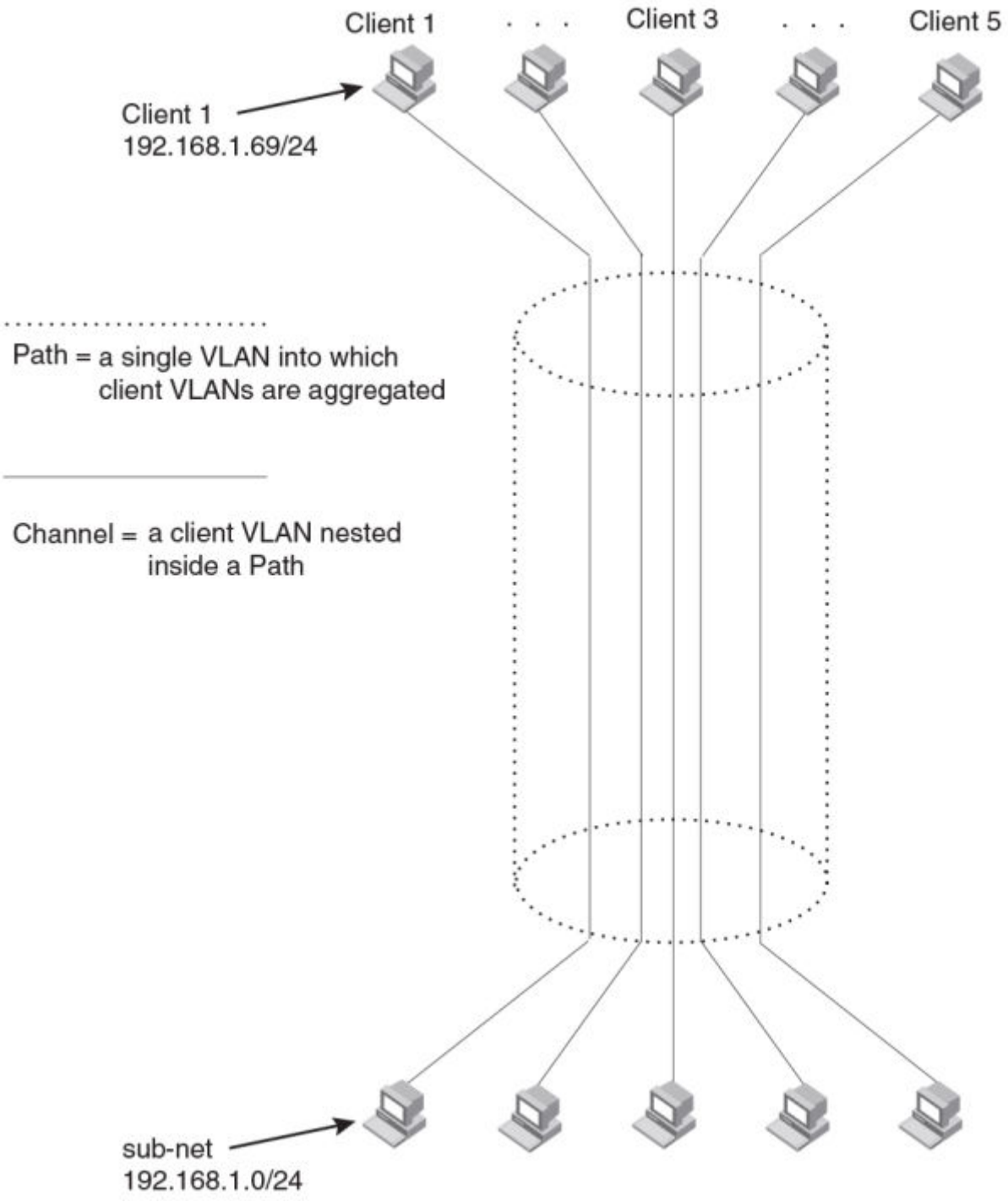
You can aggregate up to 4094 VLANs within another VLAN. This provides a total VLAN capacity on one RUCKUS device of 16,760,836 channels (4094 * 4094).

The devices connected through the channel are not visible to devices in other channels. Therefore, each client has a private link to the other side of the channel.

The feature allows point-to-point and point-to-multipoint connections.

The following figure shows a conceptual picture of the service that aggregated VLANs provide. Aggregated VLANs provide a path for multiple client channels. The channels do not receive traffic from other channels. Thus, each channel is a private link.

FIGURE 92 Conceptual Model of the Super Aggregated VLAN Application



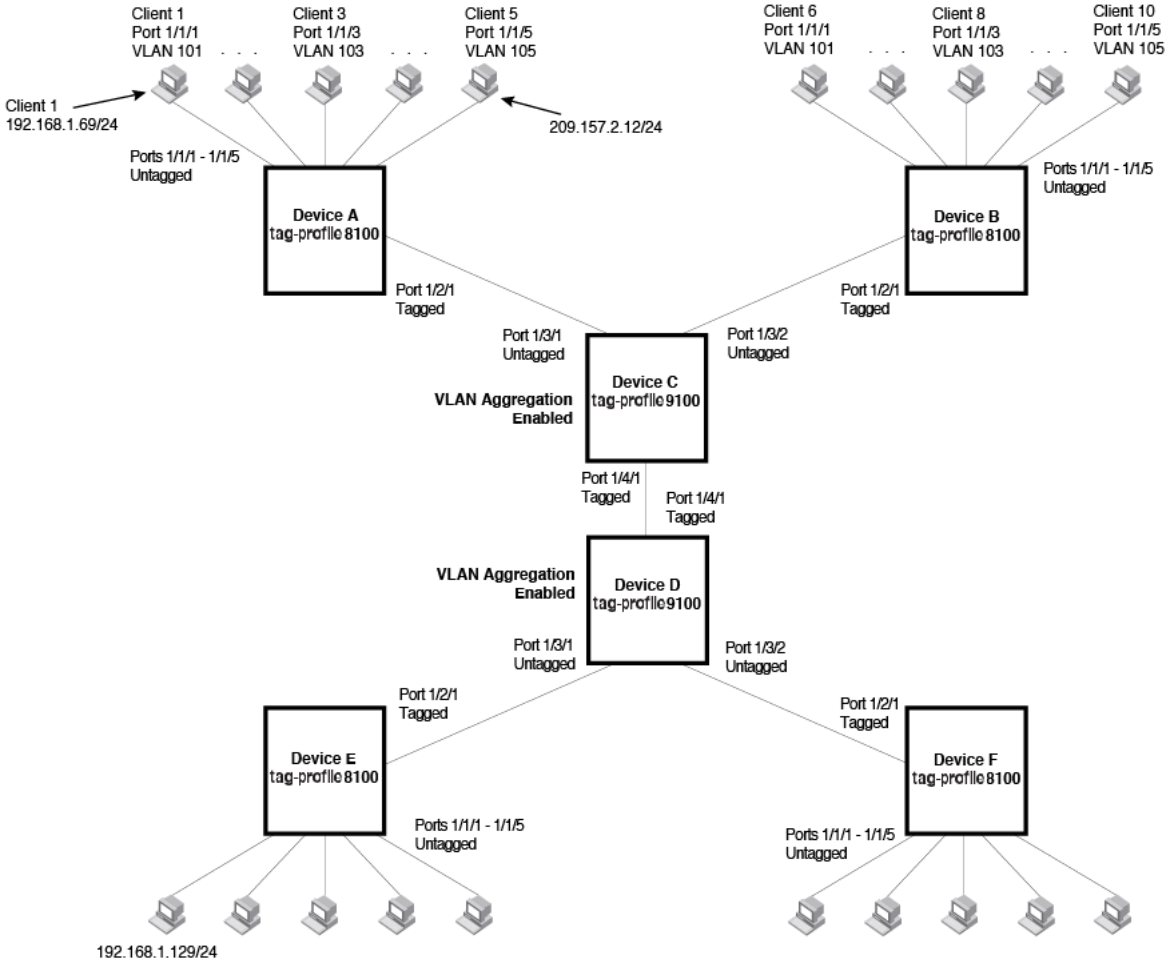
Each client connected to the edge device is in its own port-based VLAN, which is like an ATM channel. All the clients' VLANs are aggregated by the edge device into a single VLAN for connection to the core. The single VLAN that aggregates the clients' VLANs is like an ATM path.

The device that aggregates the VLANs forwards the aggregated VLAN traffic through the core. The core can consist of multiple devices that forward the aggregated VLAN traffic. The edge device at the other end of the core separates the aggregated VLANs into the individual client VLANs before

forwarding the traffic. The edge devices forward the individual client traffic to the clients. For the clients' perspective, the channel is a direct point-to-point link.

The following figure shows an example application that uses aggregated VLANs. This configuration includes the client connections shown in Figure 92.

FIGURE 93 Example of a Super Aggregated VLAN Application



In this example, a collocation service provides private channels for multiple clients. Although the same devices are used for all the clients, the VLANs ensure that each client receives its own Layer 2 broadcast domain, separate from the broadcast domains of other clients. For example, client 1 cannot ping client 5.

The clients at each end of a channel appear to each other to be directly connected and thus can be on the same subnet and use network services that require connection to the same subnet. In this example, client 1 is in subnet 192.168.1.0/24 and so is the device at the other end of client 1 channel.

Because each VLAN configured on the core devices is an aggregate of multiple client VLANs, the aggregated VLANs greatly increase the number of clients a core device can accommodate.

This example shows a single link between the core devices. However, you can use a trunk group to add link-level redundancy.

Configuration Notes for Aggregated VLANs

- Super aggregated VLANs and VSRP are not supported together on the same device.
- Super aggregated VLANs and Q-in-Q are supported using the tag-profile command.

Configuring Aggregated VLANs

To configure aggregated VLANs, perform the following tasks:

- On each edge device, configure a separate port-based VLAN for each client connected to the edge device. In each client VLAN:
  - Add the port connected to the client as an untagged port.
  - Add the port connected to the core device (the device that will aggregate the VLANs) as a tagged port. This port must be tagged because all the client VLANs share the port as an uplink to the core device.
- On each core device:
  - Enable VLAN aggregation. This support allows the core device to add an additional tag to each Ethernet frame that contains a VLAN packet from the edge device. The additional tag identifies the aggregate VLAN (the path). However, the additional tag can cause the frame to be longer than the maximum supported frame size. The larger frame support allows Ethernet frames up to 1530 bytes long.
  - To allow frames larger than 1522, you must enable jumbo frames. To globally enable jumbo support , enter commands such as the following.

```
device(config)# jumbo
device(config)# write memory
device(config)# end
device# reload
```

NOTE
Enable the VLAN aggregation option only on the core devices.

- Configure a VLAN tag profile (tag ID) that is different than the tag profile used on the edge devices. If you use the default tag profile (8100) on the edge devices, set the tag profile on the core devices to another value, such as 9100. The tag profile must be the same on all the core devices. The edge devices also must have the same tag profile but the profile must be different from the tag profile on the core devices.

NOTE
You can enable the Spanning Tree Protocol (STP) on the edge devices or the core devices, but not both. If you enable STP on the edge devices and the core devices, STP will prevent client traffic from travelling through the core to the other side.

Configuring Aggregated VLANs on an Edge Device

To configure the aggregated VLANs on device A in Figure 93 on page 295, enter the following commands.

```
device(config)# vlan 101 by port
device(config-vlan-101)# tagged ethernet 1/2/1
device(config-vlan-101)# untagged ethernet 1/1/1
device(config-vlan-101)# exit
device(config)# vlan 102 by port
device(config-vlan-102)# tagged ethernet 1/2/1
device(config-vlan-102)# untagged ethernet 1/1/2
device(config-vlan-102)# exit
device(config)# vlan 103 by port
device(config-vlan-103)# tagged ethernet 1/2/1
device(config-vlan-103)# untagged ethernet 1/1/3
device(config-vlan-103)# exit
device(config)# vlan 104 by port
device(config-vlan-104)# tagged ethernet 1/2/1
device(config-vlan-104)# untagged ethernet 1/1/4
```

```
device(config-vlan-104)# exit
device(config)# vlan 105 by port
device(config-vlan-105)# tagged ethernet 1/2/1
device(config-vlan-105)# untagged ethernet 1/1/5
device(config-vlan-105)# exit
device(config)# write memory
```

Use the **tagged** command to add the port that the device uses for the uplink to the core device. Use the **untagged** command to add the ports connected to the individual clients.

Configuring Aggregated VLANs on a Core Device

To configure the aggregated VLANs on device C in Figure 93 on page 295, enter the following commands.

```
device(config)# tag-profile 9100
device(config)# aggregated-vlan
device(config)# vlan 101 by port
device(config-vlan-101)# tagged ethernet 1/4/1
device(config-vlan-101)# untagged ethernet 1/3/1
device(config-vlan-101)# exit
device(config)# vlan 102 by port
device(config-vlan-102)# tagged ethernet 1/4/1
device(config-vlan-102)# untagged ethernet 1/3/2
device(config-vlan-102)# exit
device(config)# write memory
```

Verifying the Aggregated VLAN Configuration

You can verify the VLAN, VLAN aggregation option, and tag configuration by viewing the running-config. To display the running-config, enter the **show running-config** command from any CLI prompt. After you save the configuration changes to the startup-config, you also can display the settings in that file by entering the **show configuration** command from any CLI prompt.

Complete CLI Examples for Aggregated VLANs

The following sections show all the Aggregated VLAN configuration commands on the devices in Figure 93 on page 295.

> **NOTE**
> In these examples, the configurations of the edge devices (A, B, E, and F) are identical. The configurations of the core devices (C and D) also are identical. The aggregated VLAN configurations of the edge and core devices on one side must be symmetrical (in fact, a mirror image) to the configurations of the devices on the other side. For simplicity, the example in Figure 93 on page 295 is symmetrical in terms of the port numbers. This allows the configurations for both sides of the link to be the same. If your configuration does not use symmetrically arranged port numbers, the configurations should not be identical but must use the correct port numbers.

Commands for Configuring Aggregated VLANs on Device A

```
deviceA(config)# vlan 101 by port
deviceA(config-vlan-101)# tagged ethernet 1/2/1
deviceA(config-vlan-101)# untagged ethernet 1/1/1
deviceA(config-vlan-101)# exit
deviceA(config)# vlan 102 by port
deviceA(config-vlan-102)# tagged ethernet 1/2/1
deviceA(config-vlan-102)# untagged ethernet 1/1/2
deviceA(config-vlan-102)# exit
deviceA(config)# vlan 103 by port
deviceA(config-vlan-103)# tagged ethernet 1/2/1
deviceA(config-vlan-103)# untagged ethernet 1/1/3
deviceA(config-vlan-103)# exit
deviceA(config)# vlan 104 by port
deviceA(config-vlan-104)# tagged ethernet 1/2/1
```

```
deviceA(config-vlan-104)# untagged ethernet 1/1/4
deviceA(config-vlan-104)# exit
deviceA(config)# vlan 105 by port
deviceA(config-vlan-105)# tagged ethernet 1/2/1
deviceA(config-vlan-105)# untagged ethernet 1/1/5
deviceA(config-vlan-105)# exit
device(config)# write memory
```

Commands for Configuring Aggregated VLANs on Device B

The commands for configuring device B are identical to the commands for configuring device A. Notice that you can use the same channel VLAN numbers on each device. The devices that aggregate the VLANs into a path can distinguish between the identically named channel VLANs based on the ID of the path VLAN.

```
deviceB(config)# vlan 101 by port
deviceB(config-vlan-101)# tagged ethernet 1/2/1
deviceB(config-vlan-101)# untagged ethernet 1/1/1
deviceB(config-vlan-101)# exit
deviceB(config)# vlan 102 by port
deviceB(config-vlan-102)# tagged ethernet 1/2/1
deviceB(config-vlan-102)# untagged ethernet 1/1/2
deviceB(config-vlan-102)# exit
deviceB(config)# vlan 103 by port
deviceB(config-vlan-103)# tagged ethernet 1/2/1
deviceB(config-vlan-103)# untagged ethernet 1/1/3
deviceB(config-vlan-103)# exit
deviceB(config)# vlan 104 by port
deviceB(config-vlan-104)# tagged ethernet 1/2/1
deviceB(config-vlan-104)# untagged ethernet 1/1/4
deviceB(config-vlan-104)# exit
deviceB(config)# vlan 105 by port
deviceB(config-vlan-105)# tagged ethernet 1/2/1
deviceB(config-vlan-105)# untagged ethernet 1/1/5
deviceB(config-vlan-105)# exit
deviceB(config)# write memory
```

Commands for Configuring Aggregated VLANs on Device C

Because device C is aggregating channel VLANs from devices A and B into a single path, you need to change the tag profile and enable VLAN aggregation.

```
deviceC(config)# tag-profile 9100
deviceC(config)# aggregated-vlan
deviceC(config)# vlan 101 by port
deviceC(config-vlan-101)# tagged ethernet 1/4/1
deviceC(config-vlan-101)# untagged ethernet 1/3/1
deviceC(config-vlan-101)# exit
deviceC(config)# vlan 102 by port
deviceC(config-vlan-102)# tagged ethernet 1/4/1
deviceC(config-vlan-102)# untagged ethernet 1/3/2
deviceC(config-vlan-102)# exit
deviceC(config)# write memory
```

Commands for Configuring Aggregated VLANs on Device D

Device D is at the other end of path and separates the channels back into individual VLANs. The tag profile must be the same as tag profile configured on the other core device (Device C). In addition, VLAN aggregation also must be enabled.

```
deviceD(config)# tag-profile 9100
deviceD(config)# aggregated-vlan
deviceD(config)# vlan 101 by port
deviceD(config-vlan-101)# tagged ethernet 1/4/1
deviceD(config-vlan-101)# untagged ethernet 1/3/1
deviceD(config-vlan-101)# exit
```

```
deviceD(config)# vlan 102 by port
deviceD(config-vlan-102)# tagged ethernet 1/4/1
deviceD(config-vlan-102)# untagged ethernet 1/3/2
deviceD(config-vlan-102)# exit
deviceD(config)# write memory
```

Commands for Configuring Aggregated VLANs on Device E

Because the configuration in Figure 93 on page 295 is symmetrical, the commands for configuring device E are identical to the commands for configuring device A.

```
deviceE(config)# vlan 101 by port
deviceE(config-vlan-101)# tagged ethernet 1/2/1
deviceE(config-vlan-101)# untagged ethernet 1/1/1
deviceE(config-vlan-101)# exit
deviceE(config)# vlan 102 by port
deviceE(config-vlan-102)# tagged ethernet 1/2/1
deviceE(config-vlan-102)# untagged ethernet 1/1/2
deviceE(config-vlan-102)# exit
deviceE(config)# vlan 103 by port
deviceE(config-vlan-103)# tagged ethernet 1/2/1
deviceE(config-vlan-103)# untagged ethernet 1/1/3
deviceE(config-vlan-103)# exit
deviceE(config)# vlan 104 by port
deviceE(config-vlan-104)# tagged ethernet 1/2/1
deviceE(config-vlan-104)# untagged ethernet 1/1/4
deviceE(config-vlan-104)# exit
deviceE(config)# vlan 105 by port
deviceE(config-vlan-105)# tagged ethernet 1/2/1
deviceE(config-vlan-105)# untagged ethernet 1/1/5
deviceE(config-vlan-105)# exit
deviceE(config)# write memory
```

Commands for Configuring Aggregated VLANs on Device F

The commands for configuring device F are identical to the commands for configuring device E. In this example, Because the port numbers on each side of the configuration in Figure 93 on page 295 are symmetrical, the configuration of device F is also identical to the configuration of device A and device B.

```
deviceF(config)# vlan 101 by port
deviceF(config-vlan-101)# tagged ethernet 1/2/1
deviceF(config-vlan-101)# untagged ethernet 1/1/1
deviceF(config-vlan-101)# exit
deviceF(config)# vlan 102 by port
deviceF(config-vlan-102)# tagged ethernet 1/2/1
deviceF(config-vlan-102)# untagged ethernet 1/1/2
deviceF(config-vlan-102)# exit
deviceF(config)# vlan 103 by port
deviceF(config-vlan-103)# tagged ethernet 1/2/1
deviceF(config-vlan-103)# untagged ethernet 1/1/3
deviceF(config-vlan-103)# exit
deviceF(config)# vlan 104 by port
deviceF(config-vlan-104)# tagged ethernet 1/2/1
deviceF(config-vlan-104)# untagged ethernet 1/1/4
deviceF(config-vlan-104)# exit
deviceF(config)# vlan 105 by port
deviceF(config-vlan-105)# tagged ethernet 1/2/1
deviceF(config-vlan-105)# untagged ethernet 1/1/5
deviceF(config-vlan-105)# exit
deviceF(config)# write memory
```

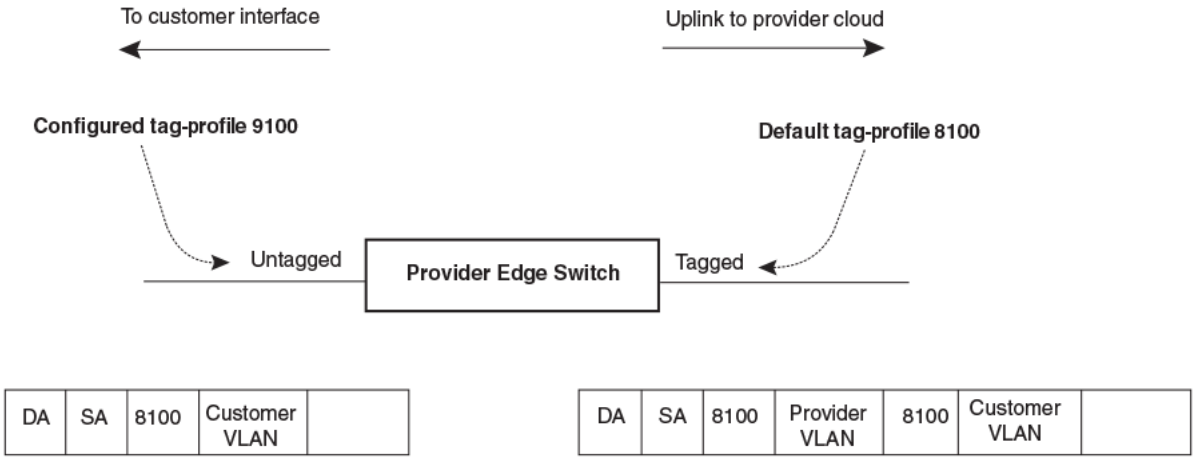
802.1ad Tagging Configuration

802.1ad tagging provides finer granularity for configuring 802.1Q tagging, enabling you to configure 802.1Q tag profile for a single port, or to direct a group of ports to a globally-configured tag profile. This feature allows you to create two identical 802.1Q tags (802.1ad tagging) on a single device. This enhancement improves SAV interoperability between RUCKUS devices and other vendors' devices that support the 802.1Q tag-profiles, but are not very flexible with the tag-profiles they accept.

> **NOTE**
> RUCKUS devices treat a double-tagged Ethernet frame as a Layer 2 only frame. The packets are not inspected for Layer 3 and Layer 4 information, and operations are not performed on the packet utilizing Layer 3 or Layer 4 information.

The following figure shows an example application with 802.1ad tagging.

FIGURE 94 802.1ad Configuration Example



In the above figure, the untagged ports (to customer interfaces) accept frames that have any 802.1Q tag other than the configured tag-profile 9100. These packets are considered untagged on this incoming port and are re-tagged when they are sent out of the uplink towards the provider. The 802.1Q tag-profile on the uplink port is 8100, so the RUCKUS device will switch the frames to the uplink device with an additional 8100 tag.

Configuration Rules for 802.1ad Tagging

- When you set the tag-profile in the global configuration mode, you need to enable tag-profile on per port basis using the **tag-profile enable** command. Only the tag-profile enabled ports use the user-configured tag-profile. The remaining ports continue using system default tag-profile. Using this method, Q-in-Q can be achieved even if ingress and egress ports are in a single device.

  For more information, refer to Enabling 802.1ad tagging on page 301.
- One global tag profile can be configured on a stackable device. For ICX 8200 devices, the number will be between 0 and 0xfffe. For all other devices, the number will be between 0 and 0xffff.
- On individual ports, if **tag-profile** is enabled, it points to the global tag profile.
- **Tag-profile** can also be enabled for provisional ports.

Enabling 802.1ad Tagging

To enable 802.1ad tagging, configure an 802.1Q tag on the untagged edge links (the customer ports) to any value other than the 802.1Q tag for incoming traffic. For example, in Figure 95 on page 302, the 802.1Q tag on the untagged edge links (ports 1/1/11 and 1/1/12) is 9100, whereas, the 802.1Q tag for incoming traffic is 8100.

To configure 802.1 ad tagging as shown in Figure 95 on page 302, enter commands such as the following on the untagged edge links of devices C and D.

1. Enter the global configuration mode.

```
device# configure terminal
```

2. Configure tag profile.

```
device(config)# tag-profile 9100
```

3. Enable tag profile for multiple ports.

```
device(config)# interface ethernet 1/1/11 ethernet 1/1/12
device(config-mif-1/1/11,1/1/12)# tag-profile enable
```

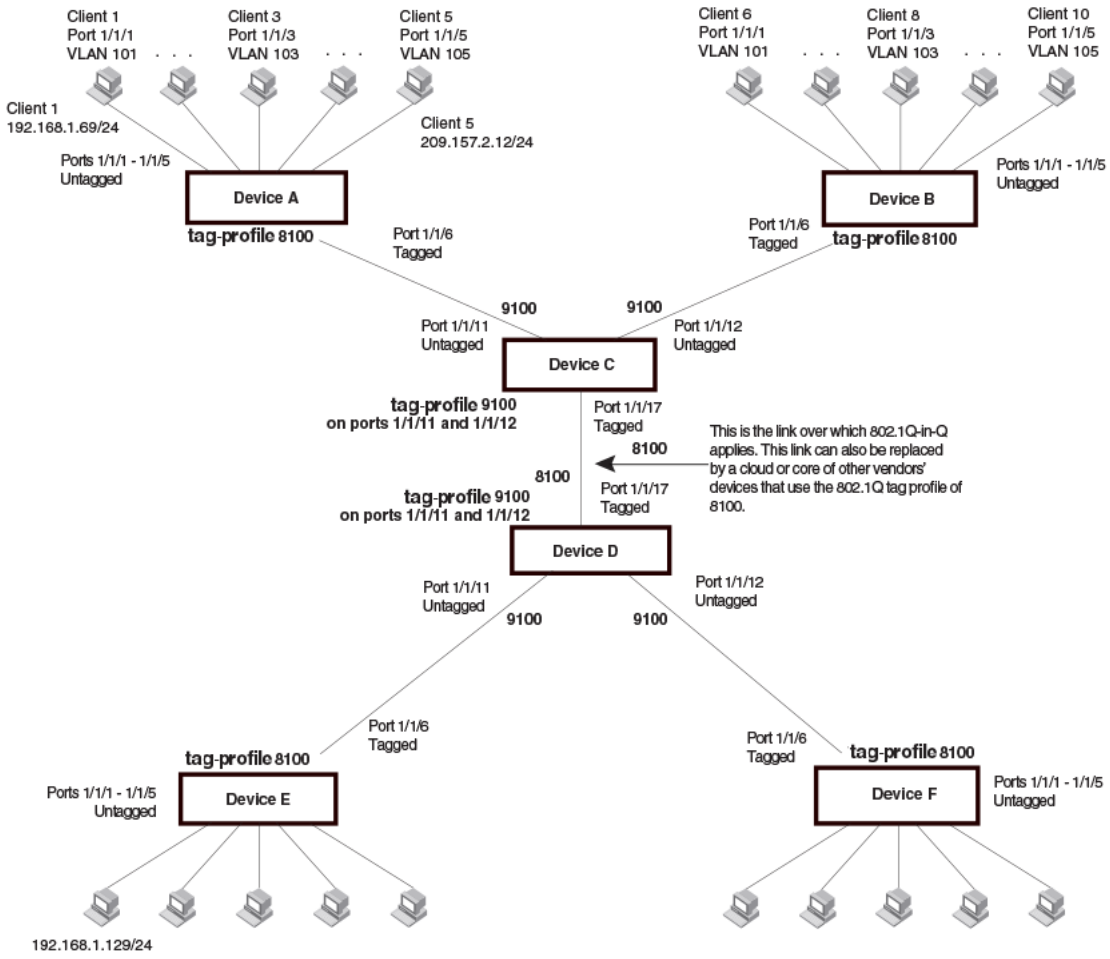
4. Enable VLAN aggregation.

```
device(config)# aggregated-vlan
```

Example 802.1ad Configuration

The following figure shows an example 802.1ad configuration.

FIGURE 95 Example 802.1ad Configuration



Selective Q-in-Q

Q-in-Q is supported in ICX products using tag-profile, where all the traffic on a tag-profile enabled port are tunneled using one service VLAN.

However, there are situations where you do not want to add a service VLAN tag to all the incoming traffic. Rather, you need to selectively tunnel a certain set of VLAN traffic while allowing regular forwarding. Selective Q-in-Q is the way to achieve Q-in-Q per CVLAN basis, where you have the flexibility to selectively choose and add a service VLAN tag based on the customer VLAN.

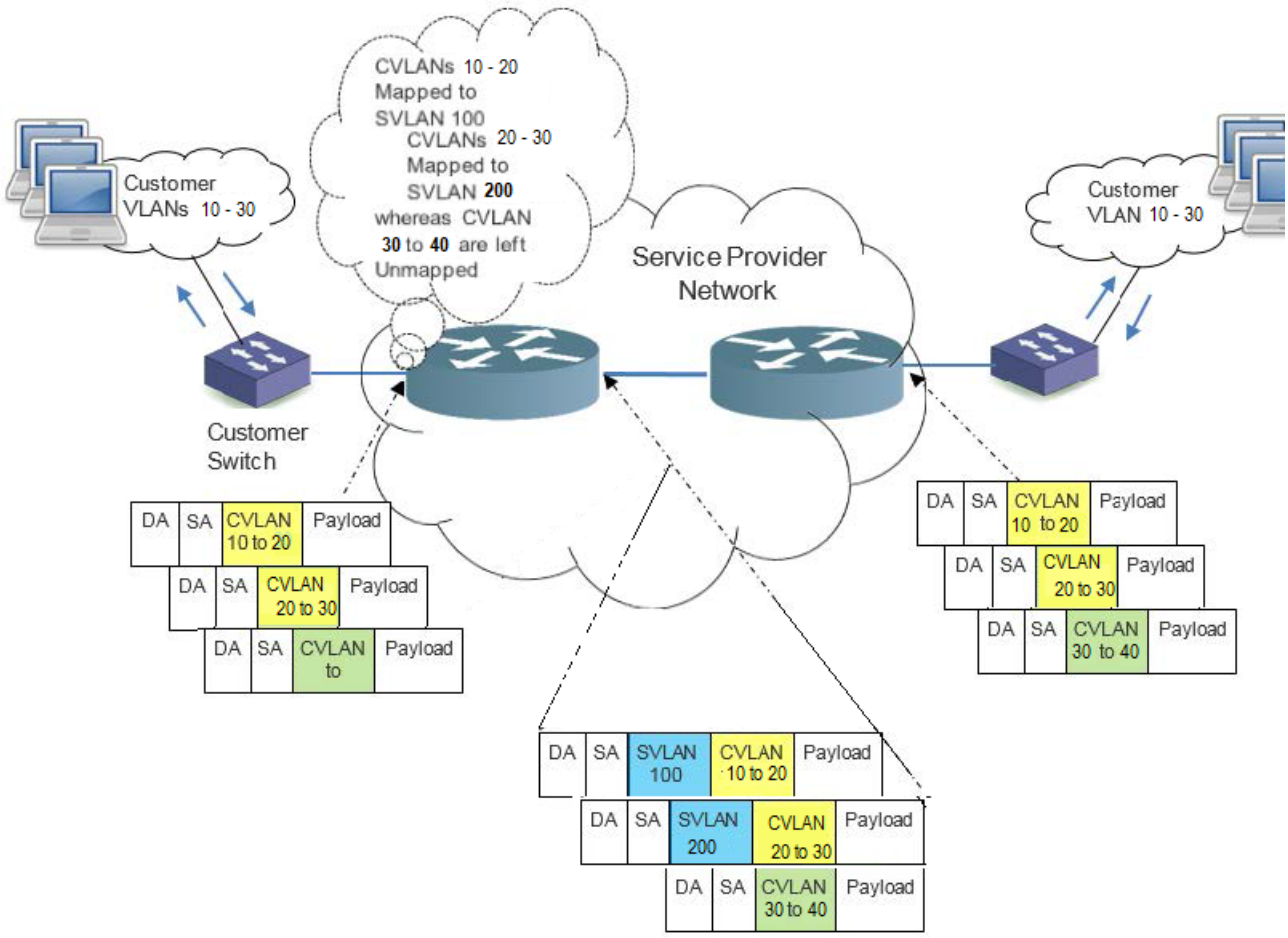
How it Works

Selective Q-in-Q enables Q-in-Q functionality per CVLAN basis. Service VLAN tags are added only for the mapped CVLANs. The unmapped CVLAN traffic is forwarded normally.

On an interface, you can configure a maximum of 50 SVLANs.

NOTE
When you downgrade to a version that does not support multiple SVLANs on an interface, only the first selective Q-in-Q configuration is accepted.

FIGURE 96 VLAN-Based Selective Q-in-Q



In the sample deployment shown in the figure above, CVLANs 10 to 20 are mapped to SVLAN 100, and are encapsulated with a service tag of 100. CVLANs 20 to 30 are mapped to SVLAN 200, and are encapsulated with a service tag of 200. Traffic from unmapped CVLANs 30 to 40 is forwarded normally. If the ingress port is not a member of the unmapped CVLANs, the traffic gets dropped as part of regular VLAN filtering.

Selective Q-in-Q VLAN Mapping

CVLANs are mapped with service VLANs using the **qinq-tunnel cvlan** command.

```
device (config-if)# qinq-tunnel cvlan <VLAN_ID [to] VLAN_ID > svlan  <svlan_id>
```

For example,

```
device(config-if)# qinq-tunnel cvlan 100 svlan 4000
device(config-if)#qinq-tunnel cvlan 2 3 5 300 to 500 svlan 4000
```

Untagged and priority tagged packets are mapped to a service VLAN using the following command.

```
device(config-if)# qinq-tunnel cvlan untag svlan <svlan_id>
```

Considerations and Limitations

Consider the following when configuring selective Q-in -Q:

- If all the customer traffic needs to be tunneled using SVLAN, it is recommended to use tag-profile based Q-in-Q.
- Q-in-Q tunnel start points and tunnel end points should have symmetric CVLANs to SVLAN mapping.
- SVLAN ID cannot be a reserved VLAN ID.
- With the implicit dual-mode support, all ports are members of untagged VLAN1 by default. Before configuring untagged selective Qin-Q tunnel on a port, you must remove it's untagged VLAN 1 membership.
- Tunneling of untagged BPDUs like LLDP or LACP works only if the selective Q-in-Q tunnel has **untag** tunneling configured.
- When an interface has multiple SVLANs configured, untagged tunneling can only be configured on one SVLAN.
- The default Tag Profile Identifier (TPID) for SVLAN tag is 8100. You can use the existing tag-profile command to change the TPID value for the SVLAN tag.
- At the tunnel end-point, the TPID of SVLAN tags in the Q-in-Q packets must match the port's configured TPID for selective Q-in-Q to function correctly where the service VLAN is removed while the traffic egresses out of service provider domain. Hence SVLAN tag TPID value should be same on tunnel start and end points.
- For selective Q-in Q, the MTU value on ICX switches is increased by 4 bytes (1522) from the default 1518 bytes to accommodate the additional dot1q tag. You can use the existing **aggregated-vlan** CLI to increase the MTU of all the ports to 1522.
- The following table displays the maximum number of selective VLAN tunneling that can be configured per stack unit in a system.

TABLE 22 Maximum Selective VLAN Tunnels per Stacking Device

ICX Platform	Maximum Number of C-VLAN Tunneling per Stack Unit
ICX 7550	8000
ICX 7650	8000
ICX 7850	8000
ICX 8200	256 unique mappings

For example, in a two unit stack, you can configure up to 16000 CVLAN mappings on the whole stack. This means, 8000 CVLANS on stack unit 1 and 8000 CVLANs on stack unit 2.
The number of VLAN mappings on a LAG is equal to number of member ports multiplied by the number of CVLAN mappings configured on the LAG interface. For example, if a LAG has 6 ports and, on the lag interface 10 CVLANs are mapped to a SVLAN, the total number of mappings is considered as 60.

Configuring Selective Q-in-Q

Perform the following steps to configure selective Q-in Q.

1. Enter global configuration mode.

```
device# configure terminal
```

2. If a port is not a member of the above specified SVLAN ID, use the below command to add the port as a tag member of the SVLAN.

```
device(config)# vlan 2
device(config-vlan-2)# tagged ethernet <stackid>/<slot>/<port>
```

3. Navigate to the interface on which Q-in-Q tunneling needs to be enabled.

```
device(config)# interface ethernet <stackid>/<slot>/<port>
device(config-if)# qinq-tunnel cvlan <vlan list> svlan <svlan_id>
```

To configure Q-in-Q tunneling for a LAG interface:

```
device(config)# interface lag <LAG_ID>
device(config-lag-if)# qinq-tunnel cvlan <vlan list> svlan <svlan_id>
```

To allow Q-in-Q tunneling of untagged customer traffic:

```
device(config)# interface ethernet 1/1/4
device(config-if-e1000-1/1/4)# qinq-tunnel cvlan untag svlan 1000
```

4. (Optional) If there is a need to change the TPID of SVLAN tag to other than 8100, complete the following configuration on the egress port which is supposed to egress out with that TPID.

```
device(config)# tag-profile <tpid/ether_type>
device(config)# interface ethernet <stackid>/<slot>/<port>
device(config-if)# tag-profile enable
```

Displaying Q-in-Q Configuration

The sample customer topology presented below shows a complete Q-in-Q configuration on both the SP edge devices.

```
device> show qinq-tunnel
        Total number of vlan(s) tunneled: 2865
        Total number of HW resource used: 2865
        Selective qinq enabled port(s): 1/1/1 1/2/2 lg1
  Port : 1/1/1 Number of CVLANs tunneled : 206
        Service vlan: 3000 Tunneled VLAN(s) : 2 to 3 5 6 7 100 to 200
        Service vlan : 3001 Tunneled VLAN(s): 400 to 500
  Port : 1/2/2 Number of CVLANs tunneled : 2600
        Service vlan : 333 Tunneled VLAN(s): 1 to 2600
  Port : lg1 Number of CVLANs tunneled: 53
        Service vlan: 3000 Tunneled VLAN (s): 500 to 550 600 , untag

device> show qinq-tunnel brief
  Total number of vlan(s) tunneled: 2865
  Total number of HW resource used: 2865
  Selective qinq enabled port(s): 1/1/1 1/2/2 lg1

device> show qinq-tunnel ethernet 1/1/1
  Port : 1/1/1 Number of CVLANs tunneled : 206
  Service vlan: 3000 Tunneled VLAN(s) : 2 to 3 5 6 7 100
```

When selective Q-in-Q is enabled, the **show interface ethernet** command displays the enabled status as follows.

```
device> show internet ethernet 1/1/16 | in Selective
  L2 Tunnel protocols enabled(mode:Selective qinq):CDP LACP LLDP STP
  Selective qinq enabled
```

Limitations

- CVLAN to SVLAN mapping must be unique. Same set of CVLAN cannot be mapped to more than one SVLAN, on the same interface.
- The CoS parameters of the CVLAN are propagated to the SVLAN. There is no configurable option provided to change this behavior.
- If incoming customer traffic is already double tagged, traffic is forwarded normally even though the outer CVLAN is mapped to a service vlan.
- Tag profile based Q-in-Q and selective Q-in-Q are mutually exclusive. Hence, cannot coexist on the same interface.
- Implicit dual-mode and untagged traffic tunneling mapping are mutually exclusive. Hence, cannot co-exist.
- When VLAN mapping is configured on a lag interface, the mapping is applied on all the member ports of the lag.
- SVLAN ID cannot be a reserved VLAN ID.
- If Bridge Protocol Data Unit (BPDU) tunneling feature is configured on the selective Q-in-Q enabled port, then BPDU tunneling is applicable only for the mapped CVLANs. BPDUs received on the unmapped CVLANs is processed in the regular manner.

- BPDU tunneling is now supported over selective Q-in-Q. But BPDU tunneling cannot be achieved without Q-in-Q tunnel. Unconfiguring selective Q-in-Q tunnel removes the BPDU tunneling configurations from the interface.
- Only one tag-profile can be configured in a system. Hence you cannot configure different TPIDs on different egress ports using the tag-profile.
- BUM or DLF traffic flooded in SVLAN domain, i.e. to all the member ports of the SVLAN might cause traffic leak between CVLANs.
- PVLAN is not supported on selective Q-in-Q enabled ports.
- Selective Q-in-Q cannot be configured on dot1x enabled ports and vice versa.
- For the ICX 8200, when the Q-in-Q tunnel is configured with CVLAN (inner VLAN) and SVLAN (outer VLAN) on an interface, the SVLAN tag (outer VLAN) is removed (if present) irrespective of CVLANs (inner VLAN) in the egress traffic.

Scaling Considerations for BPDU-Tunneling over Selective Q-in-Q

Bpdu-tunneling is used to tunnel layer2 BPDUs between customer sites over the service provider network . All the incoming BDPUs are forwarded in the slow path or CPU. Even though the total number of tunneled CVLANs is increased to 8000 per stack unit, the number of BPDUs that can be tunneled is limited, resulting in CPU load.

NOTE
The CPU usage is expected around 10 to 18% with above scale, but the actual CPU usage may vary depending on other configuration and traffic on the system. Hence, the BPDU rate can be a combination of STP, PVST, LACP and LLDP protocol PDUs.

Adding or Replacing a Customer VLAN for Q-in-Q Tunneling

When configuring selective Q-in-Q tunneling, you can add or replace a customer VLAN (CVLAN). The ability to add or replace a CVLAN enhances traffic management capabilities in environments where complex VLAN handling is required. When Internet Protocol over Ethernet (IPoE) service-tagged (S-tag) and customer-tagged (C-tag) VLANs are enabled, traffic from different clients or tenants can be segregated, ensuring the efficient use of available VLAN IDs and simplifying the network configuration.

With configuration of IPoE S-tag and C-Tag VLANs for Q-in-Q tunneling, the device can identify incoming untagged frames. It then adds both an S-tag (service VLAN or outer tag) and a C-tag (customer VLAN or inner tag) to each frame. Every port and every CPE is uniquely identified by a combination of the S-tag and C-tag. Similarly, the device can identify frames tagged with a common C-tag. Upon receipt of a tagged frame, the device adds the S-tag and replaces the common C-tag with a unique C-tag for each CPE. VLAN priority bits for Quality of Service (QoS) are preserved. For reverse traffic, the device replaces the unique C-tag with the original common C-tag for each frame before it reaches the CPE.

NOTE
IPoE S-tag and C-Tag VLANs are supported for ICX 8200 devices only.

NOTE
Selective Q-in-Q tunneling and Selective Q-in-Q tunneling with a CVLAN cannot co-exist.

The following task enables a device to add or replace new CVLANs for Q-in-Q tunneling.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Configure an Ethernet interface.

```
device(config)# interface ethernet 1/1/1
```

3. Configure Q-in-Q tunneling and add a replacement CVLAN.

```
device(config-if-e1000-1/1/1)# qinq-tunnel cvlan 1 new-cvlan 2 svlan 4000
```

   This example configures Q-in-Q tunneling for a specified CVLAN, adds a replacement CVLAN, and specifies a service VLAN to tunnel the CVLANs.

The following example configures Q-in-Q tunneling for a specified CVLAN, adds a replacement CVLAN, and specifies a service VLAN to tunnel the CVLANs.

```
device# configure terminal
device(config)# interface ethernet 1/1/1
device(config-if-e1000-1/1/1)# qinq-tunnel cvlan 1 new-cvlan 2 svlan 4000
```

The following example shows information for Q-in-Q tunnels, including information for the replacement CVLAN.

```
device> show qinq-tunnel
Total number of VLAN(s) tunneled: 2
Total number of HW resources used: 2
qinq enabled port(s): 1/1/1, 1/1/2

Port: 1/1/1    Number of CVLANs tunneled: 1
  Service VLAN: 200      Original VLAN: None (Untagged)    New CVLAN: 100

Port: 1/1/2    Number of CVLANs tunneled: 1
  Service VLAN: 200      Original VLAN: 100    New CVLAN: 75
```

802.1Q (Q-in-Q) BPDU Tunneling

BPDU tunneling over Q-in-Q enables a service provider to provide Layer 2 VPN connectivity between different customer sites. The service provider can give the customers an infrastructure to run various Layer 2 protocols and connect to all geographically-separated sites.

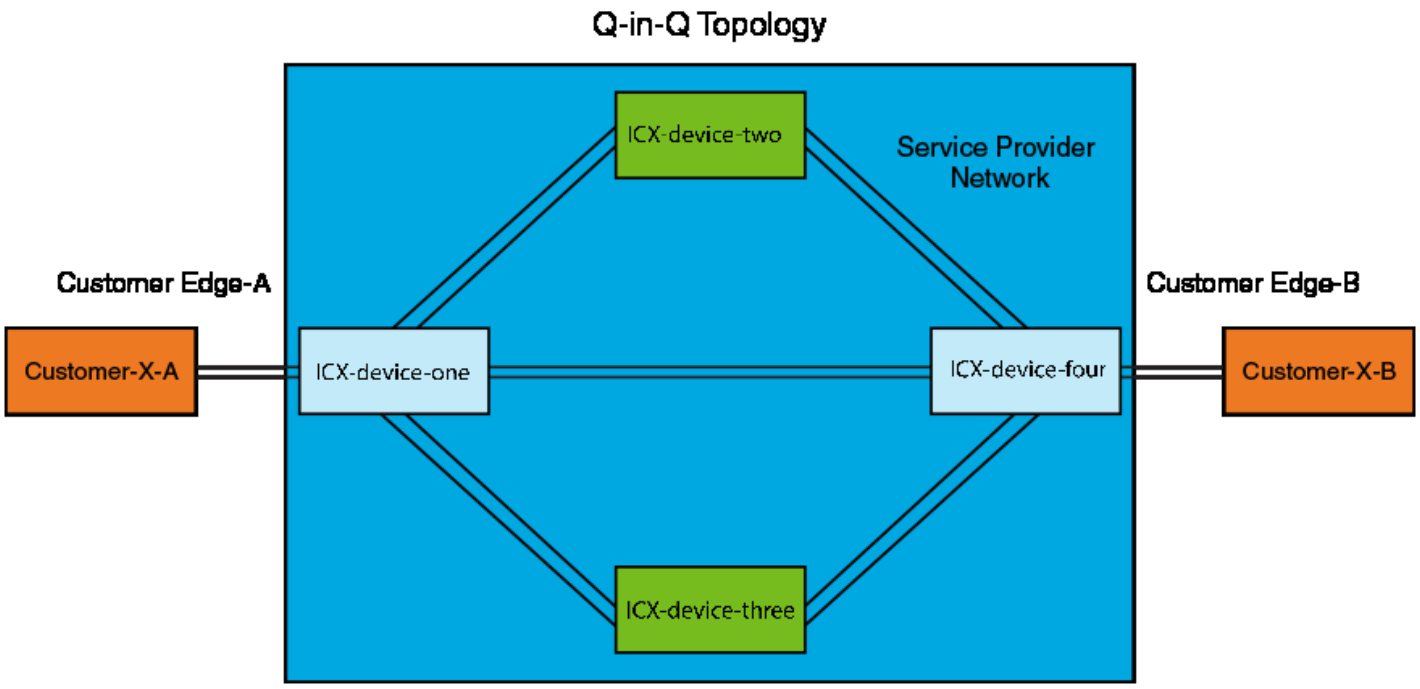
In Q-in-Q BPDU tunneling, a customer packet transferred through the service provider network is tagged twice (except for untagged customer traffic which has only one outer tag). Apart from the customer's 802.1Q VLAN tags (CVLAN), a service VLAN tag (SVLAN) is also added on all the frames. By adding different VLAN tags for each customer, traffic (control/protocol/bpdu for which tunneling is enabled) from different customers can be segregated and transferred throughout the service provider network without any VLAN conflict. Also, the service provider network is transparent to the customer and can run STP (PVST, RSTP, MSTP), LACP, CDP, and LLDP seamlessly using the Layer 2 tunneling.

How Q-in-Q BPDU Tunneling Works

When Q-in-Q BPDU tunneling is enabled, the service provider (ingress) edge device receives the BPDU packets and delivers the packets to the CPU along with CVLAN and SVLAN information. Upon ingress to the service provider network, the protocol or BPDU MAC address is replaced with a tunnel MAC address and is sent across the service provider network. Intermediate devices on the service provider network forward the frame as an unknown multicast packet. Upon egress from the service provider network to the customer network, the tunnel MAC packets are decapsulated and delivered to the customer edge device. For Layer 2 protocols such as LACP and STP to converge properly, point-to-point connections must be emulated for each port using a unique customer-to-service VLAN.

In the following topology, Customer X site A and Customer X site B are connected in the service provider network through Q-in-Q BPDU tunneling. Both data and control packets coming from the customer site with customer VLAN (CVLAN) are double-tagged with a Service VLAN (SVLAN).

FIGURE 97 Q-in-Q Topology

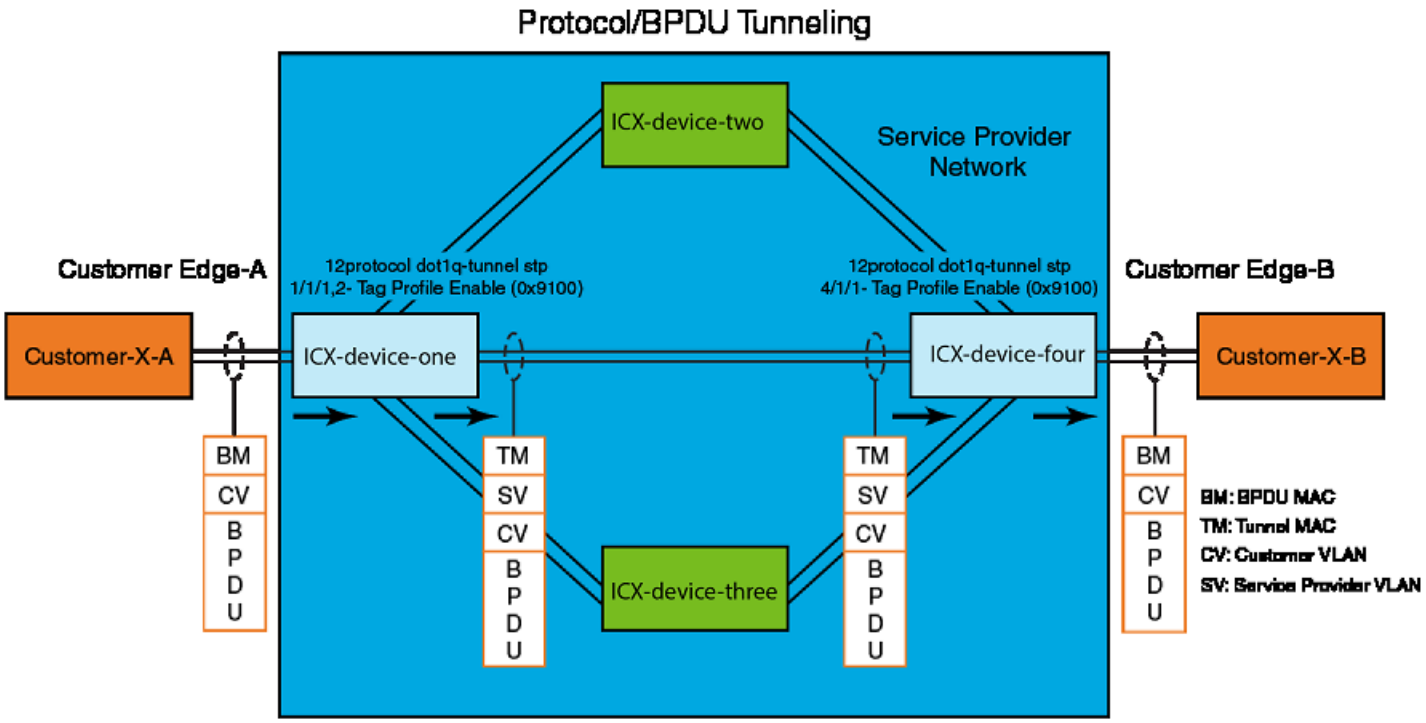


NOTE

When Layer 2 protocol tunneling is enabled on a customer-connected interface of the service provider device, all the received tunnel protocol packets will be tunneled to the service network. To prevent any locally generated protocol packets (for example, STP or LLDP) on the service provider network from switching to the customer side, the corresponding protocols must be disabled on the device.

The Protocol or BPDU packet format at various stages is shown in the following topology.

FIGURE 98 Various Stages of Protocol or BPDU Packet Format



Q-in-Q BPDU tunneling does not affect any Class of Service (CoS) values that are configured on the CVLAN. Ingress priority and CoS settings from the CVLAN are copied to the SVLAN. CoS values do not change in the reverse direction (SVLAN to CVLAN).

To add a specific check for VTP at tunnel egress, VTP is tunneled with CDP tunnel enabled using the **cdp enable** CLI. You must also run the **cdp run** CLI to enables the device to intercept and display CDP messages. For more information on CLIs, refer to the *RUCKUS FastIron Command Reference*.

Configuring Q-in-Q BPDU Tunneling

Complete the following steps to configure BPDU tunneling over Q-in-Q.

1. Enter the **configure terminal** command to enter global configuration mode.

```
device# configure terminal
```

2. Enable BPDU tunneling on tag-profile enabled port or Selective Q-in-Q enabled port.
   - Enable BPDU tunneling on a tag-profile-enabled port.

     ```
     device(config)# interface ethernet 1/1/1
     device(config-if-e1000-1/1/1)# tag-profile enable
     device(config-if-e1000-1/1/1)# l2protocol dot1q-tunnel
     ```

   - Enable BPDU tunneling on a Selective Q-in-Q-enabled port.

     ```
     device(config)# interface ethernet 1/1/1
     device(config-if-e1000-1/1/1)# qinq-tunnel cvlan 1 to 4 untag svlan 100
     device(config-if-e1000-1/1/1)# l2protocol dot1q-tunnel
     ```

   Optionally, you can enable STP (PVST, RSTP, MSTP), LACP, CDP or LLDP tunneling.

   NOTE
   ICX provides a flexibility of configuring BPDU tunneling without tag profile and selective Q in Q. This is specially for situations where the BPDU tunneling is enabled in the transit devices.

3. (Optional) Configure the maximum number of packets that can be processed on an interface before being dropped.

   ```
   device(config-if-e1000-1/1/1)# l2protocol dot1q-tunnel drop-threshold all 3000
   ```

4. (Optional) Configure the maximum number of packets that can be processed on an interface before putting the ingress port in error-disabled state.

   ```
   device(config-if-e1000-1/1/1)# l2protocol dot1q-tunnel shutdown-threshold all 3500
   ```

5. Enter the **exit** command to return to global configuration mode.

   ```
   device(config-if-e1000-1/1/1)# exit
   device(config)#
   ```

6. Specify the multicast MAC address for the tunnel from the global configuration mode.

   ```
   device(config)# l2protocol dot1q-tunnel-mac 0100.1a2b.3c4d
   ```

7. (Optional) Specify a global Class of Service (CoS) value on all Q-in-Q tunneling ports from the global configuration mode. .

   ```
   device(config)# l2protocol dot1q-tunnel cos 6
   ```

   The ingress BPDUs on the tunnel ports are encapsulated with the specified class. The default CoS value is 5.

The following example shows the steps to configure Q-in-Q BPDU tunneling.

```
device# configure terminal
device(config)# interface ethernet 1/1/1
device(config-if-e1000-1/1/1)# tag-profile enable
device(config-if-e1000-1/1/1)# l2protocol dot1q-tunnel
device(config-if-e1000-1/1/1)# l2protocol dot1q-tunnel drop-threshold all 3000
device(config-if-e1000-1/1/1)# l2protocol dot1q-tunnel shutdown-threshold all 3500
device(config-if-e1000-1/1/1)# exit
device(config)# l2protocol dot1q-tunnel-mac 0100.1a2b.3c4d
device(config)# l2protocol dot1q-tunnel cos 6
```

The following example shows the steps to configure BPDU tunneling on Selective Q-in-Q-enabled port .

```
device# configure terminal
device(config)# interface ethernet 1/1/1
device(config-if-e1000-1/1/1)# qinq-tunnel cvlan 1 to 4 untag svlan 100
device(config-if-e1000-1/1/1)# l2protocol dot1q-tunnel
device(config-if-e1000-1/1/1)# l2protocol dot1q-tunnel drop-threshold all 3000
device(config-if-e1000-1/1/1)# l2protocol dot1q-tunnel shutdown-threshold all 3500
device(config-if-e1000-1/1/1)# exit
device(config)# l2protocol dot1q-tunnel-mac 0100.1a2b.3c4d
device(config)# l2protocol dot1q-tunnel cos 6
```

NOTE

The **l2protocol dot1q-tunnel** enables protocol tunneling for xSTP, LACP, LLDP, CDP, and VTP.

Simultaneous Support for Tagged and Untagged VLANs

Every interface is an untagged member of default VLAN or any user specified VLAN, unless user explicitly removes this untagged interface from default VLAN. The interface will always remain in the default VLAN (or non-default VLAN in which the interface is configured as untagged member), when this interface is added as tagged member of non-default VLAN. Thus, all tagged interfaces are implicit dual-mode ports.

This implicit dual-mode feature simplifies the tagging concept by changing the system default behavior to allow both tagged and untagged frames simultaneously (as per configurations). The dual-mode keyword is now deprecated, thus eliminating the need of explicit dual-mode configuration.

NOTE

With the introduction of implicit dual mode, CCEP ports cannot be configured as untagged member of default VLAN, when the default VLAN is a non MCT VLAN. Hence, the dual-mode configuration is disabled on CCEP ports as part of the image upgrade to current release.

```
device(config)# vlan 100
device(config-vlan-100)# tagged ethe 1/1/1
Added tagged port(s) ethe 1/1/1 to port-vlan 100.
device(config-vlan-100)# vlan 1
device(config-vlan-1)# no untagged ethe 1/1/1
```

Since user is allowed to remove ports from default VLAN which completely removes the untagged VLAN membership of an interface, this will be saved to running configuration to retain the VLAN membership upon reload. User VLANs will continue to display membership as per user configurations.

```
device(config)# vlan 100
device(config-vlan-100)# untagged ethe 1/1/1
Added untagged port(s) ethe 1/1/1 to port-vlan 100.
device(config-vlan-100)# vlan 101
device(config-vlan-1)# tagged ethe 1/1/2 to 1/1/3
device(config-vlan-100)# vlan 1
device(config-vlan-1)# no untagged ethe 1/1/2 to 1/1/3
device(config-vlan-1)# sh ru vlan
vlan 1 name DEFAULT-VLAN by port
no untagged ethe 1/1/2 to 1/1/3***
!
vlan 100 by port
 untagged ethe 1/1/1
!
```

As you can see, only the interfaces that are deleted explicitly by the user is displayed. If an interface is moved internally as part of some other VLAN configurations (1/1/1, in the above example), it is not displayed under default-VLAN.

NOTE

Show VLAN output will no more display dual-mode and uplink ports. This command displays VLAN membership as per the option provided. There is no change in VLAN display order, but default VLAN displays the interfaces that are explicitly removed from default VLAN using the **no untagged ethernet** command.

VLAN Mapping

VLAN Mapping provides a mechanism for Service Providers to translate CVLANs to SVLANs when a packet enters its network and vice-versa, when it leaves the network.

For RUCKUS ICX 7550, ICX 7650, and ICX 7850 devices, VLAN translation is enabled on a per-port basis, where a CVLAN is mapped to an SVLAN. The CVLAN tag in the packet is replaced with the configured SVLAN tag within the service provider network. When the packet leaves the service provider network, the SVLAN tag in the packet egressing will be replaced with the CVLAN tag.

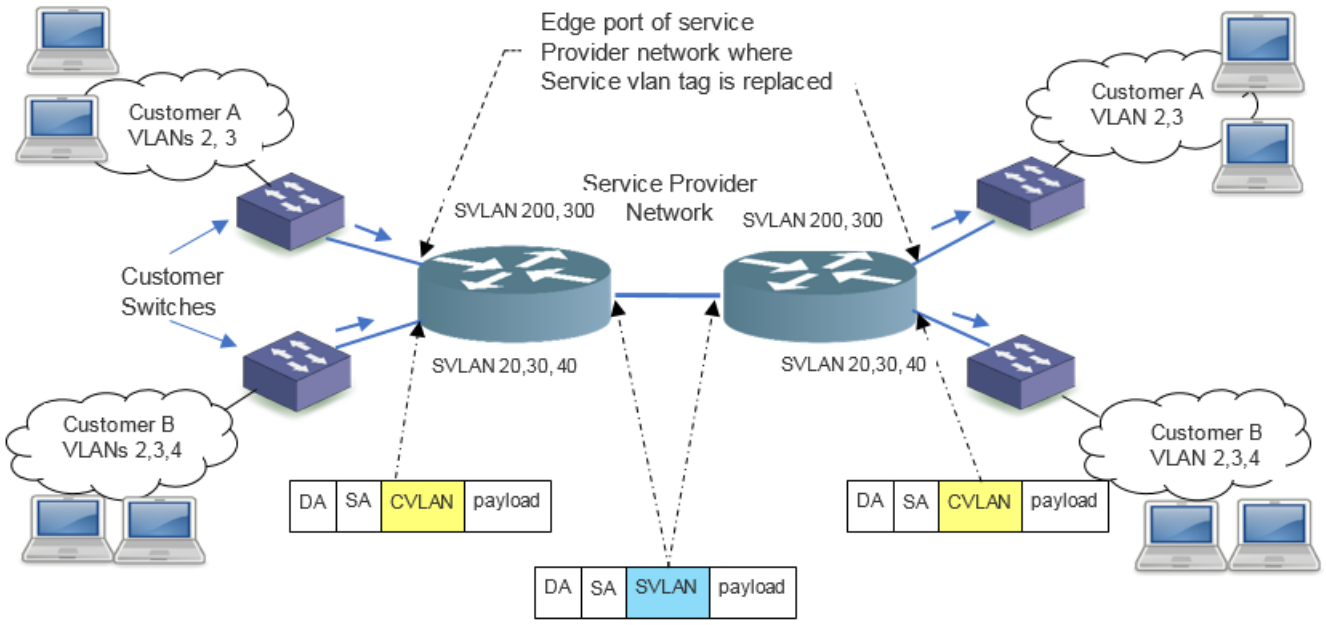
> **NOTE**
> This is different from Q-in-Q because we do not double tag; instead, we replace the VLAN tag.

Typically, the same VLAN mapping configuration must be done for all the edge ports to the same customer. This feature is supported on all the existing ICX hardware platforms.

For the RUCKUS ICX 8200 devices, VLAN translation is available in global mode and mapping is removed at the interface configuration level. Therefore, the same CVLANs cannot be supported across multiple customers.

Basic VLAN Mapping Deployment

FIGURE 99 VLAN Mapping for RUCKUS ICX 7550, ICX 7650, and ICX 7850 devices



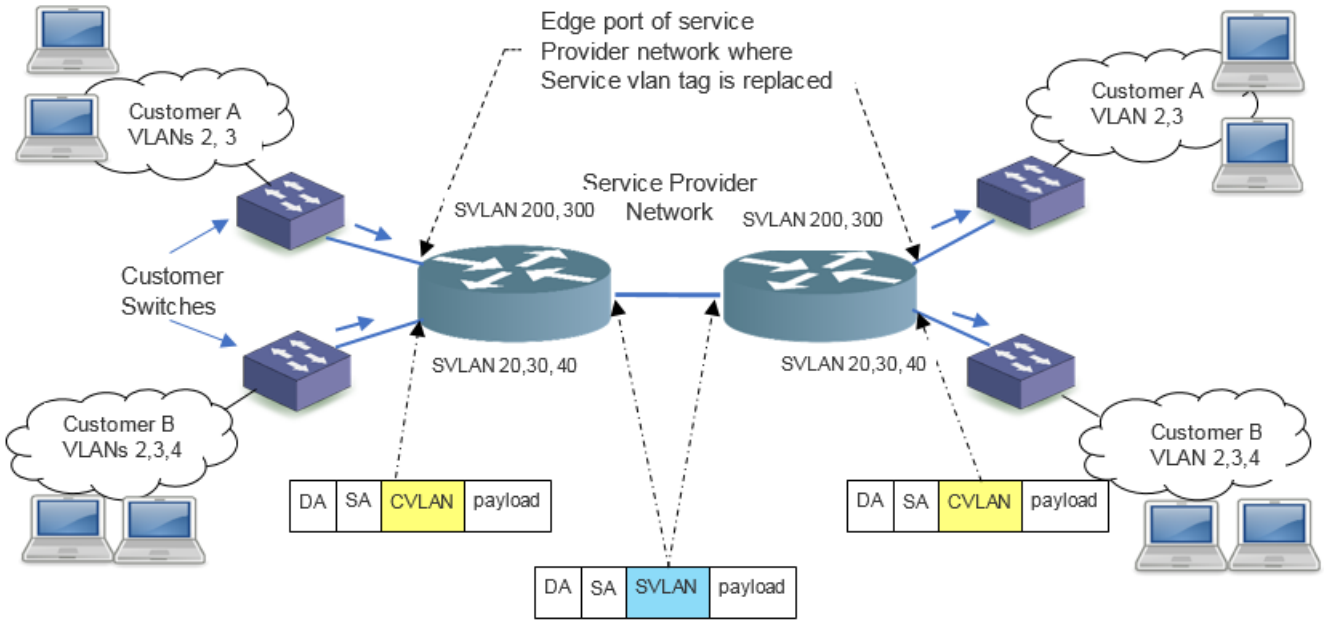
The above figure depicts a typical VLAN mapping deployment for RUCKUS ICX 7550, ICX 7650, and ICX 7850 devices.

Customer A runs on VLAN 2 and VLAN 3. The service provider maps customer A's VLAN 2 and VLAN 3 to service provider's VLAN 200 and VLAN 300 respectively. This mapping is done on both edge ports on the service provider's network, where the customer network is connected.

Similarly, Customer B runs on VLAN 2, VLAN 3 and VLAN 4. The service provider maps customer A's VLAN 2, VLAN 3 and VLAN 4 to service provider's VLAN 20, VLAN 30 and VLAN 40 respectively. This mapping is done on both edge ports on the service provider's network, where the customer network is connected.

So, within the service provider network, traffic on VLANS 200 and VLAN 300 signifies traffic for Customer A and traffic on VLANs 20, VLAN 30, and VLAN 40 signifies traffic for Customer B.

FIGURE 100 VLAN Mapping for RUCKUS ICX 8200 devices



The above figure depicts a typical VLAN mapping deployment for RUCKUS ICX 8200 devices.

VLAN Mapping Configuration for RUCKUS ICX 7550, ICX 7650, and ICX 7850 devices

The VLAN mapping is enabled using the vlan-mapping command. Execute the following steps to configure VLAN mapping for RUCKUS ICX 7550, ICX 7650, and ICX 7850 devices.

1. Enter the global configuration mode.

```
device# configure terminal
```

2. Navigate to the interface on which VLAN mapping needs to be enabled.

```
device(config)# interface ethernet <stack id/slot/port>
device(config-if)# vlan-mapping cvlan <vlan_id> svlan <svlan_id>
```

3. If the port is not a member of the above specified SVLAN ID, execute the following command to add the port as a tag member of the SVLAN.

```
device(config-if)# vlan-config add <svlan_id>
```

Alternatively, you can go to the VLAN configuration mode and add the port as member of that VLAN.

```
device(config)# vlan 2
device(config-vlan-2)# tagged ethernet <stack id/slot/port>
```

To view the VLAN mapping configuration, run the **show vlan-mapping brief** command. A sample output is as follows.

```
device# show vlan-mapping brief
   Total number of vlan(s) mapped: 30
   Total number of HW resource used: 50
   vlan-mapping enabled port(s): 1/1/33 1/1/34 lg10
```

Each port can have one or more VLAN mappings. If a packet reaches the port with a VLAN tag for which there is no mapping present, the packet flows through the service provider network, unmapped. If the network provider wants to restrict this behavior and wants all unmapped packets to be dropped, use the following configuration.

```
device(config-if)# vlan-mapping default drop
```

VLAN Mapping Configuration for RUCKUS ICX 8200 devices

1. Enter the global configuration mode.

   ```
   device# configure terminal
   ```

2. Navigate to the interface on which VLAN mapping needs enabled.

   ```
   device(config)# vlan-mapping cvlan <vlan_id> svlan <svlan_id>
   ```

3. Configure VLAN mapping enable on the interface.

   ```
   device(config-if-interface)# vlan-mapping enable
   ```

4. To view the VLAN mapping configuration, run the **show vlan-mapping brief** command. A sample output is as follows.

   ```
   device(config)# show vlan-mapping brief
   Total number of vlan(s) mapped: 1
   Total number of HW resource used: 1
   Vlan-mapping enabled port(s): 1/1/1 1/1/2
   CVLAN     SVLAN
   10     200
   ```

VLAN Mapping Considerations

- VLAN mapping is not supported in untagged port.
- CVLAN to SVLAN mapping is always one to one and exclusive for each interface. This means, on a specific interface for a specific CVLAN, there can only be one SVLAN mapped and vice versa. On an interface, more than one CVLANs cannot be mapped to the same SVLAN, and the same CVLAN cannot be mapped to more than one SVLANs.
- Both network start point and end point interfaces must have the same vlan-mapping configuration for translating CVLANs to SVLAN and vice versa.
- If incoming customer traffic is already double tagged, then the mapping is done on the outer tag.
- Tag profile shall not be used in conjunction with vlan-mapping on an interface. This means, the interface on which VLAN-mapping is enabled must not be enabled for tag profile. The default tag in the packet should be 8100, if it should be considered for VLAN mapping.
- If global spanning tree is enabled on the box, on ports where VLAN mapping is enabled, no-span should be enabled.
- VLAN mapping cannot coexist with the following features: PMS, PVLAN, selective Q-in-Q, tag profile-based Q-in-Q, and dot1x.

- For all forwarding (L2, L3 and other pipelines in packet processor) and L3 purposes, SVLAN is used.
- For RUCKUS ICX 8200 devices VLAN mapping is applicable to all VLAN mapping enabled ports.

Scaling Considerations for RUCKUS ICX 7550, ICX 7650, and ICX 7850 devices

The maximum number of CVLAN to SVLAN mapping per port limit is 10. The maximum number of VLAN mappings which can be configured in a system is 1024.

A maximum 10 VLAN mappings can be configured on an interface. However, a typical deployment scenario will need only two or three VLAN mappings per interface. The number of VLAN mapping on a lag is equal to number of member ports multiplied by the number of CVLANs mapping configured on the lag interface. For example, if a lag has 6 ports and 10 CVLANs are mapped to the SVLAN, the total number of mappings is considered as 60.

NOTE
These values are based on the hardware capabilities.

Scaling Considerations for RUCKUS ICX 8200 devices

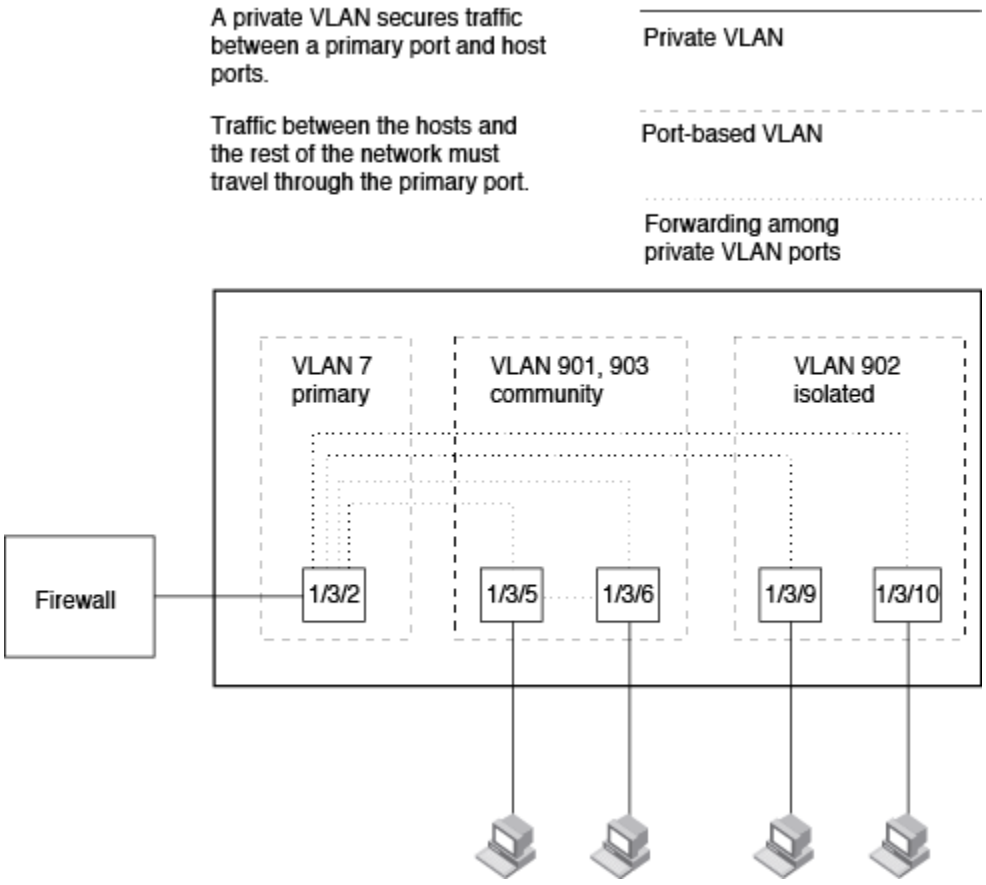
The maximum number of VLAN mappings which can be configured in a system is 1024.

Private VLAN Configuration

A private VLAN (PVLAN) is a VLAN that has the properties of standard Layer 2 port-based VLANs but also provides additional control over flooding packets on a VLAN. The following table shows an example of an application using a PVLAN.

NOTE
Flexible authentication is not supported in private VLANs.

FIGURE 101 PVLAN Used to Secure Communication Between a Workstation and Servers



This example uses a PVLAN to secure traffic between hosts and the rest of the network through a firewall. Five ports in this example are members of a PVLAN. The first port (port 1/3/2) is attached to a firewall. The next four ports (ports 1/3/5, 1/3/6, 1/3/9, and 1/3/10) are attached to hosts that rely on the firewall to secure traffic between the hosts and the rest of the network. In this example, two of the hosts (on ports 1/3/5 and 1/3/6) are in a community PVLAN, and thus can communicate with one another as well as through the firewall. The other two hosts (on ports 1/3/9 and 1/3/10), are in an isolated VLAN and thus can communicate only through the firewall. The two hosts are secured from communicating with one another even though they are in the same VLAN.

By default, unknown-unicast, unregistered multicast, and broadcast are flooded in PVLAN.

By default, on all the FastIron platforms, the device will forward broadcast, unregistered multicast, and unknown unicast packets from outside sources into the PVLAN.

You can configure a combination of the following types of PVLANs:

- Primary: The primary PVLAN ports are "promiscuous". They can communicate with all the isolated PVLAN ports and community PVLAN ports in the isolated and community VLANs that are mapped to the promiscuous port.
- Isolated: Broadcasts and unknown unicasts received on isolated ports are sent only to the promiscuous ports and switch - switch ports. They are not flooded to other ports in the isolated VLAN.

NOTE
On all devices, however, private VLANs will act as a normal VLAN and will flood unknown destinations, broadcast and multicast traffic to all ports in the VLAN if the primary VLAN does not have the PVLAN mapping that defines the uplink port for the isolated VLAN.

- Community: Broadcasts and unknown unicasts received on community ports are sent to the primary port and also are flooded to the other ports in the community VLAN.

Each PVLAN must have a primary VLAN. The primary VLAN is the interface between the secured ports and the rest of the network. The PVLAN can have any combination of community and isolated VLANs.

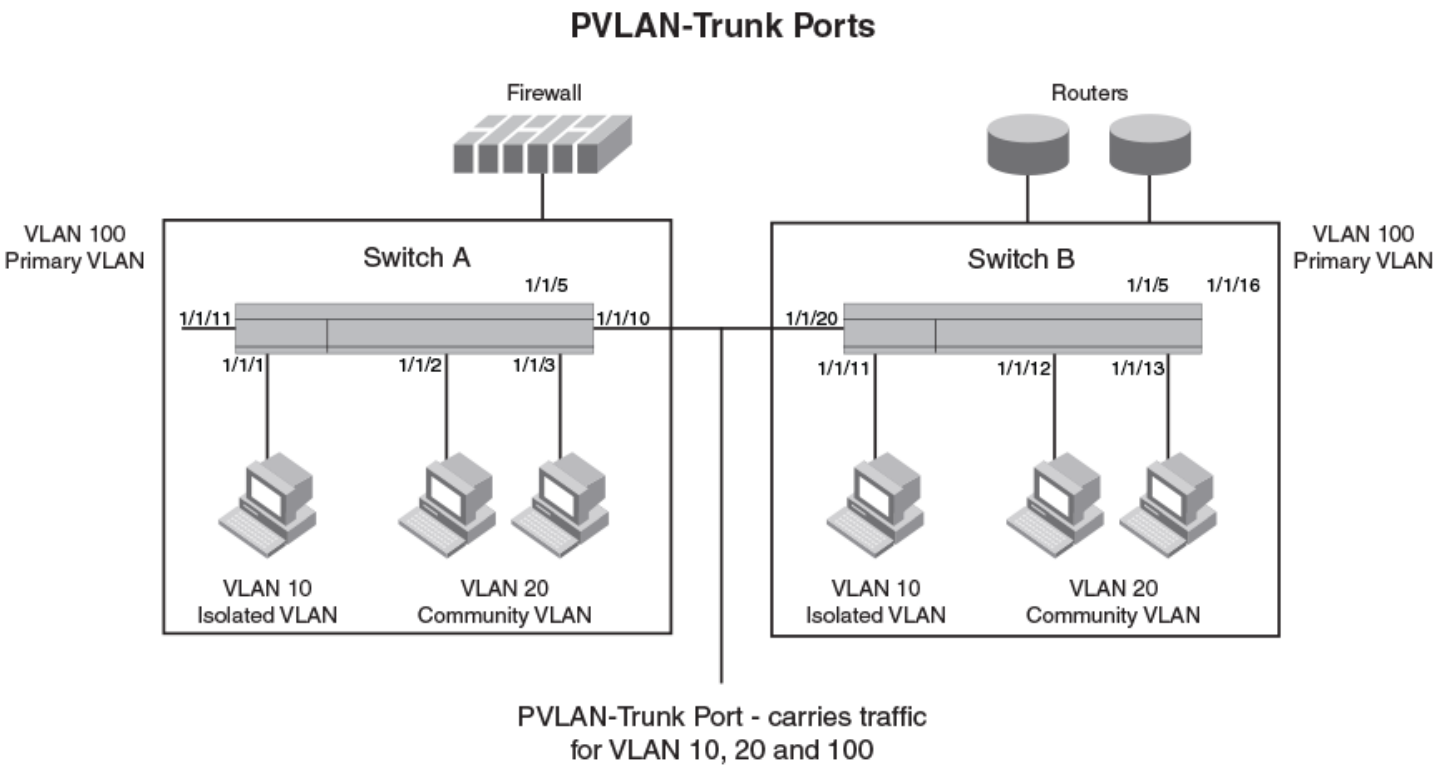
As with regular VLANs, PVLANs can span multiple switches. The PVLAN is treated like any other VLAN by the PVLAN-trunk ports.

NOTE
ISL(Inter-Switch Link) is an alias for PVLAN-trunk ports.

Figure 102 shows an example of a PVLAN network across switches:

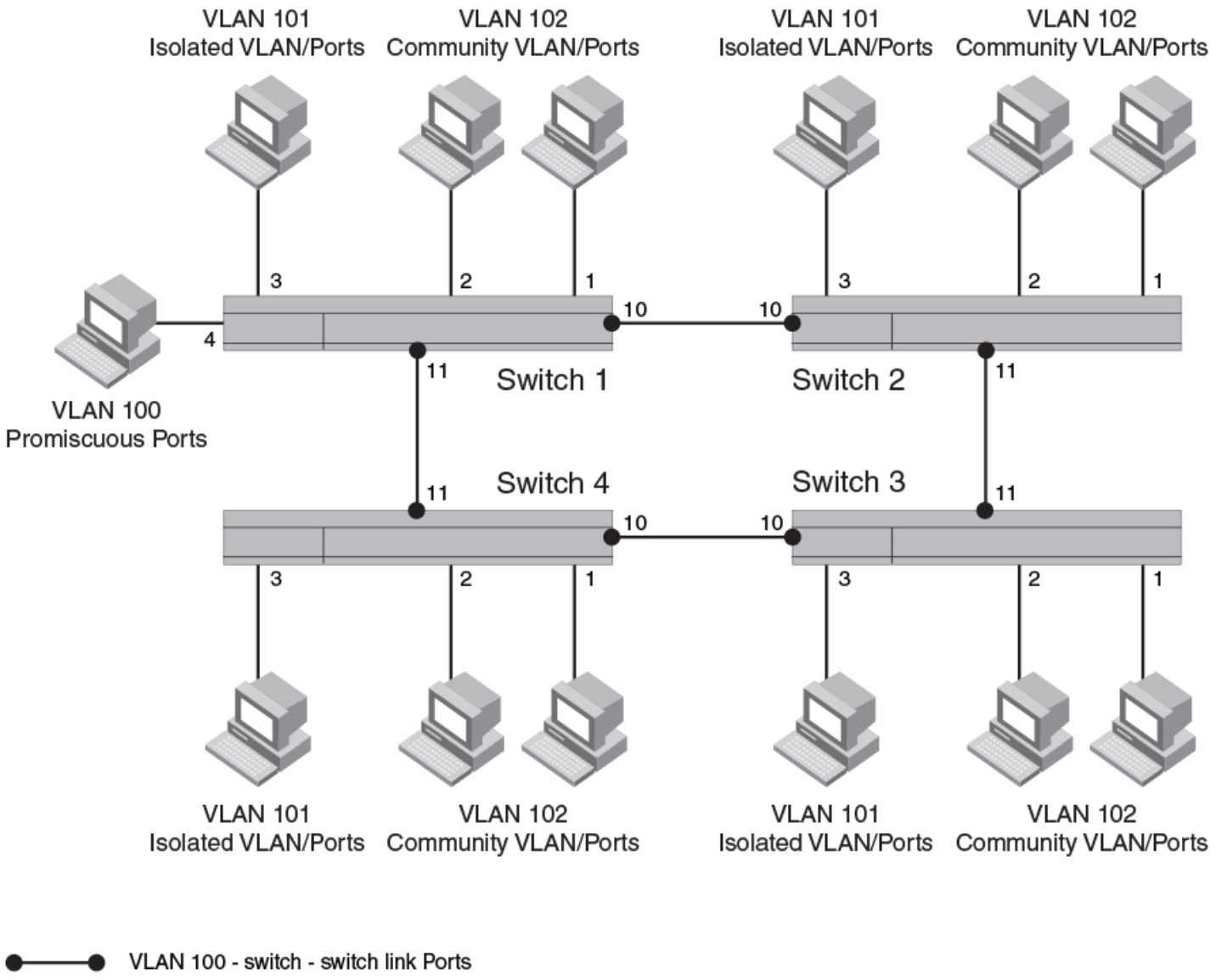
- Broadcast, unknown unicast or unregistered multicast traffic from the primary VLAN port is forwarded to all ports in isolated and community VLANs in both the switches.
- Broadcast, unknown unicast or unregistered multicast traffic from an isolated port in switch A is not forwarded to an isolated port in switch A. It will not be forwarded to an isolated port in switch B across the PVLAN-trunk port.
- Broadcast, unknown unicast or unregistered multicast traffic from a community port in switch A will be forwarded to a community port in switch B through the PVLAN-trunk port. It is forwarded to the promiscuous ports and PVLAN-trunk ports of the primary VLAN.

FIGURE 102 PVLAN Across Switches



The following figure shows an example PVLAN network with tagged switch-switch link ports.

FIGURE 103 Example PVLAN Network with Tagged Ports



The following table lists the differences between PVLANs and standard VLANs.

TABLE 23 Comparison of PVLANs and Standard Port-Based VLANs

Forwarding Behavior	Private VLANs	Standard VLANs
All ports within a VLAN constitute a common layer broadcast domain	No	Yes
Broadcasts and unknown unicasts are forwarded to all the VLAN ports by default	No (isolated VLAN) Yes (community VLAN) Yes (Primary)	Yes

TABLE 23 Comparison of PVLANs and Standard Port-Based VLANs (continued)

Forwarding Behavior	Private VLANs	Standard VLANs
Known unicasts	No (isolated VLAN) Yes (community VLAN) Yes (Primary)	Yes

Multiple Tagged and Untagged Support for PVLANs

RUCKUS ICX devices allow users to configure primary or secondary VLAN ports in multiple primary or secondary VLANs of the same type in other PVLAN domains.

Private VLANs allow Layer 2 segregation and also minimizes usage of system VLANs. Using implicit dual-mode ports in a private VLAN enables same ports to carry data or voice traffic.

The following matrix provides information about possible configurations that are allowed in a PVLAN.

NOTE
All user configurations beyond the scope of the table will not be allowed.

TABLE 24 Possible Configurations Allowed in a PVLAN

PVLAN port	Port type		Primary VLAN of other PVLAN		Isolated VLAN of the other PVLAN	Other community VLAN of the same PVLAN	Community VLAN of the other PVLAN	Regular VLAN
	Tagged	Untagged	Promiscuous ports	ISL ports				
Promiscuous ports	Yes	Yes	Yes	No	No	No	No	No
Inter Switch Link (ISL) ports	Yes	No	No	Yes	No	No	No	No
Isolated VLAN ports	Yes	Yes	NA	NA	Yes	No	No	No
Community VLAN ports	Yes	Yes	NA	NA	No	No	Yes	No

Configuration Notes for PVLANs and Standard VLANs

- PVLANs are supported on untagged ports on all RUCKUS ICX platforms.
- Normally, in any port-based VLAN, the device floods unknown unicast, unregistered multicast, and broadcast packets in hardware, although selective packets, such as IGMP, may be sent only to the CPU for analysis, based on the IGMP snooping configuration. When protocol is enabled, or if PVLAN mappings are enabled, the RUCKUS ICX device will flood unknown unicast, and unregistered multicast packets in software. The flooding of broadcast or unknown unicast from the community or isolated VLANs to other secondary VLANs will be governed by the PVLAN forwarding rules. The switching is done in hardware and thus the CPU does not enforce packet restrictions.
- RUCKUS ICX devices forward broadcast, unregistered-multicast, and unknown unicast traffic in hardware if PVLAN mappings are enabled. When PVLAN mappings are enabled, multiple MAC entries for the same MAC do not appear in the MAC table, instead all the MAC entries are learned in the primary VLAN.
- To configure a PVLAN, configure each of the component VLANs (isolated, community, and primary) as a separate port-based VLAN:
  - Use standard VLAN configuration commands to create the VLAN and add ports.
  - Identify the PVLAN type (isolated, community, or primary)
  - For the primary VLAN, map the other secondary PVLANs to the ports in the primary VLAN

- A primary VLAN can have multiple ports. All these ports are active, but the ports that will be used depends on the PVLAN mappings. Also, secondary VLANs (isolated and community VLANs) can be mapped to more than one primary VLAN port.
- You can configure PVLANs and implicit dual-mode VLAN ports on the same device. However, the VLAN ports, other than those which are implicit dual-mode in system default VLAN, can be member ports in a PVLAN domain.
- When a implicit dual-mode port in system default VLAN is added to a private VLAN, that port is removed from default VLAN.
- VLAN identifiers configured as part of a PVLAN (primary, isolated, or community) should be consistent across the switched network. The same VLAN identifiers cannot be configured as a normal VLAN or a part of any other PVLAN.
- Implicit dual mode ports which are untagged to non-default VLAN are supported in a private VLAN domain. However, since ISL ports can only be tagged ports, they cannot be enabled on implicit dual-mode ports.
- Member ports in a private VLAN domain can be extended to other domains as long as they belong to the same private VLAN type. Refer to the "**Possible configurations allowed in a PVLAN**" table to know more about allowed configurations in a PVLAN. All user configurations beyond the scope of the table will not be allowed.
- PVLAN ports cannot be added to regular VLANs. You must configure the regular VLAN as PVLAN before adding the ports.
- A regular VLAN cannot be configured as PVLAN, if the ports of the regular VLAN are part of any other regular VLAN. In such a scenario, you must configure both regular VLANs as PVLANs.
- PVST, when needed in PVLANs, should be enabled on all (primary and secondary) private VLANs across switches.
- Port MAC security is not supported on ports in a private VLAN domain.

TABLE 25 PVLAN Support Matrix

Platform	Forwarding Type	Tagged Port	Untagged Port	ISL Port	Multiple Promiscuous Port
ICX 8200	Hardware	Yes	Yes	Yes	Yes
ICX 7650	Hardware	Yes	Yes	Yes	Yes
ICX 7850	Hardware	Yes	Yes	Yes	Yes
ICX 7550	Hardware	Yes	Yes	Yes	Yes

Configuration Considerations for Isolated or Community PVLAN

The following are some of the configuration considerations to be noted for configuring isolated and community PVLANs.

Isolated VLANs

- Every isolated VLAN should be in a unique primary VLAN domain.
- An isolated port can be untagged (implicit-dual-mode) in a non-default VLAN. It need not be an implicit dual-mode port. It can be a purely tagged interface.

  NOTE
  The same is applicable to community and primary VLANs.

- An isolated port (member of an isolated VLAN) communicates with the promiscuous port, if a promiscuous port is configured. If a switch-switch port is configured, the isolated port communicates with the switch-switch port also.
- An isolated VLAN must be associated with the primary VLAN for traffic to be isolated between isolated VLAN ports and to be switched across primary VLAN ports.
- An isolated VLAN is associated with only one primary VLAN in entire switched network.
- A primary VLAN can be associated with only one isolated VLAN. An isolated VLAN can only be mapped to a promiscuous port and a switch-switch link port that belong to the same primary VLAN.

Community VLANs

- Every community VLAN should be in a unique primary VLAN domain.
- A port being added to the community VLAN is an implicit dual-mode port.
- A community VLAN is associated with only one primary VLAN and to the same primary VLAN in the entire switched network.
- A primary VLAN can be associated with multiple community VLANs.
- A community VLAN must be associated with the primary VLAN for traffic from the community port to be switched across primary VLAN ports.

These commands create port-based VLAN 901, add ports 1/3/5 and 1/3/6 to the VLAN as untagged ports, then specify that the VLAN is a community PVLAN.

The **untagged ethernet** or **tagged ethernet** command adds the ports to the VLAN.

The **pvlan type** command specifies that this port-based VLAN is a PVLAN and can be of the following types:

- **community**: Broadcasts and unknown unicasts received on community ports are sent to the primary port and also are flooded to the other ports in the community VLAN.
- **isolated**: Broadcasts and unknown unicasts received on isolated ports are sent only to the primary port. They are not flooded to other ports in the isolated VLAN.
- **primary**: The primary PVLAN ports are "promiscuous". They can communicate with all the isolated PVLAN ports and community PVLAN ports in the isolated and community VLANs that are mapped to the promiscuous port.

Changing from one PVLAN type to another (for example, from primary to community or vice versa) is allowed but the mapping will be removed.

Configuring a Primary or a Community or an Isolated PVLAN

You can configure PVLAN as a primary or a community or an isolated PVLAN using the **pvlan type** keyword from the VLAN configuration mode. By default, PVLAN mapping is not configured.

PVLAN and inter-switch link (ISL) mapping are applicable only to the primary PVLAN.

1. Enter the global configuration mode.

```
device# configure terminal
```

2. Configure VLANs.

```
device(config)# vlan 901
device(config)# vlan 902
device(config)# vlan 903
```

3. Add ports to the VLAN.

```
device(config-vlan-901)# untagged ethernet 1/3/5 to 1/3/6
```

In the example, ports 1/3/5 and 1/3/6 are added to VLAN 901 as untagged ports.

4. Configure PVLAN as a primary or a community or an isolated PVLAN.
   a) Configure PVLAN as a primary PVLAN.

```
device(config-vlan-901)# pvlan type primary
```

   b) Configure PVLAN as a community PVLAN.

```
device(config-vlan-902)# pvlan type community
```

   c) Configure PVLAN as an isolated PVLAN.

```
device(config-vlan-903)# pvlan type isolated
```

5. Map the secondary VLAN to the primary VLAN.

```
device(config-vlan-901)# pvlan mapping 101 ethernet 1/3/5
```

6. Configure ISL for the primary VLAN.

```
device(config-vlan-901)# pvlan pvlan-trunk 101 ethernet 1/3/5
```

   The **pvlan pvlan-trunk** command is not allowed on the secondary VLANs.

The following example shows how to configure a primary PVLAN and to identify ISL in the primary PVLAN.

```
device# configure terminal
device(config)# vlan 901
device(config-vlan-901)# untagged ethernet 1/3/2
device(config-vlan-901)# pvlan type primary
device(config-vlan-901)# pvlan mapping 101 ethernet 1/3/2
device(config-vlan-901)# pvlan pvlan-trunk 101 ethernet 1/3/2
```

The following example shows how to configure a community PVLAN.

```
device# configure terminal
device(config)# vlan 902
device(config-vlan-902)# untagged ethernet 1/3/5 to 1/3/6
device(config-vlan-902)# pvlan type community
```

The following example shows how to configure an isolated PVLAN.

```
device(config)# vlan 903
device(config-vlan-903)# untagged ethernet 1/3/5 to 1/3/6
device(config-vlan-903)# pvlan type isolated
```

CLI Example for a General PVLAN Network

To configure the PVLANs shown in Figure 101 on page 316, enter the following commands.

1. Create a VLAN.

```
device(config)# vlan 901
```

2. Add the untagged ethernet ports to this VLAN.

```
device(config-vlan-901)# untagged ethernet 1/3/5 to 1/3/6
```

3. Configure the PVLAN as community type.

```
device(config-vlan-901)# pvlan type community
device(config-vlan-901)# exit
```

4. Create a VLAN 902 and add the untagged ethernet ports.

```
device(config)# vlan 902
device(config-vlan-902)# untagged ethernet 1/3/9 to 1/3/10
```

5. Configure the VLAN 902 as isolated VLAN in a PVLAN.

```
device(config-vlan-902)# pvlan type isolated
device(config-vlan-902)# exit
```

6. Create another VLAN and add the untagged ethernet ports.

```
device(config)# vlan 903
device(config-vlan-903)# untagged ethernet 1/3/7 to 1/3/8
```

7. Configure the PVLAN as community type.

```
device(config-vlan-903)# pvlan type community
device(config-vlan-903)# exit
```

8. Create a new VLAN and configure as primary PVLAN type. Map the isolated VLAN and community VLAN to the primary VLAN as promiscuous ports.

```
device(config)# vlan 7
device(config-vlan-7)# untagged ethernet 1/3/2
device(config-vlan-7)# pvlan type primary
device(config-vlan-7)# pvlan mapping 901 ethernet 1/3/2
device(config-vlan-7)# pvlan mapping 902 ethernet 1/3/2
device(config-vlan-7)# pvlan mapping 903 ethernet 1/3/2
```

Configuration Example for Implicit Dual-Mode PVLAN Network

To configure the implicit dual-mode PVLAN network, enter the following commands.

```
device(config)# vlan 101
device(config-vlan-101)# pvlan type isolated
device(config-vlan-101)# tagged ethernet 1/1/25 e 1/1/35
ports ethe 1/1/25 ethe 1/1/35  are removed from default VLAN 3999 to add to Private VLAN 101
Added tagged port(s) ethe 1/1/25 ethe 1/1/35 to port-vlan 101.
device(config-vlan-101)# spanning-tree
device(config-vlan-101)# exit

device(config)# vlan 102
device(config-vlan-102)# pvlan type community
device(config-vlan-102)# spanning-tree
device(config-vlan-102)# exit

device(config)# vlan 103
device(config-vlan-103)# pvlan type community
device(config-vlan-103)# spanning-tree
device(config-vlan-103)# exit

device(config)# vlan 100
device(config-vlan-100)# pvlan type primary
device(config-vlan-100)# tagged ethernet 1/1/20
ports ethe 1/1/20  are removed from default VLAN 3999 to add to Private VLAN 100
Added tagged port(s) ethe 1/1/20 to port-vlan 100.
device(config-vlan-100)# tag ethernet 1/1/21 e 1/1/31
ports ethe 1/1/21 ethe 1/1/31  are removed from default VLAN 3999 to add to Private VLAN 100
device(config-vlan-100)# pvlan mapping 101 ethernet 1/1/20
device(config-vlan-100)# pvlan mapping 103 ethernet 1/1/20
device(config-vlan-100)# pvlan pvlan-trunk 102 ethernet 1/1/21 ethernet 1/1/31
device(config-vlan-100)# pvlan pvlan-trunk 101 ethernet 1/1/21 ethernet 1/1/31
device(config-vlan-100)# pvlan pvlan-trunk 103 ethernet 1/1/21 ethernet 1/1/31
device(config-vlan-100)# spanning-tree
```

```
device(config-vlan-100)# exit

device(config)# vlan 201
device(config-vlan-201)# untagged ethernet 1/1/25 e 1/1/35
Added untagged port(s) ethe 1/1/25 ethe 1/1/35 to port-vlan 201
device(config-vlan-201)# pvlan type isolated
device(config-vlan-201)# spanning-tree
device(config-vlan-201)# exit

device(config)# vlan 202
device(config-vlan-202)# pvlan type community
device(config-vlan-202)# spanning-tree
device(config-vlan-202)# exit

device(config)# vlan 203
device(config-vlan-203)# pvlan type community
device(config-vlan-203)# spanning-tree
device(config-vlan-203)# exit

device(config-vlan-203)# pvlan type community
device(config-vlan-203)# spanning-tree
device(config-vlan-203)# exit

device(config)# vlan 200
device(config-vlan-200)# pvlan type primary
device(config-vlan-200)# untagged ethernet 1/1/20
Added untagged port(s) ethe 1/1/20 to port-vlan 200.
device(config-vlan-200)# tagged e 1/1/21 e 1/1/31
Added tagged port(s) ethe 1/1/21 ethe 1/1/31 to port-vlan 200.
device(config-vlan-200)# pvlan mapping 203 ethernet 1/1/20
device(config-vlan-200)# pvlan mapping 201 ethernet 1/1/20
device(config-vlan-200)# pvlan mapping 202 ethernet 1/1/20
device(config-vlan-200)# pvlan pvlan-trunk 203 ethernet 1/1/21 ethernet 1/1/31
device(config-vlan-200)# pvlan pvlan-trunk 201 ethernet 1/1/21 ethernet 1/1/31
device(config-vlan-200)# pvlan pvlan-trunk 202 ethernet 1/1/21 ethernet 1/1/31
device(config-vlan-200)# spanning-tree
device(config-vlan-200)# exit
```

Multiple Promiscuous Ports Support in Private VLANs

Promiscuous ports are member ports of a primary VLAN. Prior versions of the FastIron software supported only a single promiscuous port to be mapped to a secondary VLAN. However, now each secondary VLAN can associate with one or more promiscuous ports.

An isolated VLAN with member ports, when mapped to one or more promiscuous ports of the primary VLAN achieves the same forwarding behavior as that of the uplink ports in a port based VLAN (uplink-switch). The broadcast and unknown unicast traffic from a host (isolated) port is flooded to the uplink (promiscuous) ports only. Due to the hardware forwarding functionality of the private VLAN (in the supported stackable SKUs) this method of achieving the uplink port behavior achieves a better throughput than the conventional method of achieving uplink port.

Mapping Secondary VLAN to Primary VLAN by Multiple Promiscuous Ports

To map a secondary VLAN to primary VLAN through multiple promiscuous ports, follow these example steps:

1. Add the tagged ethernet ports to a VLAN.
2. Configure the VLAN as isolated.
3. Create VLAN 101.
4. Configure the VLAN 101 as primary VLAN in a PVLAN.
5. Map the isolated VLAN to the primary VLAN with uplink-ports as promiscuous ports. All broadcast and unknown-unicast traffic from isolated VLAN ports will be sent to only promiscuous ports.

In the following example configuration, the isolated VLAN 100 has multiple promiscuous ports 1/1/3 and 1/1/4.

```
device(config)# vlan 100
device(config-vlan-100)# pvlan type isolated
device(config-vlan-100)# tagged ethernet 1/1/1 to 1/1/2
device(config-vlan-100)# vlan 101
device(config-vlan-101)# pvlan type primary
device(config-vlan-101)# tagged ethernet 1/1/3 to 1/1/5
device(config-vlan-101)# pvlan map 100 ethernet 1/1/3
device(config-vlan-101)# pvlan map 100 ethernet 1/1/4
```

PVLAN Support Over LAG

LAG is supported for Promiscuous, Isolated and Community ports. Private VLAN support over LAG port enhances the bandwidth on promiscuous, ISL and host links and increases link reliability.

This enhanced LAG support provides LAG between access and aggregation switches. It also enables ICX switches to be used as uplink switch replacements with new and existing private VLAN deployments.

Following is a sample configuration.

```
device(config)# lag ISL dynamic id 1
device(config-lag-ISL)# ports ethe 1/2/3 ethe 1/2/4
LAG ISL deployed successfully!
device(config-lag-ISL)# !
device(config-lag-ISL)# lag promiscuous dynamic id 2
device(config-lag-promiscuous)# ports ethe 1/2/1 ethe 1/2/2
LAG promiscuous deployed successfully!
device(config-lag-promiscuous)# !
device(config-lag-promiscuous)# lag wallplate-1 dynamic id 3
device(config-lag-wallplate-1)# ports ethe 1/1/1 to 1/1/4
LAG wallplate-1 deployed successfully!
device(config-lag-wallplate-1)# !
device(config-lag-wallplate-1)# lag wallplate-2 dynamic id 4
device(config-lag-wallplate-2)# ports ethe 1/1/5 to 1/1/8
LAG wallplate-2 deployed successfully!
device(config-lag-wallplate-2)# !
device(config-lag-wallplate-2)#
device(config-lag-wallplate-2)#
device(config-lag-wallplate-2)#
device(config-lag-wallplate-2)# vlan 101 by port
device(config-vlan-101)# pvlan type isolated
device(config-vlan-101)# tagged lag 3 to 4
Added tagged port(s) lag 3 to 4 to port-vlan 101.
device(config-vlan-101)# !
device(config-vlan-101)# vlan 100 by port
device(config-vlan-100)# pvlan type primary
device(config-vlan-100)# tagged ethe 1/1/10 ethe 1/1/20 lag 1 to 2
Added tagged port(s) ethe 1/1/10 ethe 1/1/20 lag 1 to 2 to port-vlan 100.
device(config-vlan-100)# pvlan mapping  101 ethe 1/1/10 lag 1
device(config-vlan-100)# pvlan pvlan-trunk 101 ethe 1/1/20 lag 2
device(config-vlan-100)# !
```

The existing **pvlan mapping** and **pvlan-trunk** CLIs are modified to accommodate the PVLAN support over LAG. Please refer the *FastIron Command Reference Guide* for more information.

Displaying VLAN Information

After you configure the VLANs, you can verify the configuration using the various **show** commands.

The following example shows how VLANs are displayed in alphanumeric order.

```
device# show run
...
```

```
vlan 2 by port
...
vlan 10 by port
...
vlan 100 by port
...
```

The following example shows how to display system-wide VLAN information.

Use the **show vlan** command to display VLAN information for all the VLANs configured on the device.

```
device(config-vlan-4000)# show vlan 10
Total PORT-VLAN entries: 6
Maximum PORT-VLAN entries: 1024

PORT-VLAN 10, Name [None], Priority level0, On
 Untagged Ports: None
   Tagged Ports: (U1/M1)  48
   Tagged Ports: (U3/M1)  48
 Mac-Vlan Ports: None
     Monitoring: Disabled
```

The following example shows how to display global VLAN information.

```
device#  show vlan brief
System-max vlan Params: Max(4095) Default(1024) Current(3210)
Default vlan Id :1
Total Number of Vlan Configured :5
VLANs Configured :1 to 4 10
```

The following example shows how to display VLAN information for specific ports.

```
device# show vlan ethernet 1/7/1
Total PORT-VLAN entries: 3
Maximum PORT-VLAN entries: 8
legend: [S=Slot]
PORT-VLAN 100, Name [None], Priority level0, Spanning tree Off
 Untagged Ports: (S7)  1  2  3  4
   Tagged Ports: None
```

The following example shows how to display a port VLAN membership for a specific port on the device.

```
device# show vlan brief ethernet 1/1/5

Port 1/1/5 is a member of 3 VLANs
VLANs 5 101 4094
Untagged VLAN  : 1
Tagged   VLANs : 5 101 4094
```

NOTE

The untagged VLAN will show the system default VLAN ID even if the port is not part of it.

The following example shows how to display a port implicit dual-mode VLAN membership where the port is untagged in any of the VLAN.

```
device# show interfaces ethernet 7
GigabitEthernet7 is down, line protocol is down
Port down for 2 days 1 hour 40 minutes 5 seconds
  Hardware is GigabitEthernet, address is 0000.00a8.4706 (bia 0000.00a8.4706)
  Configured speed auto, actual unknown, configured duplex fdx, actual unknown
  Configured mdi mode AUTO, actual unknown
  Untagged member of L2 VLAN 3999, port state is BLOCKING
```

NOTE

The port up/down time is required only for physical ports and not for loopback, virtual ethernet (ve), and tunnel ports.

The following example shows how to display a port implicit dual-mode VLAN membership where the port is tagged (explicitly removed from default VLAN).

```
device# show interfaces ethernet 7
GigabitEthernet7 is down, line protocol is down
Port down for 2 days 1 hour 40 minutes 5 seconds
  Hardware is GigabitEthernet, address is 0000.00a8.4706 (bia 0000.00a8.4706)
  Configured speed auto, actual unknown, configured duplex fdx, actual unknown
  Configured mdi mode AUTO, actual unknown
  Tagged member of 1 L2 VLANs, port state is BLOCKING
```

The following example shows how to display a port implicit dual-mode VLAN membership where the port is tagged in some of the VLANs and untagged in any VLAN (implicit dual-mode).

```
device# show interfaces ethernet 7
GigabitEthernet7 is down, line protocol is down
Port down for 2 days 1 hour 40 minutes 5 seconds
  Hardware is GigabitEthernet, address is 0000.00a8.4706 (bia 0000.00a8.4706)
  Configured speed auto, actual unknown, configured duplex fdx, actual unknown
  Configured mdi mode AUTO, actual unknown
  Tagged member of 2 L2 VLANs, untagged in VLAN 3999, port state is BLOCKING
```

The following example shows how to display port default VLAN IDs (PVIDs).

```
device# show interfaces brief
Port  Link    State   Dupl Speed Trunk Tag Pvid Pri MAC            Name
1     Up      Forward Full 1G    None  No  1    0   0000.00a8.4700 a12345678901
2     Up      Forward Full 1G    None  Yes 1    0   0000.00a8.4701
3     Up      Forward Full 1G    None  Yes NA   0   0000.00a8.4702
4     Up      Forward Full 1G    None  Yes NA   0   0000.00a8.4703
5     Up      Forward Full 1G    None  No  2    0   0000.00a8.4704
6     Down    None    None None  None  Yes NA   0   0000.00a8.4705
7     Down    None    None None  None  Yes 4    0   0000.00a8.4706
8     Down    None    None None  None  Yes 4    0   0000.00a8.4707
9     Down    None    None None  None  Yes NA   0   0000.00a8.4708
10    Down    None    None None  None  Yes NA   0   0000.00a8.4709
```

The output of the **show interfaces brief** command lists the port default VLAN IDs (PVIDs) for each port. PVIDs are displayed as follows:

- For untagged ports, the PVID is the VLAN ID number.
- For an interface configured as untagged member in any VLAN, the PVID will be that VLAN ID. If the interface is explicitly removed from default VLAN (i.e. configured as purely tagged), then PVID is not applicable.

The following example shows how to display PVLAN configuration with respect to the primary VLAN and its associated secondary VLANs and to display the member ports, promiscuous ports, and the switch-switch link ports of a PVLAN.

```
device# show pvlan
PVLAN: primary VLAN  100
 Port 1/1/4 1/1/10 1/1/11
Community VLAN  102
 Port 1/1/1 1/1/2 1/1/10 1/1/11
 Promiscuous Port: 1/1/4
 Inter switch link Port: 1/1/10 1/1/11
 BpduGuard enabled Port: 1/1/1 1/1/2
Isolate   VLAN  101
 Port 1/1/3 1/1/10 1/1/11
 Promiscuous Port: 1/1/4
 Inter switch link Port: 1/1/10 1/1/11
 BpduGuard enabled Port: 1/1/1 1/1/2
```

NOTE

The **show pvlan** command is not supported on software forwarding platforms.

Layer 2 Trace Route Utility

The Layer 2 trace route utility traces the traffic path through a specified device in a VLAN. It can also be used to probe all reachable paths to all devices in the VLAN. The utility can do the following:

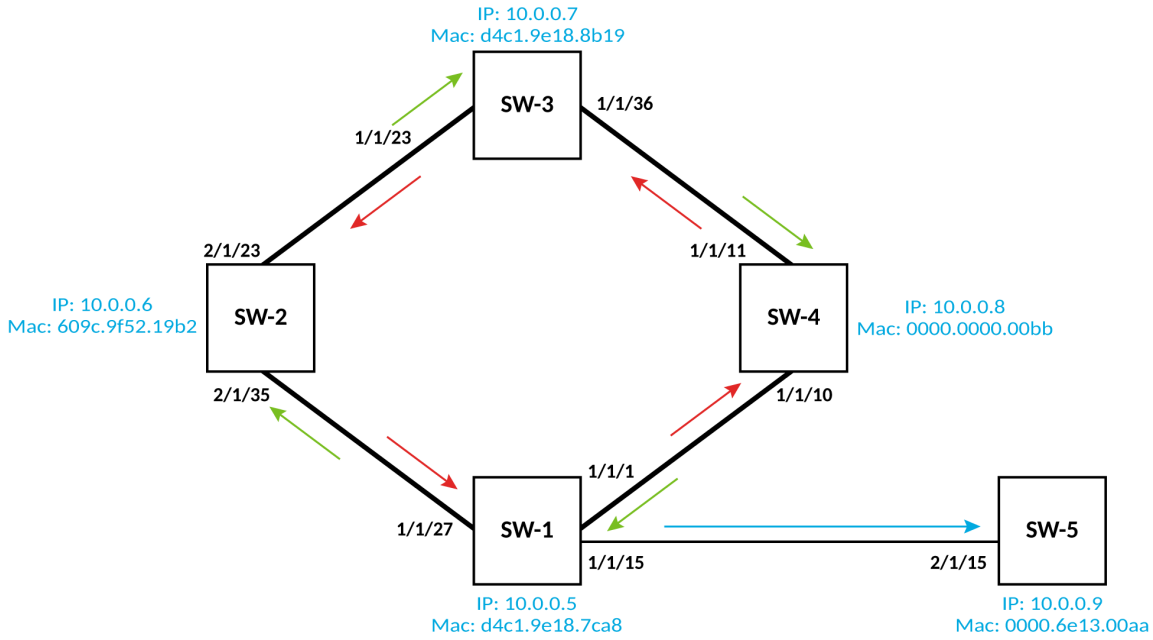
- Trace a specific IP, MAC, or hostname in a VLAN
- Probe the entire Layer 2 topology

The command output will show the following information:

- Input or output ports of each hop in the path
- Round trip travel time of each hop
- Hops in the VLAN that form a loop
- Each hop's Layer 2 protocol such as STP, RSTP, 802.1w, SSTP, metro ring, or route-only
- Hostname for each hop

The probed Layer 2 information is discarded when a new **trace-l2** command is issued.

FIGURE 104 Layer 2 Trace Path Example




```
device# trace-l2 vlan 1
device# trace-l2 show

*** Warning! The following 4 hops form a loop in vlan 1

hop   input  output IP and/or MAC address       microsec comment    hostname

  1                  d4c1.9e18.7ca8

  2   e2/1/35        609c.9f52.19b2

  3   e1/1/23        d4c1.9e18.8b19

  4   e1/1/11        0000.0000.00bb
```

```
Vlan 1 L2 topology was probed 6 sec ago, # of paths: 2
path 1 from e1/1/27, 3 hops:
  hop   input  output IP and/or MAC address       microsec comment    hostname
  1   e2/1/35 e2/1/23  10.0.0.6  609c.9f52.19b2     65611
  2   e1/1/23 e1/1/36  10.0.0.7  d4c1.9e18.8b19     64415
  3   e1/1/11          10.0.0.8  0000.0000.00bb     62042
path 2 from e1/1/1, 3 hops:
  hop   input  output IP and/or MAC address       microsec comment    hostname
  1   e1/1/10 e1/1/11  10.0.0.8  0000.0000.00bb     67943
  2   e1/1/36 e1/1/23  10.0.0.7  d4c1.9e18.8b19     6044
  3   e2/1/23          10.0.0.6  609c.9f52.19b2     1262
path 3 from e1/1/15, 1 hops:
  hop   input  output IP and/or MAC address       microsec comment    hostname
  1    e2/1/15         10.0.0.9  0000.6e13.00aa     67943
device#
```

Considerations and Limitations

- The Layer 2 trace route utility supports only Unicast destination MAC addresses.
- The results of this command are stored for no more than 10 minutes on a device or are replaced when a new **trace-l2** command is issued again.
- The Layer 2 trace route utility detects the IP as the IP of the virtual Ethernet (VE) interface, if available; else it will try to detect the IP of the Loopback interface.
- On a topology where loops are avoided by a loop avoidance protocol, the traced path will honor the loop-free network path.
- Devices and interfaces that do not support the Layer 2 trace route utility will act as transparent bridges and will be invisible to the **trace-l2** result.
- The destination for **trace-l2** operation must be a another ICX device and not an end host.
- At any point in time only one **trace-l2** operation can be running in a network.
- The Layer 2 trace route utility is not supported on breakout ports.
- The Layer 2 trace route utility results can discover up to 8 hops in a single path.
- The Layer 2 trace route utility operations takes at most one second to complete.
- To use the the Layer 2 trace route utility with a hostname, a DNS mapping with the hostname must be present in the DNS server.

VXLAN

VXLAN Gateway Overview

Virtual Extensible Local Area Network (VXLAN) is an overlay technology to create a logical Layer 2 network on top of a Layer 3 IP network.

NOTE
VXLAN is supported on RUCKUS ICX 7550, RUCKUS ICX 7650, and RUCKUS ICX 7850 devices.

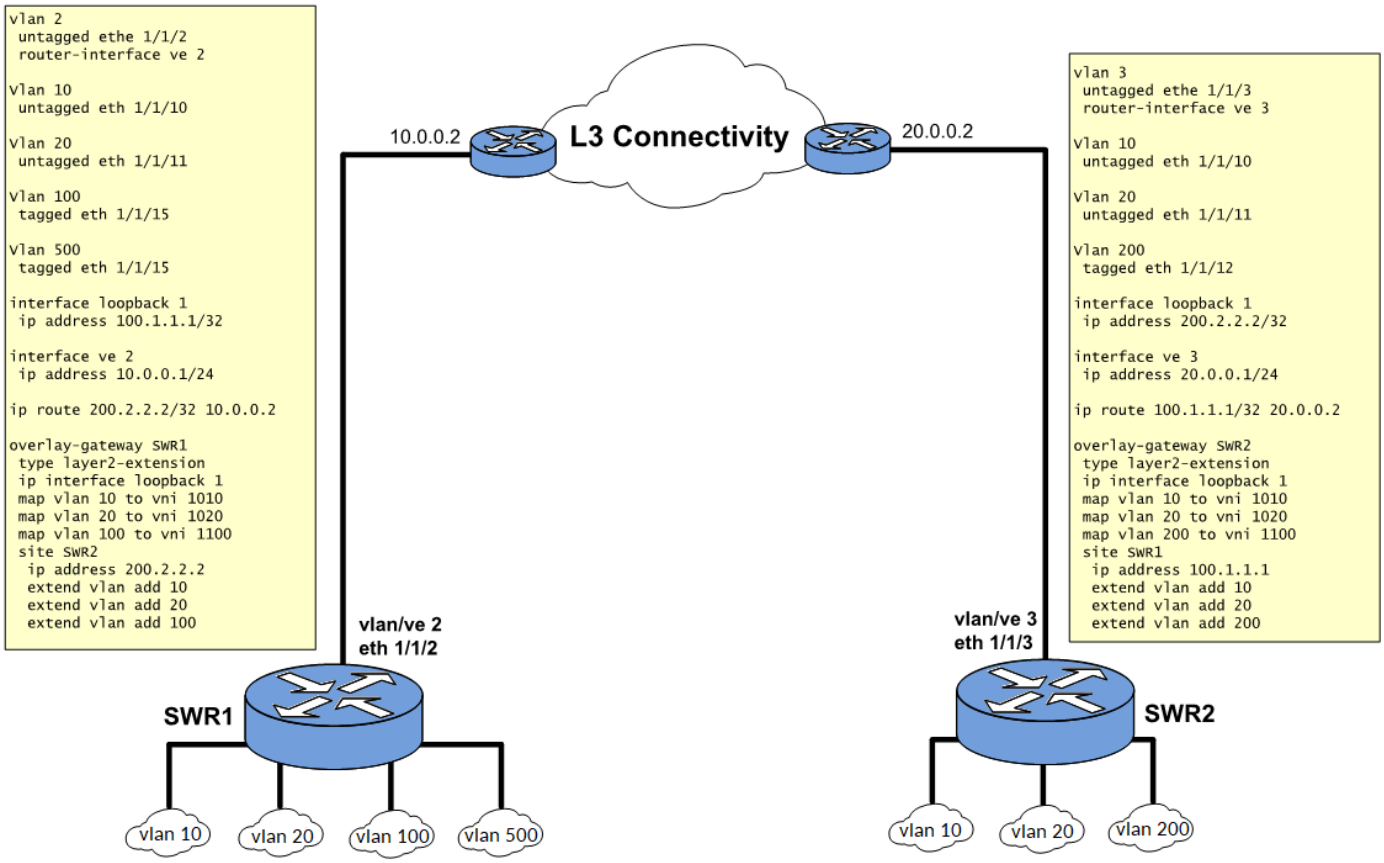
VXLAN is, with one exception, compliant with RFC 7348: *Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks*. The exception is that RFC 7348 discusses using multicast in the underlay (Layer 3) network for forwarding overlay (Layer 2) network Broadcast, Unknown Unicast, and Multicast (BUM) traffic. The FastIron VXLAN implementation uses another approach, referred to as "static-ingress replication," for forwarding overlay (Layer 2) network BUM traffic.

Addressing the need for overlay networks in Layer 2 and Layer 3 data center networks that support multi-tenant environments, VXLAN functions as a framework to create a Layer 2 logical network over the existing Layer 3 infrastructure. In this way, VXLAN addresses the scalability requirements of cloud computing.

VXLAN extends the VLAN address space by adding a 24-bit segment ID called a VXLAN Network Identifier (VNI) and enables 16 million VXLAN network segments. The VNI in each frame segregates individual logical networks, allowing millions of individual Layer 2 VXLAN segments to coexist on a common Layer 3 network. Each VLAN is mapped to a unique VNI to extend the Layer 2 VLAN segment to a remote location.

The following figure depicts how VXLAN gateways are used to provide Layer 2 connectivity between two switches separated by a Layer 3 network, so that the users connected to the same VLANs on both switches have the experience of being connected to the same Layer 2 network.

FIGURE 105 VXLAN Gateway

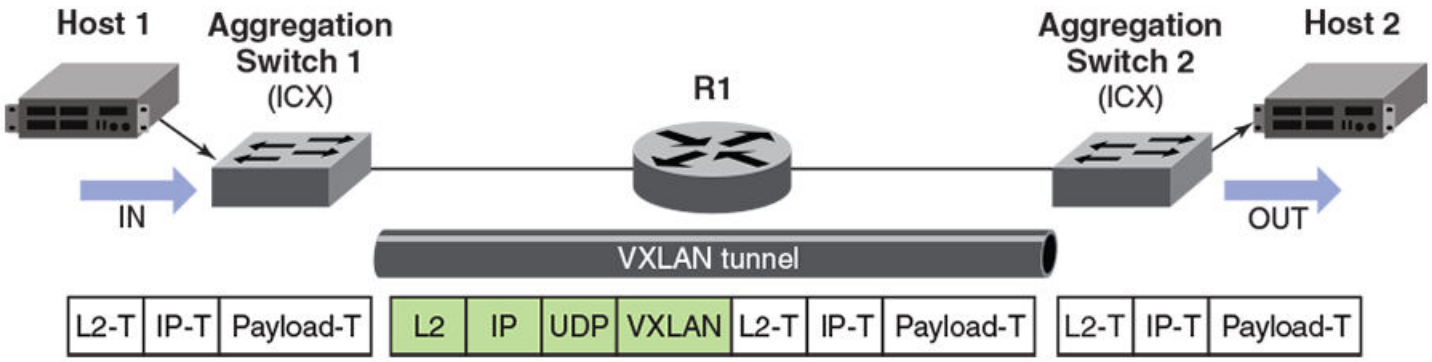


VXLAN Ethernet Frame Encapsulation

VXLAN uses a tunneling method to carry the Layer 2 overlay network traffic over the Layer 3 network. Communication is established between two tunnel endpoints called Virtual Tunnel End Points (VTEPs). VXLAN is a MAC Address-in-User Datagram Protocol (MAC-in-UDP) encapsulation, which encapsulates MAC frames at Layer 2 into a Layer 3 UDP header with an outer Ethernet header, outer IP header, outer UDP header, and VXLAN header. The outer IP header contains the corresponding source and destination VTEP IP addresses.

VTEPs are the nodes that provide the encapsulation and decapsulation functions and also map the tenant traffic to the virtual network and vice versa. The tenant's Layer 2 frame is encapsulated with the Layer 3 UDP header to send it to the remote location (VTEP). The remote end decapsulates the outer header and sends the original Layer 2 packet to the remote tenant.

FIGURE 106 VXLAN Tunneling



Because VXLAN is a tunneling technique, the VXLAN gateway is required to send traffic between VXLAN and a traditional VLAN. Using VXLAN gateway mode in aggregation-switch deployments, you can establish a tunnel at Layer 2 between two VXLAN gateways and extend the VLAN over an underlying Layer 3 infrastructure. The FastIron implementation of the VXLAN gateway allows communication between the VXLAN-aware world and the non-VXLAN-aware world. The FastIron VXLAN gateway provides E-LAN (multipoint-to-multipoint) service using a full mesh connectivity between VTEPs.

The FastIron implementation of VXLAN supports the following features:

- Multiple VNIs over the same VXLAN tunnel: RUCKUS supports the multiplex and demultiplex of multiple VNIs over the same VXLAN tunnel, allowing better scaling of the deployment.
- Multiple VXLAN tunnels: RUCKUS supports multiple VXLAN tunnels on a VTEP. These tunnels can be over the same or different uplink ports. However, multiple Layer 2 tunnels are supported with the following constraint:

  If two or more Layer 2 tunnels share the same outgoing interface (Layer 2 port), the outer Layer 2 header (Destination MAC, Source MAC, and VLAN header) must be the same for both tunnels. In other words, the Layer 3 outgoing interface, VRF, and the next-hop address must be the same for the two tunnels.
- VLAN translation across VXLAN segment: Because the VLAN tag is stripped from the Layer 2 (payload) frame before the frame is encapsulated with VXLAN tunnel headers, VXLAN can be used to interconnect the same Layer 2 subnet that is represented using different VLAN identifiers on each VTEP. For example, as shown in Figure 105, VLAN 100 on VXLAN gateway SWR1 and VLAN 200 on VXLAN gateway SWR2 are interconnected by mapping those VLANs to the same VNI (1100).
- Interoperation with other VXLAN implementations: The FastIron implementation of VXLAN can interoperate with the legacy VXLAN implementations of other vendors, as long as they use the IANA-assigned value of 4789 for the UDP destination port.
- VXLAN Routing In and Out of Tunnels (RIOT): The switch allows for associating a VE with a VLAN associated with a VNI.

  NOTE
  VXLAN RIOT is supported for RUCKUS ICX 7550 and ICX 7850 devices only.

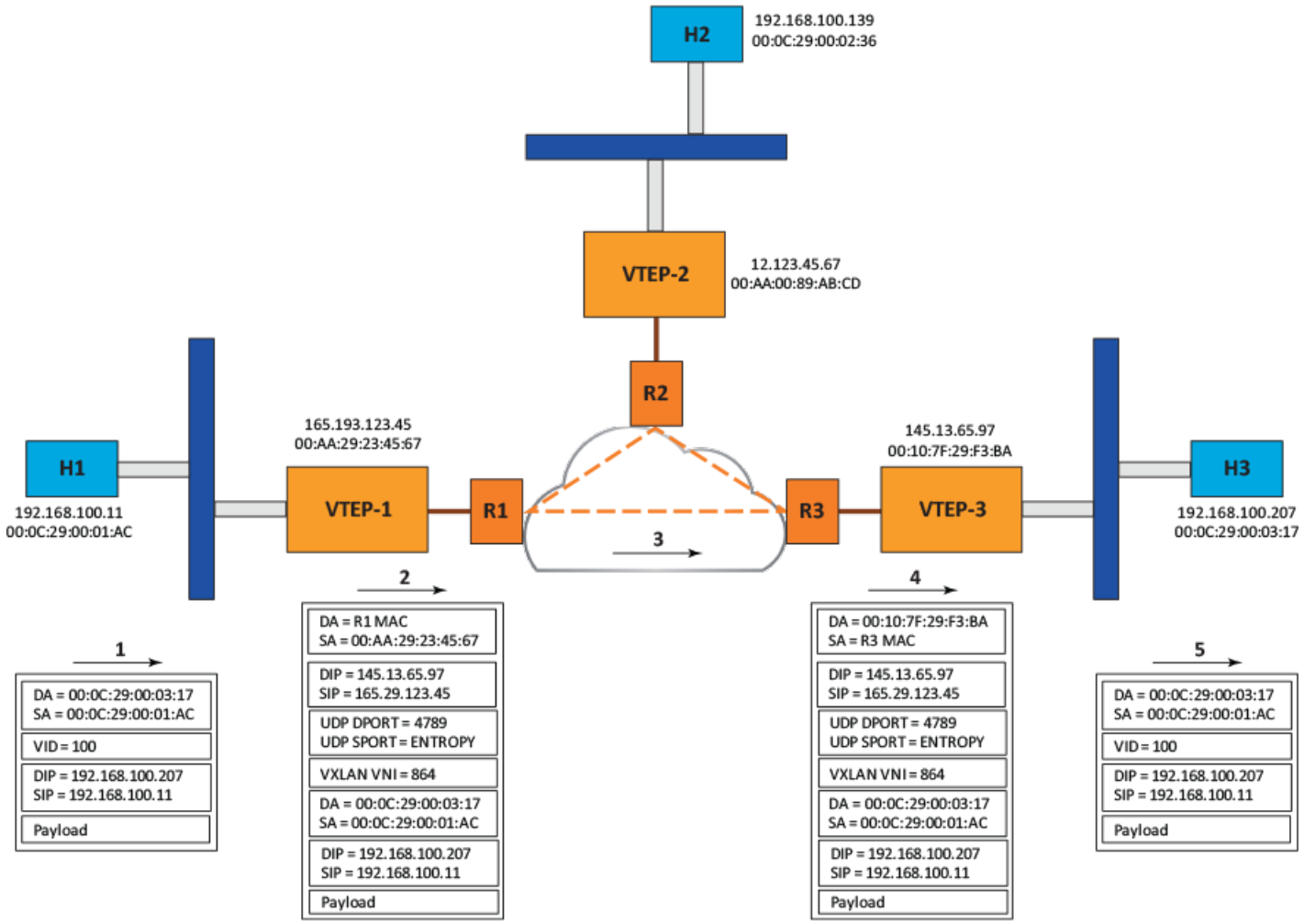
Unicast Forwarding in VXLAN Implementations

The following figure illustrates how unicast traffic is forwarded in a FastIron VXLAN implementation:

- VLAN 100 is mapped to VNI 864, and the VNI is extended to VTEP-1, VTEP-2, and VTEP-3.
- Host H1 in VLAN 100 on the access side of VTEP-1 sends a packet to Host H3 in VLAN 100 on the access side of VTEP-3.
- Host H1 knows the MAC address of Host H2 through ARP resolution.

- When VTEP-1 tries to forward the packet sent by H1 in VNI 864, it detects that the DMAC is reachable on the tunnel to VTEP-3. It then encapsulates the packet in the VXLAN tunnel to VTEP-3 and forwards it.
- When VTEP-3 receives the encapsulated packet, it removes the VXLAN tunnel header and forwards the payload (the inner packet) in VLAN 100 to Host H3.

FIGURE 107 VXLAN Unicast Traffic Forwarding Example



BUM Traffic Forwarding in VXLAN Implementations

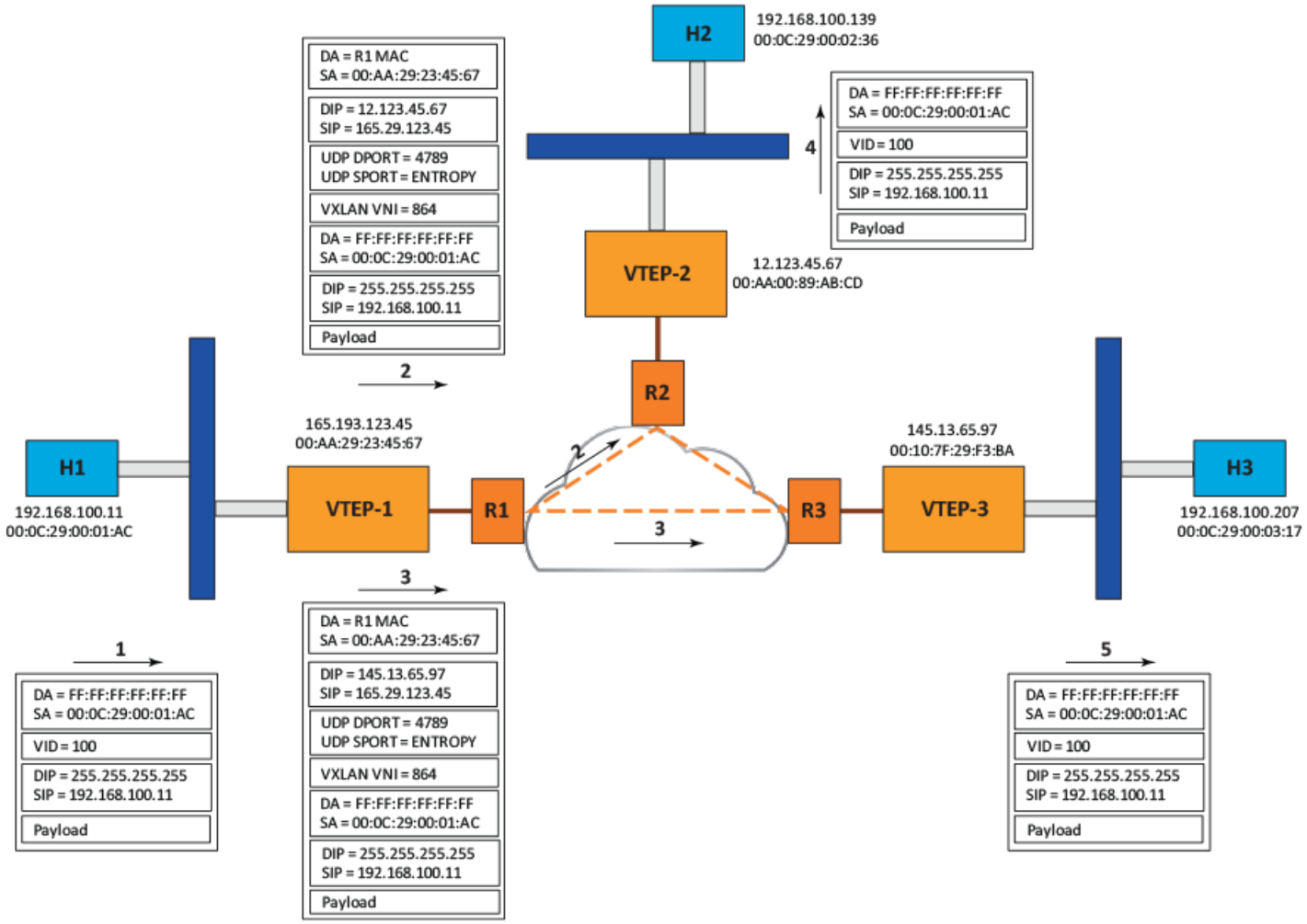
In a FastIron VXLAN implementation, multicast traffic is forwarded using the static-ingress replication method.

The following figure illustrates how multicast traffic is forwarded in a VXLAN implementation:

- VLAN 100 is mapped to VNI 864, and the VNI is extended to VTEP-1, VTEP-2, and VTEP-3.
- Host H1 in VLAN 100 on the access side of VTEP-1 sends a broadcast packet.
- When VTEP-1 tries to forward the packet sent by Host H1, it attempts to flood the broadcast packet in the VXLAN segment identified by VNI 864.

- As part of packet flooding to VNI 864, VTEP-1 encapsulates the packet in the VXLAN tunnel to each VTEP to which VNI 864 is extended (VTEP-2 and VTEP-3 in this case).
  - The Destination IP address (DIP) in the outer IP header is the unicast IP address.
  - The source VTEP does the replication.
- When VTEP-2 and VTEP-3 receive the encapsulated packet, they remove the VXLAN tunnel header and flood the payload (the inner packet) in VLAN 100.

FIGURE 108 VXLAN Multicast Traffic Forwarding Example



Inner Frame VLAN Tagging

In the VXLAN gateway, the encapsulating VTEP strips the inner VLAN tag of the packet before forwarding it to the remote VTEP. Upon reception, the remote VTEP decapsulates the packets, and a VLAN tag is assigned to the packet based on the one-to-one mapping between the VLAN and the VNI. The assignment of a VLAN tag also depends on whether the access port at the destination is tagged or untagged. If the access port is tagged, the VLAN tag is added after decapsulation and before the frame is sent. If the access port is untagged, an untagged frame is sent to the remote tenant.

Load Balancing Entropy

To enable a level of entropy for the ECMP/LAG load balancing of the VXLAN tunnel traffic across the VXLAN underlay, RFC 7348 recommends that the UDP source port number of the VXLAN tunneled packet be calculated using a hash value of the Layer 2 and Layer 3 headers of the passenger packet. The FastIron VXLAN gateway supports this approach.

MAC Learning

MAC learning is performed on both access and network ports. MAC lookup (or Layer 2 bridging) is also performed for traffic received on access or network ports.

Failure Detection and Redundant Remote Connections

You can configure keep-alive messages to detect loss of connection at the remote end of the VXLAN tunnel and redundant secondary IP addresses to restore the connection.

When you configure failure detection for a VXLAN site, keep-alive (ICMP echo) messages are sent to the remote end at a configured interval. If no response is received, the keep-alive message is resent as many times as configured for the associated retry count.

You can also configure more than one IP address for the remote site so that if the remote site does not respond to the keep-alive message within the specified time (interval multiplied by count), the next configured IP address in the configured series is tried. If a response is received from that IP address, it becomes the active connection for the remote site. One primary IP address and a maximum of 15 secondary IP addresses can be configured per VXLAN site.

NOTE
If all configured IP addresses are exhausted and all remote sites are unresponsive, the redundancy mechanism sets the keep-alive timer and retry count to their maximum values (20 seconds and 5 retries) and continues to loop through the IP address list checking for an ICMP reply from a remote VTEP. After an ICMP reply is received from a remote VTEP, the keep-alive timer and the retry count values are both reset to the values configured by the user.

NOTE
If an ICMP reply is received from another IP address after an active connection is established, the second response is dropped.

Quality of Service Support

The VXLAN-controlled tenant traffic has to compete with the tunneled data traffic. To ensure priority forwarding of VXLAN-controlled tenant traffic, Quality of Service (QoS) marking and remarking is supported. The Layer 2 User Priority and Layer 3 DSCP are copied from inner to outer headers during encapsulation, so that the transport Layer 2 and Layer 3 headers reflect the passenger Layer 2 and Layer 3 priority. For more information on DSCP remarking, refer to "DSCP Remarking Overview" in the *RUCKUS FastIron QoS and Traffic Management Configuration Guide*.

NOTE
Copying the Layer 2 User Priority from the encapsulating Layer 2 header into the outgoing Layer 2 header (after decapsulation) is not supported. Likewise, copying the Layer 3 DSCP from the encapsulating Layer 3 header into the outgoing Layer 3 header (after de-capsulation) is not supported. However, the DSCP carried in the passenger Layer 3 header is maintained across the VXLAN tunnel.

Unsupported Features

The following features are not supported in the FastIron VXLAN implementation:

- The ELINE version of VXLAN
- Auto-discovery of VTEPs
- Using multicast for forwarding BUM traffic in the underlay network
- Path MTU discovery on VXLAN tunnels
- Forwarding of Layer 2 BPDUs (such as STP and LACP) over VXLAN tunnels

VXLAN Configuration Considerations

- Only one overlay gateway is supported.
- Only one-to-one mapping is allowed between a VLAN and a VNI.
- The default VLAN cannot be mapped to a VNI.
- No ports on the VLAN mapped to a VNI can have an IP address configured.
- A VNI can be carried by one or more VXLAN tunnels.
- Mapping a range of VLANs to a VNI for a VXLAN overlay gateway is supported. Consider the following limitations when mapping a range of VLANs to a VNI for a VXLAN overlay gateway:
  - For RUCKUS ICX 7650 and ICX 7850 devices, no more than two access ports per VLAN should be configured. 4080 VLAN to VNI mappings are supported. To support 4080 VLAN to VNI mappings, the forwarding profile must be changed from profile 1 (the default) to profile 2. Refer to the **forwarding-profile** command in the *RUCKUS FastIron Command Reference* Guide and the "Configuration Fundamentals" chapter in the *RUCKUS FastIron Management Configuration Guide* for more information.
  - For RUCKUS ICX 7550 devices, no more than two access ports per VLAN should be configured. 3072 VLAN to VNI mappings are supported when a maximum of two access ports is configured for each VXLAN-mapped VLAN. 4080 VLAN to VNI mappings are supported when a maximum of one access port is configured for each VXLAN-mapped VLAN. To support these scaling numbers, the forwarding profile must be changed from profile 1 (the default) to profile 2.
- Extending a range of mapped VLANs over a VXLAN overlay gateway is supported.
- An interface must be configured for which the IP address is to be used as a source for the VxLAN tunnels on VxLAN Gateway. Only loopback and VE interfaces are supported.
- The underlay (or transport) network cannot belong to a user VRF. The source interface must be on default VRF.
- VXLAN Routing In and Out of VXLAN Tunnels (RIOT) is supported for RUCKUS ICX 7550 and ICX 7850 devices only. A VLAN with a VE configuration can be mapped to a VNI. Also, a VE configuration is allowed on a VLAN mapped to a VNI.
- VXLAN encapsulation adds approximately 50 bytes of overhead to the MAC frame, which may cause frames to be rejected if the Maximum Transmission Unit (MTU) on the transport network cannot accommodate the extra bytes. Therefore, the MTU on the transport network port must be configured with a value greater than 1550 (1500 + 50) bytes. (The typical host MTU is 1500 bytes.)
- Jumbo frame support in the transport network is required if the overlay applications use a frame size larger than 1500 bytes.
- A LAG or Ethernet port on the VTEP that is connected to the VXLAN underlay (transport) network cannot be a route-only interface. In other words, route-only cannot be enabled on a LAG or physical port on which remote VTEPs are reachable.
- If multiple VTEPs are reachable through the same Ethernet or LAG port, the next-hop information (port or LAG ID, MAC address, VLAN ID) to reach the VTEPs must be the same.
- Because the static-ingress replication method is used to send BUM traffic over VXLAN tunnels, all VTEPs must be provisioned in full-mesh mode as far as VLAN extension is concerned.

Scaling Considerations

- For RUCKUS ICX 7650 and ICX 7850 devices, a maximum of 4080 VLANS can be mapped to VNIs when no more than two access ports per VLAN are configured.
- RUCKUS ICX 7550 devices, a maximum of 4080 VLANs can be mapped to VNIs when no more than one access port is configured. A maximum of 3072 VLANs can be mapped to VNIs when no more than two access ports are configured.
- A maximum of 32 remote sites (VXLAN tunnels) can be configured.
- A maximum of 16 IPv4 addresses can be configured for each VXLAN remote site (VTEP); that is, one primary IP address and up to 15 secondary IP addresses.
- The total number of Ethernet or LAG ports in all mapped VLANs cannot exceed 8000.

Protocol Considerations

- VXLAN remote sites support IPv4 addressing only.
- A VLAN with multicast snooping enabled cannot be mapped to a VNI. Likewise, you cannot enable multicast snooping on a VLAN that is mapped to a VNI. You can configure the **no multicast active** or the **no multicast passive** command to disable multicast snooping on a VLAN. If multicast snooping is enabled globally, you can configure the **no ip multicast active**, **no ip multicast passive**, **no ipv6 multicast active**, or **no ipv6 multicast passive** command to disable global multicast snooping.
- VXLAN is not supported on MCT cluster devices.
- VXLAN RIOT is disabled by default. To enable a VE interface on a VXLAN-mapped VLAN, VXLAN RIOT must be enabled first. Refer to VXLAN RIOT on page 355.
- OSPF (and VXLAN RIOT in general) is supported only on overlay VE interfaces. An untagged Ethernet or LAG interface in a mapped VLAN cannot be used as an overlay interface.

VXLAN Feature Support

TABLE 26 VXLAN Feature Support

Functionality on VXLAN-Enabled VLAN	Support on VXLAN Access Port	Support on VXLAN Network Port	Comments
MAC FDB	Yes	Yes	
MAC learning disable	Yes	No	
MAC learning rate control	No	No	Feature not supported on FastIron devices.
Flow-based MAC address learning	No	No	Feature not supported on FastIron devices.
MAC address move notification	Yes	No	
MAC notification traps	Yes	No	No MIB for VXLAN.
Port-based VLANs - Tagged - Untagged	Yes	N/A	
Default VLAN	No	Yes	VXLAN cannot be enabled on the default VLAN.
Static MAC	Yes	No	
Static MMAC	No	No	
Port MAC security	Yes	No	

TABLE 26 VXLAN Feature Support (continued)

Functionality on VXLAN-Enabled VLAN	Support on VXLAN Access Port	Support on VXLAN Network Port	Comments
Link aggregation - Static LAG - LACP - Keep-alive LAG	Yes	N/A	Layer 3 connection to underlay network can be single port, LAG, ECMP.
LAG Hardware Failover	Yes	N/A	
VXLAN Tunnel Keep-Alive	N/A	Yes	
802.1D Spanning Tree	Yes	N/A	
802.1s Multiple Spanning Tree	Yes	N/A	
802.1W Rapid Spanning Tree Protocol (RSTP)	Yes	N/A	
PVST/PVST+ compatibility	Yes	N/A	
SAV/Q-in-Q	N/A	N/A	No Q-in-VNI support at this time.
Topology Group	Yes	N/A	
VLAN Group	Yes	N/A	
VLAN Range	Yes	N/A	
Metro Ring Protocol (MRP)	No	N/A	
Private VLAN	No	N/A	
UDLD	No	N/A	
VSRP	No	N/A	
Loop Detect	No	N/A	
MCT	No	N/A	
LOAM	No	N/A	
Ethernet remote loopback	No	N/A	

TABLE 27 Layer 3 Feature Support

Layer 3 Features and Capabilities	Support on VXLAN
Routing on VXLAN-enabled VLAN	Yes with RIOT
ARP	Yes with RIOT
Overlay ICMP Ping	Yes
Routing Protocols on Overlay	Only OSPF over Overlay is supported.
Static Routes on Overlay	Yes
IPv6 ND	No
Layer 3 Routing Protocol Packets	No

TABLE 28 Security Feature Support

Security Features	Support on VXLAN-Enabled VLAN
802.1x	No

Routing Functionality Using a Two-Device Configuration

If an ICX device is not configured with VXLAN RIOT enabled on the device, to achieve equivalent routing functionality, a two-device router configuration is possible. In case an ICX device acting as the VTEP does not provide routing for VLANs mapped to VNIs, a second device can be configured to provide routing for these VLANs. The following figure illustrates this configuration:

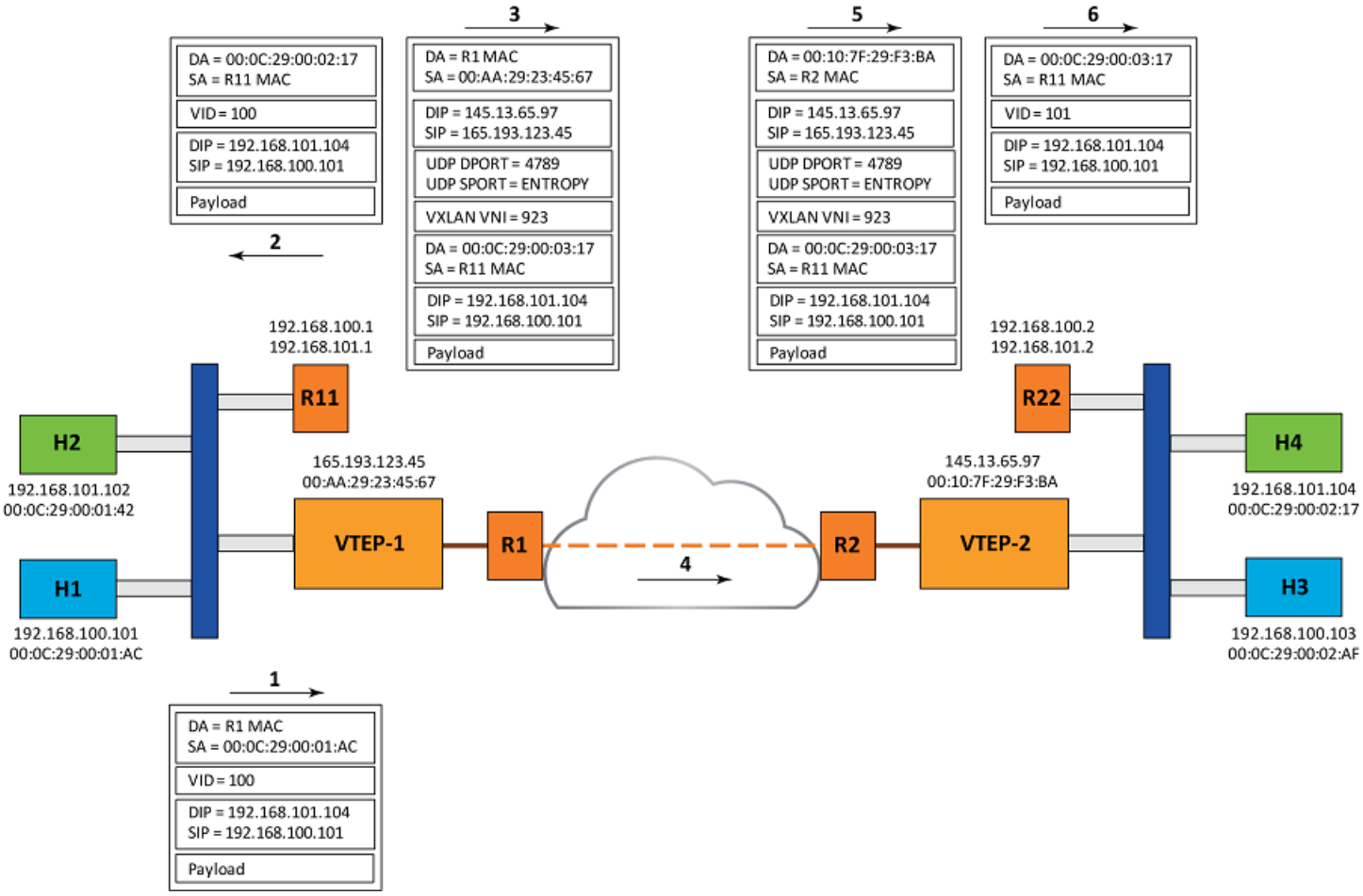
- VLANs 100 and 101 are mapped to VNIs 864 and 923, respectively.
- Devices R11 and R12 act as the default gateway for hosts in VLAN 100 and VLAN 101 on the access side of VTEP-1 and VTEP-2, respectively.
- Host H1 in VLAN 100 on the access side of VTEP-1 sends a packet to Host H4 in VLAN 101 on the access side of VTEP-2.

  For the packet, the SMAC is the H1 MAC, and the DMAC is the R11 MAC.
- R11 routes the packet and forwards it in VLAN 101.

  Here, the packet SMAC is the R11 MAC, and the DMAC is the H4 MAC.

  The packet VLAN is changed from 100 to 101.
- When VTEP-1 tries to forward the packet sent by R11 in VNI 923, it detects that the DMAC is reachable on the tunnel to VTEP-2. It then encapsulates the packet in the VXLAN tunnel to VTEP-2 and forwards it.
- When VTEP-2 receives the encapsulated packet, it removes the VXLAN tunnel header and forwards the payload (the inner packet) in VLAN 101 to Host H4.

FIGURE 109 Dual-Device Configuration for VXLAN RIOT Example



Configuring VXLAN

Configuring a VXLAN gateway involves the following tasks.

1. Configure a VXLAN gateway record.
2. Set the VXLAN gateway type as a Layer 2 extension.
3. Specify the loopback or VE interface with the IP address that will be used as the source IP address for VXLAN tunnels.

   NOTE
   The interface must not belong to a user VRF. Be sure the specified IPv4 address is configured on this interface; otherwise, all VXLAN tunnels will be down.

4. Map previously created VLANs, or a range of VLANs, to VXLAN Network Identifiers (VNIs).

5. Configure one or more remote site records.
   - For each remote site, specify the primary IP address of the remote site VTEP and up to 15 secondary IP addresses. The primary IP address is used as the destination address for the VXLAN tunnel to that remote site. If the connection to the primary IP address fails and failure detection is enabled, the secondary IP addresses are used in the configured order based on the assigned index provided.
   - Make sure the remote VTEP IP addresses are reachable in the default VRF; otherwise, the VXLAN tunnel to the remote VTEP will be down.
6. Configure failure detection.
   - Configure the interval (in seconds) at which a keep-alive message will be sent to the remote site (VTEP).
   - Configure the number of times (1 through 5) the keep-alive message is resent if no response is received.
7. Extend each mapped VLAN, or range of VLANs, to the remote site.

Configure a VXLAN Overlay Gateway

Complete the following steps to configure a VXLAN overlay gateway.

NOTE
Only one overlay gateway can be configured.

1. Name the overlay gateway and enter overlay-gateway configuration mode.

```
device# configure terminal
device(config)# overlay-gateway gate1
device(config-overlay-gw-gate1)#
```

2. Set the overlay gateway type as a Layer 2 extension.

```
device(config-overlay-gw-gate1)# type layer2-extension
```

3. Use the **ip interface** command in overlay-gateway configuration mode to configure the IP interface for the overlay gateway.

   NOTE
   You must configure the source interface, rather than an explicit IPv4 address, as the source address.

```
device(config-overlay-gw-gate1)# ip interface loopback 1
```

4. Map a VLAN you intend to extend over the VXLAN segment (that is, over the overlay gateway) to a VXLAN Network Identifier (VNI).

   NOTE
   A maximum of 256 VLAN to VNI pairs can be configured.

   NOTE
   The default VLAN (typically, VLAN 1) cannot be mapped to a VNI or extended over a VXLAN segment.

```
device(config-overlay-gw-gate1)# map vlan 2 to vni 3
```

5. Create a remote site (VXLAN tunnel) and configure the IP addresses for the site.

   NOTE
   Only IPv4 tunnels are supported.

   NOTE
   A maximum of 32 remote sites can be created.

   NOTE
   A maximum of 16 IP addresses; that is, one primary address and up to 15 secondary addresses can be configured per remote site. If the primary address fails, the secondary addresses will be tried in succession, based on the index number configured for each address. If a number has already been assigned to the IP address you configure, an error message is displayed. You must remove the existing configuration before you can reuse a configured number.

```
device(config-overlay-gw-gate1)# site site1
device(config-overlay-gw-gate1-site1)# ip address 67.67.67.1 primary
device(config-overlay-gw-gate1-site1)# ip address 67.67.67.2 secondary 1
device(config-overlay-gw-gate1-site1)# ip address 67.68.0.1 secondary 2
```

6. Configure failure detection for the remote site. The allowable keep-alive interval is 1 through 20 seconds. The retry count can be configured as 1 through 5.

```
device(config-overlay-gw-gate1-site1)# failure-detection keep-alive 20 retry 5
```

7. Extend the mapped VLAN to the remote site.

```
device(config-overlay-gw-gate1-site1)# extend vlan add 2
```

The following example configures overlay gateway gate1. VLAN 2 maps to VNI 3. A VXLAN tunnel is configured by creating a remote site (site1) and configuring its IP address (67.67.67.1) as the destination address. Redundant addresses and failure detection are configured for the remote site. The mapped VLAN (VLAN 2) is extended over the overlay gateway.

```
device# configure terminal
device(config)# overlay-gateway gate1
device(config-overlay-gw-gate1)# type layer2-extension
device(config-overlay-gw-gate1)# ip interface loopback 1
device(config-overlay-gw-gate1)# map vlan 2 to vni 3
device(config-overlay-gw-gate1)# site site1
device(config-overlay-gw-gate1-site1)# ip address 67.67.67.1 primary
device(config-overlay-gw-gate1-site1)# ip address 67.67.67.2 secondary 1
device(config-overlay-gw-gate1-site1)# ip address 67.68.0.1 secondary 2
device(config-overlay-gw-gate1-site1)# failure-detection keep-alive 20 retry 5
device(config-overlay-gw-gate1-site1)# extend vlan add 2
device(config-overlay-gw-gate1-site1)# end

device# show running-config overlay-gateway
overlay-gateway gateway1
 type layer2-extension
 ip interface Loopback 1
 map vlan 2 vni 3
 site site1
   ip address 67.67.67.1 primary
   ip address 67.67.67.2 secondary 1
   ip address 67.68.68.1 secondary 2
   extend vlan add 2

device# show overlay-gateway
Overlay Gateway Name    : gateway1
Type                    : layer2-extension
Source IP Interface     : loopback 1 (vrf: default-vrf, IP address: 32.32.31.13)
Total Mapped Vlans      : 1
Total Sites             : 1
```

The following example configures overlay gateway gate1. A range of VLANs (VLAN 200 through VLAN 300) maps to VNI 20000 and starts the VNI. A VXLAN tunnel is configured by creating a remote site (s1). The mapped VLANs (VLAN 200 through VLAN 300) are extended over the overlay gateway.

```
device# configure terminal
device(config)# overlay-gateway gate1
device(config-overlay-gw-gate1)# type layer2-extension
device(config-overlay-gw-gate1)# ip interface loopback 1
device(config-overlay-gw-gate1)# map vlan-range 200 to 300 vni-start 20000
device(config-overlay-gw-gate1)# site s1
device(config-overlay-gw-v1-site-s1)# extend vlan-range add 200 to 300

device# show overlay-gateway
Current configuration for overlay-gateway:
overlay-gateway v1
  type layer2-extension
  ip interface loopback 1
  map vlan-range 200 to 300 vni-start 20000
   map vlan 400 to vni 20400
   site s1
     extend vlan-range add 200 to 300
```

Gathering VXLAN Statistics

You can collect and control certain statistics pertaining to VXLAN virtual ports and VXLAN Network Identifiers (VNIs). The following task enables detailed statistics collection for a VNI.

NOTE
Statistics can be gathered for a maximum of 10 VNIs at a time.

Complete the following steps to gather VXLAN statistics.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Name the overlay gateway and enter overlay-gateway configuration mode.

```
device(config)# overlay-gateway gate1
device(config-overlay-gw-gate1)#
```

3. Enable VNI counters for the VXLAN overlay gateway and enable Virtual Forwarding Instance (VFI) counters for the VNIs. Also, collect statistics from Access Virtual Ports.

```
device(config-overlay-gw-gate1)# vni-counters vni 30000 stats detailed
```

The following example enables VNI counters for a VXLAN overlay gateway. It also enables Virtual Forwarding Instance (VFI) counters for the VNIs. Statistics from Access Virtual Ports are also collected. Statistics are then displayed using the **show overlay-gateway** command.

```
device# configure terminal
device(config)# overlay-gateway gate1
device(config-overlay-gw-gate1)# vni-counters vni 30000 stats detailed

device# show overlay-gateway 1 vnid 30000 counters

------ The VNI Counter Stats ------
     InUcastPkts: 30                          InUcastBytes: 4590
     InMcastPkts: 0                           InMcastBytes: 0
     InBcastPkts: 0                           InBcastBytes: 0

     OutKnownUcastPkts: 38                    OutKnownUcastBytes: 4590
     OutBumPkts       : 0                     OutBumBytes       : 0

Bum Traffic consists Unknown Ucast, Mcast, Bcast Traffic
```

Clearing VXLAN Statistics

You can clear statistics for an entire VXLAN gateway tunnel or for a particular VXLAN Network Identifier (VNI).

Use any of the following commands, as required, to clear VXLAN statistics. The commands do not need to be entered in the specified order. Using these commands is optional and they can be entered in any order. For more information on the full list of commands, refer to the *RUCKUS FastIron Command Reference*

1. Clear the network counters for a site.

```
device# clear overlay-gateway 1 site mm2-site network-counters
Cleared the counters for the site !

device# show overlay-gateway 1 site mm2-site network-counters
------ The Network Counter Stats ------
     InUcastPkts: 0                           InUcastBytes: 0
     InMcastPkts: 0                           InMcastBytes: 0
     InBcastPkts: 0                           InBcastBytes: 0

     OutKnownUcastPkts: 0                     OutKnownUcastBytes: 0
     OutBumPkts       : 0                     OutBumBytes       : 0

Bum Traffic consists Unknown Ucast, Mcast, Bcast Traffic
```

2. Clear the VNI counters for an overlay gateway.

```
device# clear overlay-gateway 1 vnid 30000 counters
Cleared the counters for VNI !

device# show overlay-gateway 1 vnid 30000 counters
------ The VNI Counter Stats ------
     InUcastPkts: 0                           InUcastBytes: 0
     InMcastPkts: 0                           InMcastBytes: 0
     InBcastPkts: 0                           InBcastBytes: 0

     OutKnownUcastPkts: 0                     OutKnownUcastBytes: 0
     OutBumPkts       : 0                     OutBumBytes       : 0

Bum Traffic consists Unknown Ucast, Mcast, Bcast Traffic
```

3. Clear the access counters for an Ethernet interface.

```
device# clear overlay-gateway 1 vnid 30000 ethernet 2/1/17 access-counters
Cleared the counters for the access counters !

device# show overlay-gateway 1 vnid 30000 ethernet 2/1/17 access-counters

------ The Access Counter Stats ------
     InUcastPkts: 0                          InUcastBytes: 0
     InMcastPkts: 0                          InMcastBytes: 0
     InBcastPkts: 0                          InBcastBytes: 0

     OutKnownUcastPkts: 0                    OutKnownUcastBytes: 0
     OutBumPkts       : 0                    OutBumBytes       : 0

Bum Traffic consists Unknown Ucast, Mcast, Bcast Traffic
```

4. Clear the access counters for a LAG interface.

```
device# clear overlay-gateway 1 vnid 30000 lag 2 access-counters
Cleared the counters for the access counters !

device# show overlay-gateway 1 vnid 30000 lag 2 access-counters

------ The Access Counter Stats ------
     InUcastPkts: 0                          InUcastBytes: 0
     InMcastPkts: 0                          InMcastBytes: 0
     InBcastPkts: 0                          InBcastBytes: 0

     OutKnownUcastPkts: 0                    OutKnownUcastBytes: 0
     OutBumPkts       : 0                    OutBumBytes       : 0

Bum Traffic consists Unknown Ucast, Mcast, Bcast Traffic
```

The following example clears the network counters for a site.

```
device# clear overlay-gateway 1 site mm2-site network-counters
Cleared the counters for the site !

device# show overlay-gateway 1 site mm2-site network-counters
------ The Network Counter Stats ------
     InUcastPkts: 0                          InUcastBytes: 0
     InMcastPkts: 0                          InMcastBytes: 0
     InBcastPkts: 0                          InBcastBytes: 0

     OutKnownUcastPkts: 0                    OutKnownUcastBytes: 0
     OutBumPkts       : 0                    OutBumBytes       : 0

Bum Traffic consists Unknown Ucast, Mcast, Bcast Traffic
```

The following example clears the VNI counters for an overlay gateway.

```
device# clear overlay-gateway 1 vnid 30000 counters
Cleared the counters for VNI !

device# show overlay-gateway 1 vnid 30000 counters
------ The VNI Counter Stats ------
     InUcastPkts: 0                          InUcastBytes: 0
     InMcastPkts: 0                          InMcastBytes: 0
     InBcastPkts: 0                          InBcastBytes: 0

     OutKnownUcastPkts: 0                    OutKnownUcastBytes: 0
     OutBumPkts       : 0                    OutBumBytes       : 0

Bum Traffic consists Unknown Ucast, Mcast, Bcast Traffic
```

The following example clears the access counters for an Ethernet interface.

```
device# clear overlay-gateway 1 vnid 30000 ethernet 2/1/17 access-counters
Cleared the counters for the access counters !

device# show overlay-gateway 1 vnid 30000 ethernet 2/1/17 access-counters

------ The Access Counter Stats ------
     InUcastPkts: 0                          InUcastBytes: 0
     InMcastPkts: 0                          InMcastBytes: 0
     InBcastPkts: 0                          InBcastBytes: 0

     OutKnownUcastPkts: 0                    OutKnownUcastBytes: 0
     OutBumPkts       : 0                    OutBumBytes       : 0

Bum Traffic consists Unknown Ucast, Mcast, Bcast Traffic
```

The following example clears the access counters for a LAG interface.

```
device# clear overlay-gateway 1 vnid 30000 lag 2 access-counters
Cleared the counters for the access counters !

device# show overlay-gateway 1 vnid 30000 lag 2 access-counters

------ The Access Counter Stats ------
     InUcastPkts: 0                          InUcastBytes: 0
     InMcastPkts: 0                          InMcastBytes: 0
     InBcastPkts: 0                          InBcastBytes: 0

     OutKnownUcastPkts: 0                    OutKnownUcastBytes: 0
     OutBumPkts       : 0                    OutBumBytes       : 0

Bum Traffic consists Unknown Ucast, Mcast, Bcast Traffic
```

Displaying VXLAN Information

You can display information on VXLAN gateways, connections, and tunnels.

Use the following commands to view VXLAN information. Using these commands is optional, and they can be entered in any order. For more information on the full list of commands, refer to the *RUCKUS FastIron Command Reference*.

1. To display information on the VXLAN running configuration, enter the **show running-config overlay-gateway** command.

```
device# show running-config overlay-gateway
overlay-gateway sanjose
 type layer2-extension
 ip interface loopback 1
 map vlan 101 to vni 25838
 map vlan 102 to vni 67924
 site denver
  ip address 2.2.2.2
  ip address 3.3.3.1 secondary index 2
  ip address 4.4.4.1 secondary index 3
  ip address 5.5.5.1 secondary index 5
  failure-detection keep-alive 10 retry 4
  extend vlan add 101
  extend vlan add 102
!
!
```

   The example shows that two VLANs are configured and extended to the remote site denver, which has a primary IP address of 2.2.2.2 and three secondary IP addresses. Information on failure detection is present because the option is enabled.

2. To display brief information on all VXLAN gateways, enter the **show overlay-gateway** command. To display brief information about a specific VXLAN gateway, enter the **show overlay-gateway** command followed by the name of the gateway, as shown in the following example.

```
device# show overlay-gateway sanjose

Overlay Gateway Name    : sanjose
Type                    : layer2-extension
Source IP Interface     : loopback 1 (vrf: default-vrf, IP address: 7.7.7.7)
Total Mapped Vlans      : 2
Total Sites             : 1
```

   The example displays brief information for the overlay gateway sanjose. Information on the VRF and configured IP address is included.

3. To display detailed information for a particular overlay gateway, enter the **show overlay-gateway** command, the name of the overlay gateway, and the **detail** keyword, as shown in the following example.

```
device# show overlay-gateway gt-1 detail

Overlay Gateway Name    : gt-1
Type                    : layer2-extension
Source IP Interface     : loopback 1 (vrf: default-vrf, IP address: 10.10.10.1)
Total Mapped Vlans      : 1
Total Sites             : 2
#    VLAN-ID VN-ID    VFI     Access-Port Extended-Site Cross-Connect
-    ------- -----    ---     ----------- ------------- -------------
1    100     10000    100         2           2        DISABLED

#    SiteName               Active IP-Address    Status     Monitoring   Connection Ext-Vlans
-    --------               -----------------    ------     ----------   ---------- ---------
1    mm1-site               20.20.20.1           Up         DISABLED     N/A        (100)
2    mm2-site               70.70.70.1           Up         DISABLED     N/A        (100)
```

   The example displays detailed output for the overlay gateway gt-1, including the following information:

   - For each mapped VLAN, the VNI and the Virtual Forwarding Instance (VFI) (the internal Layer 2 VPN identifier, which is the same as the VLAN identifier), along with the number of access (local) ports belonging to that VLAN and the number of sites to which this mapped VLAN is extended
   - For each site, details such as the site name, the site IP address, the status of the VXLAN tunnel to that site, and the VLANs that are extended to the site

4. To display information on a particular VLAN, enter the **show overlay-gateway** command followed by the overlay gateway name, the **vlan** keyword, and the VLAN number, as shown in the following example.

```
device# show overlay-gateway sanjose vlan 101
Overlay Gateway Name    : sanjose
#    VLAN-ID    VN-ID    VFI     Access-Port Extended-Site
-    -------    -----    ---     ----------- -------------
1    101        25838    101     5           1
```

   For the mapped VLAN, the VNI and the Virtual Forwarding Instance (VFI) (the internal Layer 2 VPN identifier, which is the same as the VLAN identifier) are shown, along with the number of access (local) ports belonging to that VLAN and the number of sites to which this mapped VLAN is extended.

5. To display information on a particular VNI, enter the **show overlay-gateway** command followed by the overlay gateway name, the **vni** keyword, and the VNI number, as shown in the following example.

```
device# show overlay-gateway sanjose vni 25838
Overlay Gateway Name    : sanjose
#    VN-ID      VLAN-ID  VFI     Access-Port Extended-Site
-    -----      -------  ---     ----------- -------------
1    25838      101      101     5           1

device# show overlay-gateway sanjose vni 67924
Overlay Gateway Name    : sanjose
#    VN-ID      VLAN-ID  VFI     Access-Port Extended-Site
-    -----      -------  ---     ----------- -------------
1    67924      102      102     3           1
```

The two examples of command output display information for the two VNIs configured for the overlay gateway sanjose. For the VNI, the VLAN and Virtual Forwarding Instance (VFI) (the internal Layer 2 VPN identifier, which is the same as the VLAN identifier) are shown, along with the number of access (local) ports belonging to the VLAN and the number of sites to which the mapped VLAN is extended.

6. To display information on a remote site, enter the **show overlay-gateway** command followed by the overlay gateway name, the **site** keyword, and the name of the remote site, as shown in the following examples.

```
device# show overlay-gateway gt-1 site mm1-site
Overlay Gateway Site Name     : mm1-site
Active IP address             : 20.20.20.1
Status                        : Up
Monitoring                    : Disabled
Connection                    : N/A
Extended Vlans                :
   100

device# show overlay-gateway gt-1 site mm2-site
Overlay Gateway Site Name     : mm2-site
Active IP address             : 70.70.70.1
Status                        : Up
Monitoring                    : Disabled
Connection                    : N/A
Extended Vlans                :
   100
```

The previous examples show output for a system in which failure detection (Monitoring) is disabled.

```
device# show overlay-gateway sanjose site denver
Overlay Gateway Site Name     : denver
Active IP address             : 2.2.2.2
IP List                       : 3.3.3.1, 4.4.4.1, 5.5.5.1

Status                        : Up
Monitoring                    : ENABLED
Connection                    : Alive
Extended Vlans                :
   101, 102
Total 2 Extended Vlan
```

The previous example shows output for a system in which failure detection is enabled. The example displays the IP address for the remote site denver, the status of the VXLAN tunnel to that site, and the mapped VLANs that are extended to the site.

```
device# show overlay-gateway sanjose site denver
Overlay Gateway Site Name     : denver
IP address                    : 2.2.2.2
Status                        : Down(No Source Interface)
Monitoring                    : DISABLED
Connection                    : N/A
Extended Vlans                :
   101, 102
Total 2 Extended Vlan
```

The previous example shows the remote site denver is down because no source interface has been configured for the remote site.

```
device# show overlay-gateway sanjose site denver
Overlay Gateway Site Name     : denver
IP address                    : 2.2.2.2
Status                        : Down(No Route to Destination)
Monitoring                    : DISABLED
Connection                    : N/A
Extended Vlans                :
   101, 102
Total 2 Extended Vlan
```

The previous example shows the remote site denver is down because no route is available to the destination.

7. To display information on MAC addresses learned from a particular remote site, enter the **show mac-address vxlan gw** command followed by the overlay gateway name, the **site** keyword , and the remote site name, as shown in the following example.

```
device# show mac-address vxlan gw sanjose site denver
Total active entries from remote-site denver = 2
MAC-Address        VNI         Type         Port
000c.2900.0022     25838       Dynamic      VxL-2.2.2.2
000c.2900.0023     67924       Dynamic      VxL-2.2.2.2
```

The example shows the MAC addresses learned for the site denver on the VXLAN overlay gateway sanjose. VNI information is displayed for the MAC address instead of the VLAN. The port information is the IP address of the site denver with the prefix "VxL-".

NOTE
MAC addresses learned from the remote site are shown with the prefix "VxL-" followed by the IP address of the remote site.

NOTE
The IP address for the VXLAN tunnel (remote site) and its configured extended VLANs are also displayed in the output for the basic **show mac-address** command.

8. To display MAC address information for a particular VNI, enter the **show mac-address vxlan vni** command followed by the VNI number, as shown in the following example.

```
device# show mac-address vxlan vni 25838
Total active entries from VNI 25838 = 2
MAC-Address      VNI          Type          Port
000c.2900.0011   25838        Dynamic       1/1/48
000c.2900.0022   25838        Dynamic       VxL-2.2.2.2
```

The example displays two MAC addresses for VNI 25838, one associated with port 1/1/48, which is the local VXLAN access port, and the other associated with the remote site (VxL-2.2.2.2). Previous output from the **show overlay-gateway sanjose vni 25838** command (refer to Step 5) shows that the VNI is associated with local VLAN 101.

9. To display MAC address information for a particular VLAN, enter the **show mac-address vlan** command followed by the VLAN number.

```
device# show mac-address vlan 101
Total active entries from VLAN 101 = 2
MAC-Address     Port         Type          VLAN
000c.2900.0011  1/1/48       Dynamic       101
000c.2900.0022  VxL-2.2.2.2  Dynamic       101
```

The example shows two MAC addresses associated with VLAN 101, one for the local VXLAN access port 1/1/48, and the other for the VXLAN remote site with the destination IP address 2.2.2.2.

NOTE
The **clear mac-address** command can be used with all the parameters given in the **show mac-address** examples to clear old table entries. For example, the **clear mac-address vlan 101** command clears the MAC address table entries for the VLAN.

10. To display information for all remote sites, enter the **show overlay-gateway** command, specifying the site name, with the **site** and **all** keywords.

```
device(config-overlay-gw-gt-1)# show overlay-gateway gt-1 site all

Overlay Gateway Name    : gt-1
Type                    : layer2-extension
Source IP Interface     : loopback 1 (vrf: default-vrf, IP address: 10.10.10.1)
Total Mapped Vlans      : 1
Total Sites             : 2
#    SiteName                                    IP-Address       Status Ext-Vlans
-    --------                                    ----------       ------ ---------
1    mm1-site                                    20.20.20.1       Up     (100)
2    mm2-site                                    70.70.70.1       Up     (100)
```

The example shows that two remote sites are configured for the overlay gateway gt-1. The status of the VXLAN tunnel for each one is up. Both sites are mapped to the same VLAN.

11. To display network counters for a site, enter the **show overlay-gateway** command, specifying the site name, with the **network-counters** keyword.

```
device# show overlay-gateway 1 site mm2-site network-counters

------ The Network Counter Stats ------
     InUcastPkts: 15                         InUcastBytes: 2670
     InMcastPkts: 0                          InMcastBytes: 0
     InBcastPkts: 0                          InBcastBytes: 0

     OutKnownUcastPkts: 7                    OutKnownUcastBytes: 1246
     OutBumPkts       : 8                    OutBumBytes       : 1424

Bum Traffic consists Unknown Ucast, Mcast, Bcast Traffic
```

12. To display VLAN VNI mapping information for a given range of VLANs for an overlay gateway, enter the **show overlay-gateway** command along with the site name, using the **vlan** and **to** keywords to specify the range of VLANs.

```
device# show overlay-gateway v1 vlan 200 to 204

Overlay Gateway Name : v1
Type : layer2-extension
Source IP Interface : loopback 1 (vrf: default-vrf, IP address: 10.10.10.1)
Total Mapped Vlans : 5
Total Sites : 1
# VLAN-ID VN-ID VFI Access-Port Extended-Site Cross-Connect
- ------- ----- --- ----------- ------------- --------------
1 200     20200 200      1           1        DISABLED
2 201     20201 201      1           1        DISABLED
3 202     20202 202      1           1        DISABLED
4 203     20203 203      1           1        DISABLED
5 204     20204 204      1           1        DISABLED
```

13. To display the counters for a VNI associated with the VXLAN tunnel, enter the **show overlay-gateway** command with the site name, followed by the **vnid** keyword and value to specify a VNI and the **counters** keyword.

```
device# show overlay-gateway 1 vnid 30000 counters

------ The VNI Counter Stats ------
     InUcastPkts: 30                         InUcastBytes: 4590
     InMcastPkts: 0                          InMcastBytes: 0
     InBcastPkts: 0                          InBcastBytes: 0

     OutKnownUcastPkts: 38                   OutKnownUcastBytes: 4590
     OutBumPkts       : 0                    OutBumBytes       : 0

Bum Traffic consists Unknown Ucast, Mcast, Bcast Traffic
```

14. To display the access counters for an Ethernet port associated with the VLAN mapped in the VXLAN tunnel, enter the **show overlay-gateway** command with the site name, followed by the **vnid** keyword and value to specify a VNI, the port number in the form **ethernet** *unit/slot/port*, and the **access-counters** keyword.

```
device# show overlay-gateway 1 vnid 30000 ethernet 2/1/17 access-counters

------ The Access Counter Stats ------
     InUcastPkts: 15                          InUcastBytes: 1920
     InMcastPkts: 0                           InMcastBytes: 0
     InBcastPkts: 0                           InBcastBytes: 0

     OutKnownUcastPkts: 23                    OutKnownUcastBytes: 1920
     OutBumPkts       : 0                     OutBumBytes       : 0

Bum Traffic consists Unknown Ucast, Mcast, Bcast Traffic
```

15. To display the access counters for a LAG interface associated with the VLAN mapped in the VXLAN tunnel, enter the **show overlay-gateway** command with the site name, followed by the **vnid** keyword and value to specify a VNI, the **lag** keyword and ID, and the **access-counters** keyword.

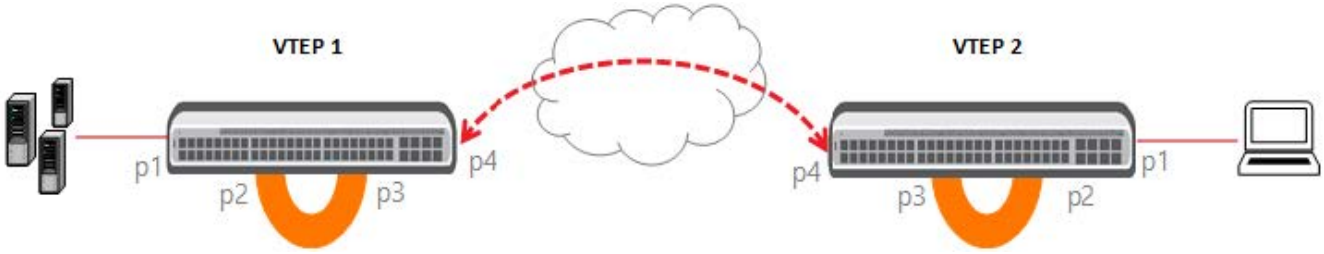
```
device# show overlay-gateway 1 vnid 30000 lag 2 access-counters

------ The Access Counter Stats ------
     InUcastPkts: 15                          InUcastBytes: 1920
     InMcastPkts: 0                           InMcastBytes: 0
     InBcastPkts: 0                           InBcastBytes: 0

     OutKnownUcastPkts: 30                    OutKnownUcastBytes: 3840
     OutBumPkts       : 0                     OutBumBytes       : 0

Bum Traffic consists Unknown Ucast, Mcast, Bcast Traffic
```

MACsec over VXLAN

MACsec over VXLAN is an end-to-end security protocol that provides a secured environment to protect Ethernet frames traveling over IP networks. Though Virtual Extensible Local Area Network (VXLAN) provides a Layer 2 extension across the LAN/WAN, a connection between sites is not secured. MACsec over VXLAN addresses this and achieves a secure Layer 2 extension across the LAN/WAN using cross-connect VXLAN Network Identifiers (VNIs).

In this solution, a Media Access Control security (MACsec) session must be established over a VXLAN tunnel using an external loopback connection. For each MACsec session, from one site to another site there should be a dedicated port for the loopback connection and a dedicated VLAN that is mapped to the VNI. The VNI that is mapped to the dedicated VLAN is the cross-connect VNI which provides point-to-point Layer 2 extensions across the LAN/WAN. The external loopback port connected to the MACsec port must be the only member of the dedicated VLAN. The VLAN that is used for the cross-connect VNI cannot be used as a regular user VLAN, and any other VLAN-level feature must not be enabled. The protocol packets coming to the port which belong to the cross-connect VNI are not punted to the CPU. MAC learning is disabled on the port too. Any packets routed to the MACsec port will be encrypted and tunneled to the remote Virtual Tunnel End Point (VTEP). On the receive path, packets will be decrypted and forwarded.

FIGURE 110 MACsec over VXLAN Packet Flow



In the example in Figure 110, port p1 is connected to the end user. Port p2 is the MACsec port, which has an external loopback connection to port p3. The traffic that enters the device on access port p1 is switched to MACsec port p2. At port p2, traffic is encrypted and sent out of port p2. Because there is an external loopback configuration, encrypted traffic is received on port p3, which is the only member of the dedicated VLAN. This VLAN is VLAN X. VLAN X is mapped to the VNI so that the encrypted traffic is VXLAN-tunneled to the other VTEP. When the traffic is received on the other VTEP, after VXLAN de-tunneling, the payload-encrypted Layer 2 frame is switched to port p3. Due to the external loopback, encrypted traffic is received back on port p2. At port p2, traffic is decrypted and switched to port p1.

Configuring MACsec over VXLAN

Complete the following steps to configure MACsec over VXLAN.

1. Configure MACsec for a loopback interface (p2 in the example, where ports p1 and p2 are part of the same VLAN).
   a) Enable MACsec on the device.

```
device(config)# dot1x-mka-enable
```

   b) Enable MACsec Key Agreement (MKA) to support the MACsec licensing functionality on a specific interface (loopback interface p2 in this case).

```
device(config-dot1x-mka)# enable-mka ethernet 1/3/1
device(config-dot1x-mka-1/3/1)# pre-shared-key abcdeabcdeabcdeabcde111111111111 key-name abc2
```

2. Configure the VXLAN tunnel with cross-connect VNI mapping (port p3 is an untagged member of the extended VLAN).
   a) Configure a VXLAN gateway record.

```
device# configure terminal
device(config)# overlay-gateway gate1
```

   b) Set the VXLAN gateway type as a Layer 2 extension.

```
device(config-overlay-gw-gate1)# type layer2-extension
```

   c) Specify the loopback interface with the IP address that will be used as the source IP address for VXLAN tunnels.

```
device(config-overlay-gw-gate1)# ip interface loopback 1
```

   You must use a loopback interface, rather than an explicit IPv4 address, as the source address.

   d) Map the VLAN with an untagged member as the loopback interface (p3) to the cross-connect VNI.

```
device(config-overlay-gw-gate1)# map vlan 2 to vni 3 cross-connect
```

   The default VLAN (typically, VLAN 1) cannot be mapped to a VNI or extended over a VXLAN segment.

   e) Create a remote site (VXLAN tunnel) and configure the IP address for the site.

```
device(config-overlay-gw-gate1)# site site1
device(config-overlay-gw-gate1-site1)# ip address 67.67.67.1
```

   f) Extend the mapped VLAN to the remote site.

```
device(config-overlay-gw-gate1-site1)# extend vlan add 2
```

   The following example configures overlay gateway gate1 and maps VLAN 2 to cross-connect VNI 3. A VXLAN tunnel is configured by creating a remote site (site1) and configuring its IP address (67.67.67.1) as the destination address. Finally, the VLAN (VLAN 2) is mapped over the overlay gateway.

```
device# configure terminal
device(config)# overlay-gateway gate1
device(config-overlay-gw-gate1)# type layer2-extension
device(config-overlay-gw-gate1)# ip interface loopback 1
device(config-overlay-gw-gate1)# map vlan 2 to vni 3 cross-connect
device(config-overlay-gw-gate1)# site site1
device(config-overlay-gw-gate1-site1)# ip address 67.67.67.1
device(config-overlay-gw-gate1-site1)# extend vlan add 2
device(config-overlay-gw-gate1-site1)# end
```

VXLAN RIOT

VXLAN routing is the process in which a Virtual Tunnel End Point (VTEP) receives a VXLAN packet, removes the VXLAN header, and then performs a Layer 3 route lookup on the inner decapsulated packet.

NOTE
VXLAN RIOT is supported for RUCKUS ICX 7550 and ICX 7850 devices only.

NOTE
Only unicast routing is supported with VXLAN RIOT.

Within the same administrative domain, hosts that are distributed across multiple VXLAN Network Identifiers (VNIs) and VXLAN segments, must be allowed to communicate with one another. This requirement drives the need for the ability to route traffic into and out of Layer 2 VXLAN tunnels, also known as Routing In and Out of Tunnels (RIOT).

VXLAN RIOT must be enabled to configure a VE interface over a VLAN mapped to a VNI. In a VXLAN RIOT configuration, ARP (broadcast and unicast) and OSPF (multicast) packets received over the tunnel must be processed and mapped to one of the overlay VE interfaces. The overlay VE interface must be configured as a part of a non-default user VRF for routing and forwarding. The protocol packets are copied to the CPU and processed according to the corresponding protocols. The data packets (both from the access port and the tunnel port) are L3 routed in the user VRF assigned to the overlay VE interface. However, uRPF will not be supported on these overlay VE interfaces for which VLAN is mapped to a VxLAN VNI.

VXLAN RIOT requires two lookups (overlay and underlay) sequentially for the same packet, which requires a partition of the egress Layer 3 interface table and egress Layer 3 next-hop table (EGR_L3_INTF and EGR_L3_NEXT_HOP) between the underlay and overlay. Therefore, the hardware tables must be partitioned into two parts. By default, VXLAN RIOT is disabled, so nothing is reserved for the overlay. The full next-hop and interface table will be completely reserved for the underlay. VXLAN RIOT must be enabled to configure the overlay partition size for the next-hop table. The partition size for the overlay next-hop table is configurable in increments of 4000 entries (for example, 4000, 8000, 12000, and 16000). The remaining space is reserved for the underlay.

By default, the partition size for the overlay egress interface table is set to 2000 entries when VXLAN RIOT is enabled. The remaining space is reserved for the underlay. The partition size for the overlay egress interface table is not configurable.

Because the tables are divided between overlay and underlay, Layer 3 scale numbers are impacted.

FIGURE 111 VXLAN RIOT Underlay Transport Network Example

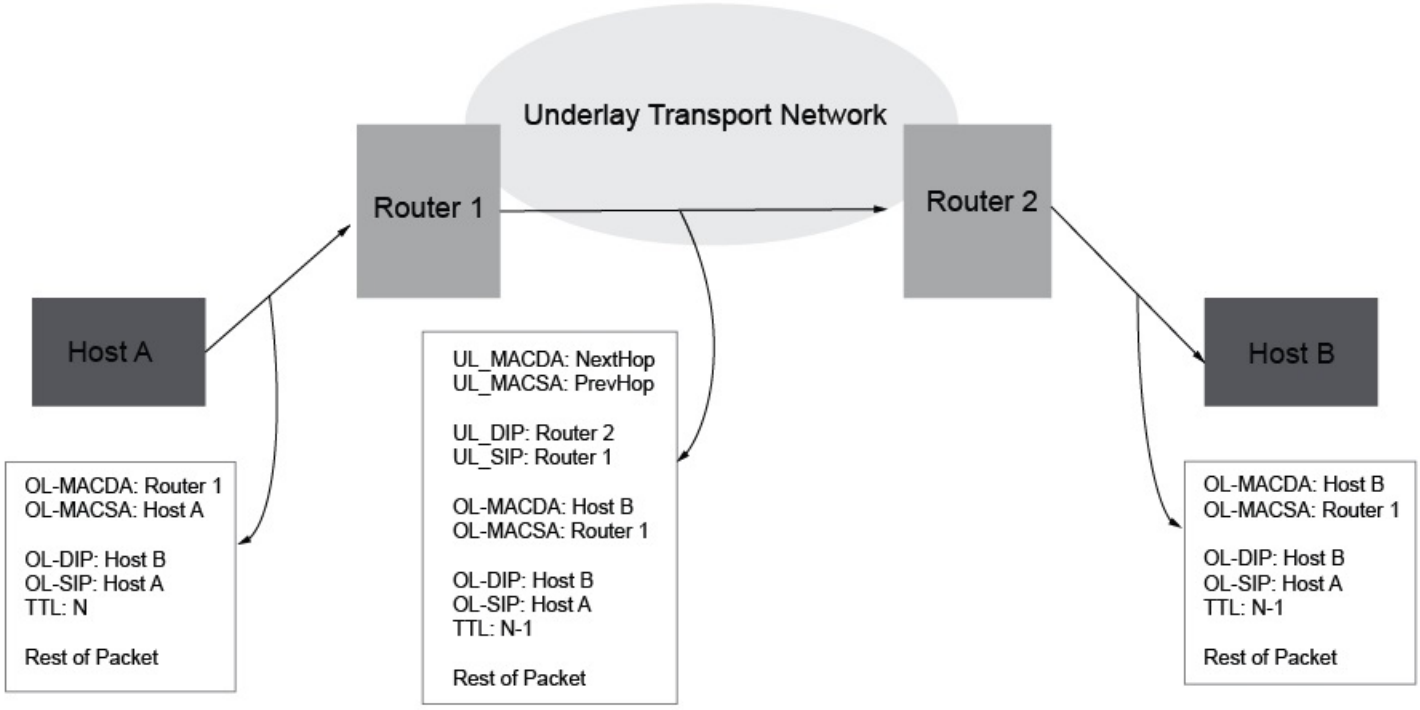
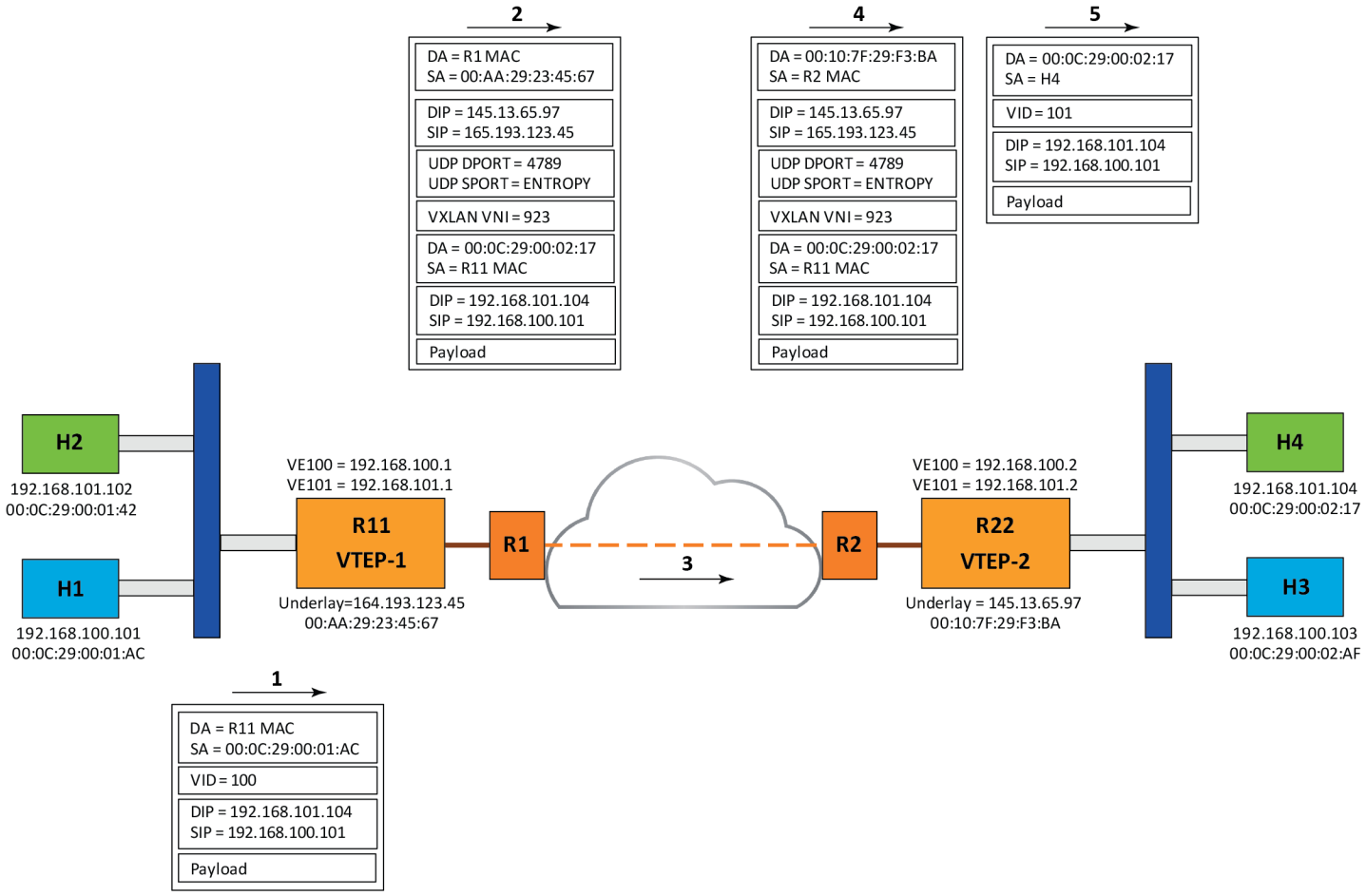


FIGURE 112 VXLAN Routing In and Out of Tunnels Example



If a VXLAN device supports RIOT, VXLAN RIOT must be enabled before the switch allows for associating a VE interface with a VLAN associated with a VNI.

Packets received on the VXLAN tunnel with that VNI with the gateway router's MAC address as the DMAC address will be decapsulated and processed by the routing engine.

Packets routed to the VE interface or VLAN with a MAC address (known or unknown) destined for the VXLAN tunnel will be encapsulated and forwarded using the SMAC address of the switch associated with that VE interface, and the DMAC address of the destination device.

Enabling VXLAN RIOT

Complete the following steps to enable VXLAN RIOT.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enable VXLAN RIOT.

```
device(config)# vxlan-riot enable
```

3. Write the changes to memory.

```
device(config)# write memory
```

4. Exit global configuration mode.

```
device(config)# exit
```

5. Reload the software changes to the hardware.

```
device# reload
```

The following example enables VXLAN RIOT.

```
device# configure terminal
device(config)# vxlan-riot enable
device(config)# write memory
device(config)# exit
device# reload
```

Configuring the Overlay Next-Hop Partition Size

You must enable VXLAN RIOT before specifying the overlay next-hop hardware partition size.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Specify the size of the overlay next-hop hardware partition.
   - 4096 (4000 entries)
   - 8192 (8000 entries)
   - 12288 (12000 entries)
   - 16384 (16000 entries)

```
device(config)# vxlan-riot overlay-nhop-partition 16384
```

3. Write the changes to memory.

```
device(config)# write memory
```

4. Exit global configuration mode.

```
device(config)# exit
```

5. Reload the software changes to the hardware.

```
device# reload
```

The following example specifies a partition size of 16384 entries.

```
device# configure terminal
device(config)# vxlan-riot overlay-nhop-partition 16384
device(config)# write memory
device(config)# exit
device# reload
```

Anycast Gateway with VXLAN

Overview

Anycast gateway with Virtual Extensible Local Area Network (VXLAN) allows the same gateway IP addresses to be used across all Virtual Tunnel End Points (VTEPs) in a VXLAN network. This ensures that every VTEP can act as the default gateway for connected devices. Anycast gateway enables seamless host mobility, flexible workload placement, and optimal traffic forwarding throughout the VXLAN fabric.

Important Terminology for Anycast Gateway with VXLAN

The below are common terms used in reference to anycast gateway with VXLAN:

- **ARP**: Address Resolution Protocol
- **EVPN**: Ethernet Virtual Private Network.
- **MAC**: Media Access Control
- **VE**: Virtual Ethernet
- **VLAN**: Virtual Local Area Network
- **VNI**: Virtual Network Identifier
- **VRF**: Virtual Routing and Forwarding
- **VTEP**: Virtual Tunnel End Point. In BGP EVPN terms, a VTEP can also be referred to as a leaf. Refer to the BGP EVPN on page 363 and BGP EVPN Multihoming on page 389 sections for more information.
- **VXLAN**: Virtual Extensible Local Area Network

Requirements

- Anycast gateway with VXLAN is not supported for RUCKUS ICX 8200 devices.

Considerations

- Anycast gateway with VXLAN is disabled by default.
- The current anycast gateway with VXLAN method supported is manual MAC configuration. This involves manually configuring the same MAC address, on the virtual Ethernet (VE) interfaces for which overlay virtual routing and forwarding (VRF) is configured, for all VTEPs within the VXLAN network.
- To enable anycast gateway with VXLAN, you must configure an anycast gateway MAC address on an IP interface that supports VRF. Anycast gateway with VXLAN becomes active on the IP interface once an anycast MAC address is configured.

Prerequisites

- Devices must support VXLAN protocols. For more details on VXLAN device support for RUCKUS ICX devices, refer to the *RUCKUS FastIron Features and Standards Support Matrix*.
- VLANs must be configured in advance.
- VE interfaces must be configured with overlay VRF.

Limitations

- IPv6 is not supported.
- Only VE interfaces are supported.
- The maximum number of anycast gateway IP interfaces that can be configured is the same as the maximum number of VE interfaces supported by the platform. This number may vary if the Virtual Router Redundancy Protocol (VRRP) or IP MAC address is enabled for other VE interfaces.

Configuring Anycast Gateway in a VXLAN Network

You can configure anycast gateway with Virtual Extensible Local Area Network (VXLAN) to ensure that all Virtual Tunnel End Points (VTEP) in the VXLAN network can act as the default gateway for the devices connected to it. The following task demonstrates how to configure an anycast gateway for one Virtual Ethernet (VE) interface.

NOTE
To ensure that the MAC address in the Address Resolution Protocol (ARP) entry of the host device matches the MAC address across all VXLAN Tunnel Endpoints (VTEPs), you can configure anycast gateway using manual MAC configuration via the FastIron CLI. This involves manually configuring the same MAC address on the sameVE interfaces on all VTEPs within the VXLAN network.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Configure a VE interface.

   ```
   device(config)# interface ve 5
   ```

3. Assign the VE interface to a non-default overlay Virtual Routing and Forwarding (VRF) instance.

   ```
   device(config-vif-5)# vrf forwarding overlay-vrf
   ```

   VE 5 is assigned to a non-default VRF called "overlay-vrf " ensuring that all traffic on this interface is routed according to the routing table and policies defined for the overlay-vrf VRF instance. This isolates the interface from other VRF instances and ensures that it follows the specific routing rules and paths set for overlay-vrf.

4. Set the IP address and subnet mask for the VE interface.

   ```
   device(config-vif-5)# ip address 100.1.1.1/24
   ```

5. Configure the anycast gateway MAC address for the VE interface.

   ```
   device(config-vif-5)# anycast-gw-mac aabb.ccdd.eeff
   ```

The following example demonstrates how to configure VE interfaces and map them to VLANs in a VXLAN environment. VE interfaces 5 and 6 are assigned IP addresses and assigned to the “overlay-vrf” VRF instance. Additionally, the anycast gateway MAC address is configured on these VE interfaces. These VE interfaces correspond to VLANs 5 and 6, which are then mapped to VNIs 1005 and 1006, respectively, enabling VXLAN functionality and extending these VLANs to a remote site labeled R2.

VE 5:

```
device# configure terminal
device(config)# interface ve 5
device(config-vif-5)# vrf forwarding overlay-vrf
device(config-vif-5)# ip address 100.1.1.1/24
device(config-vif-5)# anycast-gw-mac aabb.ccdd.eeff
```

VE 6 :

```
device# configure terminal
device(config)# interface ve 6
device(config-vif-6)# vrf forwarding overlay-vrf
device(config-vif-6)# ip address 101.1.1.1/24
device(config-vif-6)# anycast-gw-mac aabb.ccdd.eeff
```

VXLAN :

```
device# configure terminal
device(config)# overlay-gateway R1
device(config-overlay-gw-R1)# map vlan 5 to vni 1005
device(config-overlay-gw-R1)# map vlan 6 to vni 1006
device(config-overlay-gw-R1)# remote-site R2
device(config-overlay-gw-R1)# extend vlan add 5
device(config-overlay-gw-R1)# extend vlan add 6
```

Displaying Anycast Gateway VXLAN Information

You can use a **show** command to view information about anycast gateway with VXLAN configurations.

Use the following command to view anycast gateway with VXLAN information. For more detailed information on this command, refer to the *RUCKUS FastIron Command Reference Guide*.

Enter the **show ip interface ve** command to display information about IP configurations for a specified VE interface, including anycast gateway configurations.

```
device> show ip interface ve 100

Interface Ve 100
  members: ethe 1/1/19 ethe 1/1/31 ethe 1/1/35 ethe 1/1/48 ethe 2/1/14 ethe 2/1/23 lag lg1 lag lg10
  active: ethe 1/1/35 ethe 1/1/48
  port enabled
  port state: UP
  ip address: 100.1.1.1         subnet mask: 255.255.255.0
  Port belongs to VRF: overlay-vrf
  encapsulation: ETHERNET, mtu: 9216, metric: 1
  directed-broadcast-forwarding: disabled
  ICMP redirect: enabled
  proxy-arp: disabled
  ip arp-age:  10 minutes
  no delay in notification
  No Helper Addresses are configured.
  No inbound ip access-list is set
  No outgoing ip access-list is set
  Anycast gateway: enabled
  Anycast gateway mac : 0002.1111.1110
```

In this example, the anycast gateway feature is enabled for the VE interface. A MAC address of 0002.1111.1110 is assigned to the anycast gateway.

BGP EVPN

BGP EVPN Overview

Border Gateway Protocol (BGP) Ethernet Virtual Private Network (EVPN) provides an efficient control plane protocol for Virtual Extensible LAN (VXLAN). When BGP EVPN is configured for a VXLAN, remote Virtual Tunnel End Points (VTEPs) and Virtual Network Identifiers (VNIs) do not need to be manually configured, and remote MAC addresses can be learned via the control plane.

VTEPs in a VxLAN network can use BGP EVPN to exchange locally learned Layer 2 and Layer 3 destinations with remote VTEPs.

The following benefits apply to the BGP EVPN control plane for VXLAN:

- Auto-discovery of remote VTEPs and their Layer 2 VPN membership information. Manual configuration of remote VTEP information is not required.
- Distribution of locally learned MAC addresses to remote VTEPs so that the flooding of unknown unicast traffic does not occur.
- Distribution of locally learned MAC-IP bindings from the ARP table to remote VTEPs.
- Assistance in implementing various advanced features, such as multi-homing, traffic load balancing, and ARP suppression.

Important Terminology for BGP EVPN

The below are common terms used in reference to BGP EVPN:

- **CE**: Customer Edge.
- **EVPN**: Ethernet Virtual Private Network. A common implementation is Ethernet over VXLAN. This is the implementation outlined in this chapter.
- **MP-BGP**: Multi-Protocol Border Gateway Protocol. BGP capability to carry routes for multiple, different address families.
- **PE**: Provider Edge.
- **RR**: Route Reflector for internal BGP peers.
- **RT**: Route Target.
- **RD**: Route Distinguisher.
- **VNI**: Virtual Network Identifier.
- **VTEP**: Virtual Tunnel End Point. In BGP EVPN terms, a VTEP can also be referred to as the leaf.
- **VXLAN**: Virtual Extensible Local Area Network.

NOTE
For more information on VXLAN-related terminologiy, refer to the VXLAN on page 331 chapter.

Multiple modules in the system are involved in the implementation of BGP EVPN. These are described in subsequent sections.

EVPN Support in BGP

A new Address Family Identifier (AFI) and Subsequent Address Family Identifier (SAFI) combination of L2VPN EVPN is introduced in MP-BGP to carry EVPN routing protocol information between peers. Two BGP speakers, capable of both advertising and processing EVPN Network Layer Reachability Information (NLRI), are employed.

A VTEP advertises the IP address of a local tunnel endpoint and locally configured L2-VNIs to other VTEPs. Deletion of L2-VNI memberships is also advertised to other VTEPs. Consequently, receiving VTEPs gain knowledge about remote VTEPs and their L2 VPN memberships. Receiving VTEPs can then establish or terminate a VXLAN tunnel for the advertising VTEP, and update the ingress replication list for each L2-VNI common in both the local and remote member list.

VTEPs can learn MAC addresses of locally attached tenant systems from traffic sources such as data traffic, IEEE 802.1x, Link Layer Discovery Protocol (LLDP), or Address Resolution Protocol (ARP) generated by those tenant systems. Additionally, VTEPs learn remote MAC addresses by advertising locally learned MAC addresses to all other VTEPs in that EVPN instance.

EVPN Support in FDB

In BGP EVPN, MAC addresses are acquired from the control plane through BGP modules, deviating from the conventional data-plane MAC learning approach. MACs learned from the local access port within the BGP EVPN domain are exported to the BGP module from the Forwarding Database (FDB), enabling BGP to advertise these MACs.

These MAC addresses learned from the control plane are termed autonomous system (AS) remote-MACs and are exempted from the MAC aging process. Only the BGP control plane can add or remove remote MACs. This approach supports MAC mobility.

EVPN Manager

The BGP EVPN management system is designed to ensure the validation and processing of various message types received from peers, directing them to internal modules for local consumption.

Messages generated by internal modules undergo validation before being forwarded to BGP for peer advertisement. Configuration parameters, such as the mapping of Ethernet Tag-Id and VNI information, are received from the VXLAN manager. Tunnel IP information is also acquired and subsequently sent to BGP, which advertises it as an Integrated Routing and Bridging (IRB) Multicast Ethernet Tag (IMET) route to peers.

Additionally, the BGP EVPN management system receives MAC information from the FDB. This data is validated against the configurational database and then forwarded to BGP for peer advertisement. The management system also registers with BGP to obtain remote MAC IP information, which is validated and sent to the FDB and ARP modules. MAC IP information from the ARP module is validated and subsequently sent to BGP for peer advertisement.

BGP EVPN Considerations and Limitations

Consider the following when configuring BGP EVPN.

BGP EVPN Considerations

- When BGP EVPN is configured, remote MAC addresses are learned via the control plane only.

- VXLAN remote site configuration is not required when BGP EVPN is configured.
- BGP EVPN extensively uses standard and extended BGP path attributes. The **send-community** option must be configured for each BGP EVPN peer. Refer to BGP L2VPN EVPN Address Family on page 373 or to the **neighbor send-community** command in the *RUCKUS FastIron Command Reference* for more information on this command.
- Singlehoming and multihoming configurations are supported.

BGP EVPN Limitations

- BGP EVPN is supported for RUCKUS ICX 7850 and ICX 7550 devices only.

BGP EVPN Configuration

A VXLAN network can be divided into two main parts, consisting of the underlay and overlay. At the edges of the underlay network, Virtual Tunnel Endpoints (VTEPs) or Network Virtualization Edge Devices (NVEs) facilitate Layer 2 and Layer 3 connectivity among customer edge devices, or tenant systems, that belong to the overlay network. The underlay network functions as a logical switching fabric for the tenant systems within the overlay network.

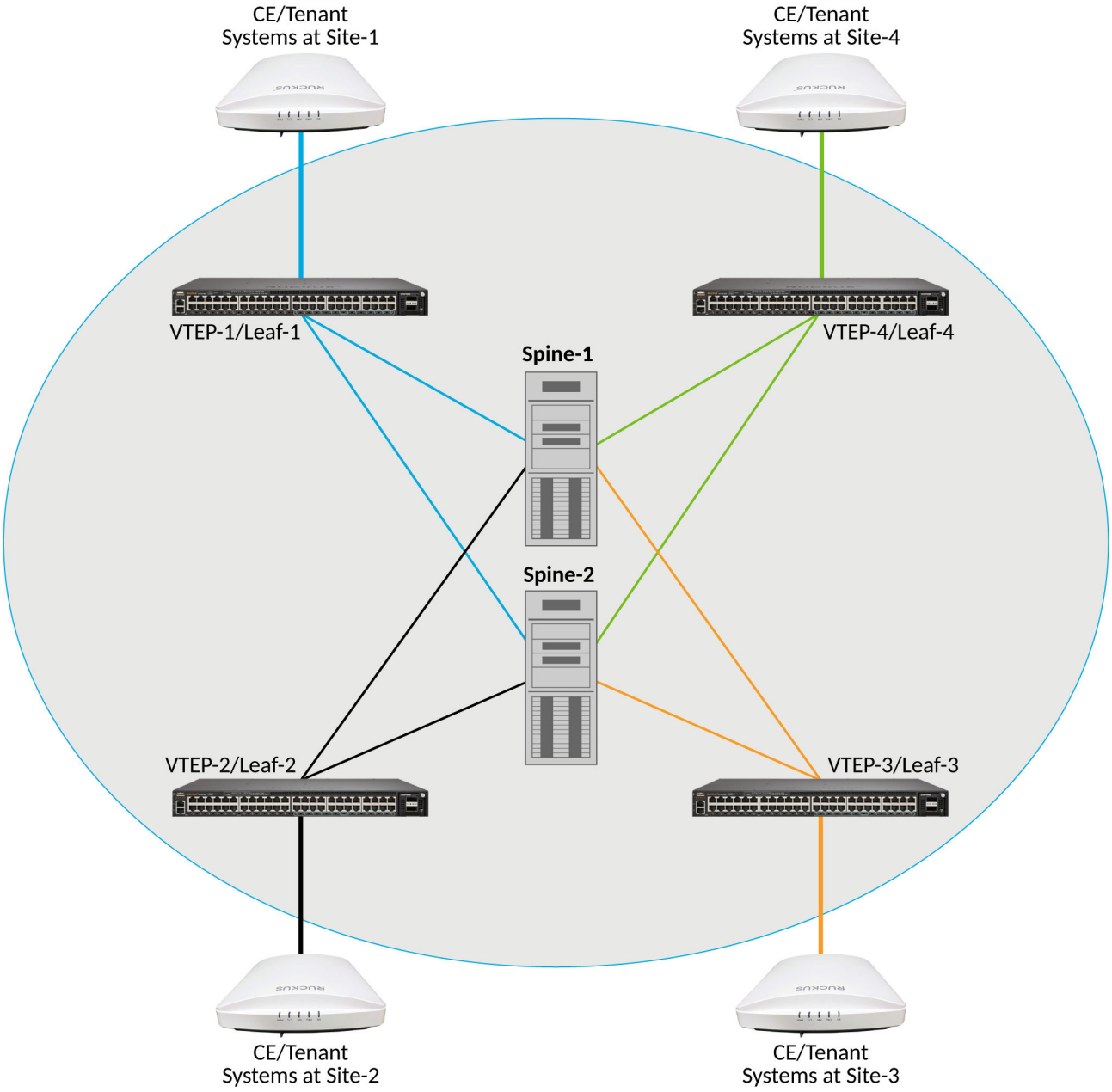
There are four steps involved for configuring BGP EVPN. Refer to these sections for more information:

- Configuring the Underlay Network on page 366
- Configuring iBGP EVPN Peering Between VTEPs on page 368
- Configuring EVPN Instances for the Overlay Network on page 369 (this step is not required for Route Reflectors (RR).
- Configuring the Overlay for VXLAN Routing on page 371

Typical BGP EVPN Single-homed Network Topology

The illustration below shows a BGP EVPN Single-homed typical network topology. Here, a leaf-spine topology comprises VTEPs (also known as "leaf" devices) and "spine" devices. The spine devices connect all the leaf devices within the underlay network, thereby supporting BGP peering. This image shows a single-homed BGP EVPN topology that consists of four customer sites. There is one VTEP at each site. Each VTEP is connected to the spine devices. VTEP to VTEP communication is through the spine devices. The underlay network can run OSPF or RIP between the VTEPs and the spine to build Layer 3 connectivity. iBGP EVPN peers are also part of the underlay network. Internal BGP (iBGP) is used to propagate routes within a single autonomous system (AS), while BGP (or, external BGP) is used to propagate routes between different ASs. iBGP EVPN peering and the VXLAN tunnel endpoints are the loopback addresses configured on the VTEPs.

FIGURE 113 BGP EVPN Typical Network Typology



Configuring the Underlay Network

The BGP EVPN underlay network must be configured to build Layer 3 connectivity among VTEPs and intermediate routers. Loopback interface addresses, used as VTEP and BGP peer endpoints, must be routable in the underlay. While an Interior Gateway Protocol (IGP) such as OSPF or RIP can be enabled for underlay routing, it is recommended to employ OSPF for optimal underlay routing.

The following task shows how to configure EVPN for one VTEP (VTEP 1.1.1.1).

NOTE
Enabling OSPF for the underlay is recommended.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Configure a Virtual Ethernet interface.

```
device(config)# interface ve 2
```

3. Assign IP addresses to the VE interface.

```
device(config-vif-2)# ip address 2.0.0.1 255.255.255.0
```

4. Assign the VE interface to an OSPF area.

```
device(config-vif-2)# ip ospf area 0
```

5. Configure a loopback interface on the VE interface.

```
device(config-vif-2)# interface loopback 1
```

6. Assign IP addresses to the loopback interface.

```
device(config-lbif-1)# ip address 1.1.1.1 255.255.255.255
```

7. Assign the loopback interface to an OSPF area.

```
device(config-lbif-1)# ip ospf area 0
```

8. Set the loopback interface as passive.

```
device(config-lbif-1)# ip ospf passive
```

9. Return to global configuration mode.

```
device(config-lbif-1)# exit
device(config)#
```

10. Enable OSPF, and enter OSPF VRF router configuration mode

```
device(config)# router ospf
```

11. Configure an OSPF area.

```
device(config-ospf-router)# area 0
```

The following is an example showing EVPN configuration for two VTEPs and an RR on a spine for a BGP EVPN underlay network.

VTEP-1(1.1.1.1) — ve2 — RR :

```
device(config)# interface ve 2
device(config-vif-2)# ip address 2.0.0.1 255.255.255.0
device(config-vif-2)# ip ospf area 0
device(config-vif-2)# interface loopback 1
device(config-lbif-1) ip address 1.1.1.1 255.255.255.255
device(config-lbif-1) ip ospf area 0
device(config-lbif-1) ip ospf passive
device(config-lbif-1)# exit
device(config)# router ospf
device(config-ospf-router)# area 0
```

VTEP-2(2.2.2.2)--- ve3 — RR :

```
device(config)# interface ve 3
device(config-vif-3)# ip address 3.0.0.1 255.255.255.0
device(config-vif-3)# ip ospf area 0
device(config-vif-3)# interface loopback 1
device(config-lbif-1) ip address 2.2.2.2 255.255.255.255
device(config-lbif-1) ip ospf area 0
device(config-lbif-1) ip ospf passive
device(config-lbif-1)# exit
device(config)# router ospf
device(config-ospf-router)# area 0
```

VTEP-1 <----- ve2----> RR(3.3.3.3) <--- ve3---> VTEP-2:

```
device(config)# interface ve 2
device(config-vif-2)# ip address 2.0.0.100 255.255.255.0
device(config-vif-2)# ip ospf area 0
device(config-vif-2)# interface ve 3
device(config-vif-3)# ip address 3.0.0.100 255.255.255.0
device(config-vif-3)# ip ospf area 0
device(config-vif-3)# interface loopback 1
device(config-lbif-1) ip address 3.3.3.3 255.255.255.255
device(config-lbif-1) ip ospf area 0
device(config-lbif-1) ip ospf passive
device(config-lbif-1)# exit
device(config)# router ospf
device(config-ospf-router)# area 0
```

Configuring iBGP EVPN Peering Between VTEPs

iBGP EVPN peers are part of the underlay in a single-AS BGP EVPN VXLAN. iBGP EVPN peering can be configured in two ways:

- End-to-end EVPN peering among VTEPs, running on top of underlying hop-by-hop routing. This option can be used with a small number of VTEPs.
- EVPN peering between each VTEP and a Route Reflector (RR). The RR can run as part of the spine routers or as a dedicated node.

The BGP EVPN underlay network must be configured to build Layer 3 connectivity among VTEPs and intermediate routers. Loopback interface addresses, used as VTEP and BGP peer endpoints, need to be routable in the underlay. It is recommended to run an IGP such as OSPF in the underlay. The following task configures iBGP EVPN peering for a VTEP.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enable BGP routing.

   ```
   device(config)# router bgp
   ```

3. Configure the Autonomous System Number (ASN).

   ```
   device(config-bgp-router)# local-as 656
   ```

4. Specify the ASN for the remote neighbor.

   ```
   device(config-bgp-router)# neighbor 3.3.3.3 remote-as 656
   ```

5. Specify the neighbor interface over which the neighbor and local device will exchange prefixes.

   ```
   device(config-bgp-router)# neighbor 3.3.3.3 update-source loopback 1
   ```

6. Enable the BGP L2VPN EVPN address family.

```
device(config-bgp-router)# address-family l2vpn evpn
```

7. Enable the exchange of information with the neighbor.

```
device(config-bgp-l2vpn-evpn)# neighbor 3.3.3.3 activate
```

8. Send standard and extended community attributes to the neighbor.

```
device(config-bgp-l2vpn-evpn)# neighbor 3.3.3.3 send-community both
```

The following is an example showing an iBGP-EVPN peering configuration for two VTEPs and a route reflector (RR) on a spine for a BGP EVPN underlay network. For the RR, client-to-client reflection is enabled by default.

VTEP-1(1.1.1.1) — ve2 — RR :

```
device(config)# router bgp
device(config-bgp-router)# local-as 656
device(config-bgp-router)# neighbor 3.3.3.3 remote-as 656
device(config-bgp-router)# neighbor 3.3.3.3 update-source loopback 1
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-l2vpn-evpn)# neighbor 3.3.3.3 activate
device(config-bgp-l2vpn-evpn)# neighbor 3.3.3.3 send-community both
```

VTEP-2(2.2.2.2)--- ve3 — RR :

```
device(config)# router bgp
device(config-bgp-router)# local-as 656
device(config-bgp-router)# neighbor 3.3.3.3 remote-as 656
device(config-bgp-router)# neighbor 3.3.3.3 update-source loopback 1
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-l2vpn-evpn)# neighbor 3.3.3.3 activate
device(config-bgp-l2vpn-evpn)# neighbor 3.3.3.3 send-community both
```

.

VTEP-1 <----- ve2----> RR(3.3.3.3) <--- ve3---> VTEP-2:

```
device(config)# router bgp
device(config-bgp-router)# local-as 656
device(config-bgp-router)# neighbor 1.1.1.1 remote-as 656
device(config-bgp-router)# neighbor 1.1.1.1 update-source loopback 1
device(config-bgp-router)# neighbor 2.2.2.2 remote-as 656
device(config-bgp-router)# neighbor 2.2.2.2 update-source loopback 1
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-l2vpn-evpn)# neighbor 1.1.1.1 activate
device(config-bgp-l2vpn-evpn)# neighbor 1.1.1.1 send-community both
device(config-bgp-l2vpn-evpn)# neighbor 1.1.1.1 route-reflector-client
device(config-bgp-l2vpn-evpn)# neighbor 2.2.2.2 activate
device(config-bgp-l2vpn-evpn)# neighbor 2.2.2.2 send-community both
device(config-bgp-l2vpn-evpn)# neighbor 2.2.2.2 route-reflector-client
```

Configuring EVPN Instances for the Overlay Network

In order to configure BGP EVPN as a Layer 2 extension protocol, certain configurations are required, including enabling BGP EVPN as the layer 2 extension protocol. The following task creates three EVPN instances for a VTEP and sets the configurations for the EVPN instances. EVPN instances corresponding to respective VNIs are also configured. This task sets the configurations for one VTEP (VTEP 1.1.1.1).

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enable EVPN configuration mode.

```
device(config)# l2vpn evpn
```

3. Create an EVPN instance and specify that a VLAN-based VXLAN service model is used.

```
device(config-evpn)# evpn-instance 10035 vlan-based
```

4. Configure a Layer 2 (L2) VNI.

```
device(config-evpn-evi)# vni 10035 l2
```

5. Specify that route distinguishers (RDs) are configured automatically for the VNI.

```
device(config-evpn-evi)# rd auto
```

6. Specify that export routes are configured automatically for the VNI.

```
device(config-evpn-evi)# route-target export auto
```

7. Specify that import routes for the VNI are configured automatically.

```
device(config-evpn-evi)# route-target import auto
```

8. Return to EVPN configuration mode.

```
device(config-evpn-evi)# exit
```

9. Create an EVPN instance and specify that a VLAN-based VXLAN service model is used.

```
device(config-evpn)# evpn-instance 10036 vlan-based
```

10. Configure a VNI.

```
device(config-evpn-evi)# vni 10036 l2
```

11. Specify that RDs, export routes and import routes are configured automatically for the VNI.

```
device(config-evpn-evi)# rd auto
device(config-evpn-evi)# route-target export auto
device(config-evpn-evi)# route-target import auto
```

12. Return to EVPN configuration mode.

```
device(config-evpn-evi)# exit
```

13. Create an EVPN instance and specify that a VLAN-based VXLAN service model is used.

```
device(config-evpn)# evpn-instance 10037 vlan-based
```

14. Configure a VNI.

```
device(config-evpn-evi)# vni 10037 l2
```

15. Specify that RDs, export routes and import routes are configured automatically for the VNI.

```
device(config-evpn-evi)# rd auto
device(config-evpn-evi)# route-target export auto
device(config-evpn-evi)# route-target import auto
```

The following example configures three EVPN instances each for two VTEPs (VTEP 1.1.1.1 and VTEP 2.2.2.2) in an overlay network, and sets the configurations for the EVPN instances.

VTEP-1(1.1.1.1) — ve2 — RR :

```
device(config)# l2vpn evpn
device(config-evpn)# evpn-instance 10035 vlan-based
device(config-evpn-evi)# vni 10035 l2
device(config-evpn-evi)# rd auto
device(config-evpn-evi)# route-target export auto
device(config-evpn-evi)# route-target import auto
device(config-evpn-evi)# exit
device(config-evpn)# evpn-instance 10036 vlan-based
device(config-evpn-evi)# vni 10036 l2
device(config-evpn-evi)# rd auto
device(config-evpn-evi)# route-target export auto
device(config-evpn-evi)# route-target import auto
device(config-evpn-evi)# exit
device(config-evpn)# evpn-instance 10037 vlan-based
device(config-evpn-evi)# vni 10037 l2
device(config-evpn-evi)# rd auto
device(config-evpn-evi)# route-target export auto
device(config-evpn-evi)# route-target import auto
```

VTEP-2(2.2.2.2)--- ve3 — RR :

```
device(config)# l2vpn evpn
device(config-evpn)# evpn-instance 10035 vlan-based
device(config-evpn-evi)# vni 10035 l2
device(config-evpn-evi)# rd auto
device(config-evpn-evi)# route-target export auto
device(config-evpn-evi)# route-target import auto
device(config-evpn-evi)# exit
device(config-evpn)# evpn-instance 10036 vlan-based
device(config-evpn-evi)# vni 10036 l2
device(config-evpn-evi)# rd auto
device(config-evpn-evi)# route-target export auto
device(config-evpn-evi)# route-target import auto
device(config-evpn-evi)# exit
device(config-evpn)# evpn-instance 10037 vlan-based
device(config-evpn-evi)# vni 10037 l2
device(config-evpn-evi)# rd auto
device(config-evpn-evi)# route-target export auto
device(config-evpn-evi)# route-target import auto
```

Configuring the Overlay for VXLAN Routing

BGP EVPN can be configured as a Layer 2 extension protocol within the overlay of a VXLAN network.

To set up BGP EVPN as a Layer 2 extension protocol, specific configurations are required, including the setup of local VXLAN site information and the mapping of VLANs to VNIs. The following task sets the VXLAN overlay gateway type as Layer2 extension EVPN. It configures the IP interface for the overlay gateway, and maps a range of VLANs to be extended over the overlay gateway to a VXLAN Network Identifier (VNI). This task sets the configurations for one VTEP (VTEP 1.1.1.1).

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Configure an overlay gateway and enter gateway configuration mode.

   ```
   device(config)# overlay-gateway site-local
   ```

3. Set the VXLAN gateway type as Layer2 extension EVPN.

```
device(config-overlay-gw-site-local)# type layer2-extension evpn
```

4. Configure the IP interface for the overlay gateway

```
device(config-overlay-gw-site-local)# ip interface loopback 1
```

5. Map a range of VLANs (VLAN 30 through VLAN 40) to a VNI for the VXLAN overlay gateway.

```
device(config-overlay-gw-site-local)# map vlan-range 30 to 40 vni-start 10030
```

6. Map a VLAN to a VNI for the VXLAN overlay gateway.

```
device(config-overlay-gw-site-local)# map vlan 41 to vni 10041
```

The following example sets the VXLAN overlay gateway type as Layer2 extension EVPN. It configures the IP interface for the overlay gateway, and maps a range of VLANs to be extended over the overlay gateway to a VNI. The example sets the configurations for two VTEPs (VTEP 1.1.1.1 and VTEP 2.2.2.2).

VTEP-1(1.1.1.1) — ve2 — RR :

```
device(config)# overlay-gateway site-local
device(config-overlay-gw-site-local)# type layer2-extension evpn
device(config-overlay-gw-site-local)# ip interface loopback 1
device(config-overlay-gw-site-local)# map vlan-range 30 to 40 vni-start 10030
device(config-overlay-gw-site-local)# map vlan 41 to vni 10041
```

VTEP-2(2.2.2.2)--- ve3 — RR :

```
device(config)# overlay-gateway site-local
device(config-overlay-gw-site-local)# type layer2-extension evpn
device(config-overlay-gw-site-local)# ip interface loopback 1
device(config-overlay-gw-site-local)# map vlan-range 30 to 40 vni-start 10030
device(config-overlay-gw-site-local)# map vlan 41 to vni 10041
```

BGP Commands Supported for BGP EVPN

The following BGP commands, when configured in global configuration mode are also applicable to the BGP L2VPN EVPN address family:

- **capability**
- **client-to-client-reflection**
- **dampening**
- **default-local-preference**
- **default-metric**
- **graceful-restart**
- **local-as**
- **neighbor as4**
- **multipath**
- **neighbor advertisement-interval**
- **neighbor ao**
- **neighbor local-as**
- **neighbor peer-group**

- **neighbor shutdown**
- **neighbor soft-reconfiguration**
- **neighbor update-source**

The following BGP commands are configurable in BGP L2VPN EVPN address family configuration mode:

- **address-family**
- **client-to-client-reflection**
- **graceful-restart**
- **neighbor activate**
- **neighbor advertisement-interval**
- **neighbor ao**
- **neighbor description**
- **neighbor local-as**
- **neighbor maxas-limit**
- **neighbor password**
- **neighbor peer-group**
- **neighbor remote-as**
- **neighbor route-map**
- **neighbor route-reflector-client**
- **neighbor send-community**
- **neighbor shutdown**
- **neighbor soft-reconfiguration**
- **neighbors timers**
- **neighbor update-source**
- **neighbor weight**

NOTE
For more information on these commands, refer to the *RUCKUS FastIron Command Reference* Guide.

BGP L2VPN EVPN Address Family

A variety of BGP EVPN options can be configured in the BGP L2VPN EVPN address family configuration mode. The commands that you enter at this level apply only to the BGP L2VPN EVPN address family.

NOTE
The BGP L2VPN EVPN address family must be enabled for all EVPN peers.

The following configuration options are allowed under BGP L2VPN EVPN configuration mode:

- **client-to-client-reflection**
- **graceful-restart**
- **neighbor activate**
- **neighbor advertisement-interval**
- **neighbor ao**

- **neighbor description**
- **neighbor local-as**
- **neighbor maxas-limit**
- **neighbor password**
- **neighbor peer-group**
- **neighbor remote-as**
- **neighbor route-map**
- **neighbor route-reflector-client**
- **neighbor send-community**
- **neighbor shutdown**
- **neighbor soft-reconfiguration**
- **neighbors timers**
- **neighbor update-source**
- **neighbor weight**

For detailed examples of all the BGP commands configurable for the BGP L2VPN EVPN address family, refer to the *RUCKUS FastIron Command Reference*.

Configuring the BGP L2VPN EVPN Address Family

After you enter BGP L2VPN EVPN address family configuration mode, you can configure a variety of BGP EVPN options. Perform the following steps to enter BGP L2VPN EVPN address family configuration mode.

NOTE
The BGP L2VPN EVPN address family must be enabled for all BGP EVPN peers.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enable BGP routing.

```
device(config)# router bgp
```

3. Enable BGP L2VPN EVPN address family configuration mode.

```
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-l2vpn-evpn)#
```

The following example enables BGP L2VPN EVPN address family configuration mode so that a number of BGP EVPN options can be configured.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-l2vpn-evpn)#
```

Enabling Client-to-Client Reflection for the BGP L2VPN EVPN Address Family

Client-to-client reflection enables routes from one client to be reflected to other clients by the host device on which it is configured. Client-to-client reflection is enabled by default for the BGP L2VPN EVPN address family.

The following task re-enables client-to-client reflection for the BGP L2VPN EVPN address family if it has been disabled.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enable BGP routing.

```
device(config)# router bgp
```

3. Configure the autonomous system number (ASN).

```
device(config-bgp-router)# local-as 656
```

4. Specify the ASN for the neighbor device.

```
device(config-bgp-router)# neighbor 1.1.1.35 remote-as 656
```

5. Enable BGP L2VPN EVPN address family configuration mode.

```
device(config-bgp-router)# address-family l2vpn evpn
```

6. Activate the neighbor to enable the exchange of information with the neighbor.

```
device(config-bgp-l2vpn-evpn)# neighbor 1.1.1.35 activate
```

7. Enable the sending of both standard and extended attributes in updates to the specified BGP neighbor.

```
device(config-bgp-l2vpn-evpn)# neighbor 1.1.1.35 send-community both
```

8. Enable client-to-client reflection.

```
device(config-bgp-l2vpn-evpn)# client-to-client-reflection
```

The following example re-enables client-to-client reflection for the BGP L2VPN EVPN address family if it has been disabled.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 656
device(config-bgp-router)# neighbor 1.1.1.35 remote-as 656
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-l2vpn-evpn)# neighbor 1.1.1.35 activate
device(config-bgp-l2vpn-evpn)# neighbor 1.1.1.35 send-community both
device(config-bgp-l2vpn-evpn)# client-to-client-reflection
```

Configuring Graceful Restart for the BGP L2VPN EVPN Address Family

BGP graceful restart (GR) allows for restarts where BGP neighboring devices participate in the restart, helping to ensure that no route and topology changes occur in the network for the duration of the restart. The following task enables BGP GR and sets the restart time to 60 seconds for the BGP L2VPN EVPN address family.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enable BGP routing.

```
device(config)# router bgp
```

3. Configure the autonomous system number (ASN).

```
device(config-bgp-router)# local-as 656
```

4. Specify the ASN for the neighbor device.

```
device(config-bgp-router)# neighbor 1.1.1.35 remote-as 656
```

5. Enable BGP L2VPN EVPN address family configuration mode.

```
device(config-bgp-router)# address-family l2vpn evpn
```

6. Activate the neigbor to enable the exchange of information with the neigbor.

```
device(config-bgp-l2vpn-evpn)# neighbor 1.1.1.35 activate
```

7. Enable the sending of both standard and extended attributes in updates to the specified BGP neighbor.

```
device(config-bgp-l2vpn-evpn)# neighbor 1.1.1.35 send-community both
```

8. Enable GR and set the restart time.

```
device(config-bgp-l2vpn-evpn)# graceful restart restart-time 60
```

   The restart time is set to 60 seconds.

The following example enables GR and sets the restart time to 60 seconds for the BGP L2VPN EVPN address family.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 656
device(config-bgp-router)# neighbor 1.1.1.35 remote-as 656
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-l2vpn-evpn)# neighbor 1.1.1.35 activate
device(config-bgp-l2vpn-evpn)# neighbor 1.1.1.35 send-community both
device(config-bgp-l2vpn-evpn)# graceful restart restart-time 60
```

Applying a Route Map to a BGP Peer

Route maps can be applied to BGP peers, either as the inbound or outbound routing policy for neighbors under the specified address family. The following task applies a route map to an incoming route for the BGP L2VPN EVPN address family.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enable BGP routing.

```
device(config)# router bgp
```

3. Configure the autonomous system number (ASN).

```
device(config-bgp-router)# local-as 656
```

4. Specify the ASN for the neighbor device.

```
device(config-bgp-router)# neighbor 1.1.1.35 remote-as 656
```

5. Enable BGP L2VPN EVPN address family configuration mode.

```
device(config-bgp-router)# address-family l2vpn evpn
```

6. Activate the neighbor to enable the exchange of information.

```
device(config-bgp-l2vpn-evpn)# neighbor 1.1.1.35 activate
```

7. Enable the sending of both standard and extended attributes in updates to the specified BGP neighbor.

```
device(config-bgp-l2vpn-evpn)# neighbor 1.1.1.35 send-community both
```

8. Apply a route map to an incoming route.

```
device(config-bgp-l2vpn-evpn)# neighbor 10.1.1.35 route-map in myroutemap
```

   A route map named "myroutemap" is applied to an incoming route from 10.1.1.35 for the BGP L2VPN EVPN address family.

The following example applies a route map named "myroutemap" to an incoming route from 10.1.1.35 for the BGP L2VPN EVPN address family.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 656
device(config-bgp-router)# neighbor 1.1.1.35 remote-as 656
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-l2vpn-evpn)# neighbor 1.1.1.35 activate
device(config-bgp-l2vpn-evpn)# neighbor 1.1.1.35 send-community both
device(config-bgp-l2vpn-evpn)# neighbor 10.1.1.35 route-map in myroutemap
```

Configuring Neighbors as Route Reflector Clients

BGP devices can act as route reflector clients, or as route reflectors (RRs). You can configure a BGP peer as an RR client from the device that is going to reflect the routes and act as the RR. The following task configures a specified neighbor to act as an RR client for the BGP L2VPN EVPN address family.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enable BGP routing.

```
device(config)# router bgp
```

3. Configure the autonomous system number (ASN).

```
device(config-bgp-router)# local-as 656
```

4. Specify the ASN for the neighbor device.

```
device(config-bgp-router)# neighbor 1.1.1.35 remote-as 656
```

5. Enable BGP L2VPN EVPN address family configuration mode.

```
device(config-bgp-router)# address-family l2vpn evpn
```

6. Activate the neighbor to enable the exchange of information.

```
device(config-bgp-l2vpn-evpn)# neighbor 1.1.1.35 activate
```

7. Enable the sending of both standard and extended attributes in updates to the specified BGP neighbor.

```
device(config-bgp-l2vpn-evpn)# neighbor 1.1.1.35 send-community both
```

8. Configure a neighbor to act as an RR client.

```
device(config-bgp-l2vpn-evpn)# neighbor 10.1.1.35 route-reflector-client
```

   A neighbor with the IP address 10.1.1.35 is configured to act as an RR client for the BGP L2VPN EVPN address family.

The following example configures a neighbor with the IP address 10.1.1.35 to be act as an RR client for the BGP L2VPN EVPN address family.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 656
device(config-bgp-router)# neighbor 1.1.1.35 remote-as 656
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-l2vpn-evpn)# neighbor 1.1.1.35 activate
device(config-bgp-l2vpn-evpn)# neighbor 1.1.1.35 send-community both
device(config-bgp-l2vpn-evpn)# neighbor 10.1.1.35 route-reflector-client
```

Configuring BGP EVPN Route Filters

An Ethernet Virtual Private Network (EVPN) prefix list can be configured, permitting or denying specific routes. By default, routes that do not match a prefix list are learned or advertised. To prevent a route from being learned or advertised, you must configure and apply a prefix list to deny the route.

The following task creates an EVPN prefix list. The EVPN prefix list denies MAC addresses 02e0.52cb.55c8 and 02e0.52cb.55c9 and permits all other MAC addresses.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Configure an IP prefix list instance. Specify that MAC address 02e0.52cb.55c8 is denied.

```
device(config)# ip evpn-prefix-list plist deny 02e0.52cb.55c8
```

3. For the same IP prefix list instance, specify that MAC address 02e0.52cb.55c9 is denied.

```
device(config)# ip evpn-prefix-list plist deny 02e0.52cb.55c9
```

4. For the same IP prefix list instance, specify that all other MAC addresses are permitted.

```
device(config)# ip evpn-prefix-list plist permit any
```

The following example creates an EVPN prefix list that denies MAC addresses 02e0.52cb.55c8 and 02e0.52cb.55c9 and permits all other MAC addresses.

```
device# configure terminal
device(config)# ip evpn-prefix-list plist deny 02e0.52cb.55c8
device(config)# ip evpn-prefix-list plist deny 02e0.52cb.55c9
device(config)# ip evpn-prefix-list plist permit any
```

Configuring BGP EVPN Route Maps

Route maps can be used in the inbound or outbound direction for BGP peers for EVPN routes. When a route map is applied to a BGP peer, a ROUTE REFRESH message is sent. This results in the remote peer resending all the EVPN routes. If a route is already learned from a peer, the route map is applied. This results in the route being denied or the matching condition not being found. The learned route is then removed.

The following task creates a route map instance and specifies that the EVPN route type for the route map instance is 2. Only Type-2 routes are denied for the route map instance.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Create a route map instance and allow a matching pattern.

   ```
   device(config)# route-map myroutemap deny 1
   ```

3. Permit other EVPN routes.

   ```
   device(config)# route-map myroutemap permit 2
   ```

4. Configure the route map to match on the specified prefix.

   ```
   device(config-routemap myroutemap)# match evpn-route-type 2
   ```

   Only Type-2 routes (MAC/IP advertisement routes) are denied and all other route types (Ethernet Auto-Discovery (A-D) routes, Inclusive Multicast Ethernet Tag (IMET) routes, Ethernet Segment routes and IP Prefix route) are permitted. For full details on these EVPN routes, refer to the **match evpn-route-type** command in the *RUCKUS FastIron Command Reference*.

The following example specifies that the EVPN route type for a route-map instance is 2. Only Type-2 routes are denied for the route map instance. Other EVPN routes are permitted.

```
device# configure terminal
device(config)# route-map myroutemap deny 1
device(config)# route-map myroutemap permit 2
device(config-routemap myroutemap)# match evpn-route-type 2
```

The following example configures the EVPN prefix name to be used for MAC address filtering for a route map instance called "myroutemap".

```
device# configure terminal
device(config)# ip evpn-prefix-list deny 02e0.52cb.55c8
device(config)# ip evpn-prefix-list deny 02e0.52cb.55c9
device(config)# ip evpn-prefix-list plist permit any
device(config)# route-map myroutemap permit 1
device(config-routemap myroutemap)# match ip adddress evpn-prefix-list plist
```

Creating an EVPN Instance

A Layer 2 Ethernet Virtual Private Network (EVPN) instance can be created so that a device enters EVPN EVI configuration mode. After the device enters EVPN EVI configuration mode, you can configure a variety of EVPN options. Perform the following steps to enter EVPN EVI configuration mode.

The following task enables BGP L2VPN EVPN address family configuration mode.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter EVPN configuration mode.

```
device(config)# l2vpn evpn
```

3. Create an EVPN instance.

```
device(config-evpn)# evpn-instance 1 vlan-based
device(config-evpn-evi)#
```

An EVPN instance with ID 1 is created and a VLAN-based VXLAN service model is configured.

The following example creates an EVPN instance with ID 1 and specifies that a VLAN-based service model is used.

```
device# configure terminal
device(config)# l2vpn evpn
device(config-evpn)# evpn-instance 1 vlan-based
```

Configuring an EVPN Instance

After a Layer 2 Ethernet Virtual Private Network (EVPN) instance is created, the device enters EVPN EVI configuration mode, allowing a variety of EVPN options can be configured in EVPN EVI configuration mode.

The following task configures a variety of EVPN options for an EVPN instance:

- A Layer 2 VNI with an ID of 1000 is created.
- VXLAN is configured as the encapsulation type for the EVPN instance.
- The auto-generation of a route distinguisher (RD) is enabled.
- The auto-generation of import and export route-target community attributes is enabled.
- BGP is configured as the mechanism for host reachability advertisement.
- Duplicate detection for MAC addresses is enabled.
- ARP suppression is enabled.
- Unknown unicast traffic is suppressed.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter EVPN configuration mode.

```
device(config)# l2vpn evpn
```

3. Create an EVPN instance.

```
device(config-evpn)# evpn-instance 1 vlan-based
device(config-evpn-evi)#
```

An EVPN instance with ID 1 is created and a VLAN-based VXLAN service model is configured.

4. Configure a Layer 2 (L2) Virtual Network Identifier (VNI).

```
device(config-evpn-evi)# vni 1000 l2
```

A Layer 2 VNI with an ID of 1000 is created.

5. Configure VXLAN as the encapsulation type for the VNI.

```
device(config-evpn-evi)# encapsulation vxlan
```

6. Enable the auto-generation of a route distinguisher (RD) for the VNI.

```
device(config-evpn-evi)# rd auto
```

7. Enable the auto-generation of export route-target community attributes for the VNI.

```
device(config-evpn-evi)# route-target export auto
```

8. Enable the auto-generation of import route-target community attributes for the VNI.

```
device(config-evpn-evi)# route-target import auto
```

9. Configure BGP as the mechanism for host reachability advertisement for the VNI.

```
device(config-evpn-evi)# host-reachability protocol bgp
```

10. Enable duplicate detection for MAC addresses. Specify the maximum number of MAC moves allowed before timer expiry and set the MAC duplicate detection timeout interval.

```
device(config-evpn-evi)# mac duplication limit 150 expiry time 1200
```

    The maximum number of MAC moves allowed before timer expiry is 150 and the MAC duplicate detection timeout interval is 1200 seconds.

11. Enable ARP suppression for the VNI.

```
device(config-evpn-evi)# arp-suppression
```

12. Suppress unknown unicast traffic for the Layer 2 EVPN.

```
device(config-evpn-evi)# unknown-unicast-suppress
```

The following example configures a variety of EVPN options for an EVPN instance. A Layer 2 VNI with an ID of 1000 is created. VXLAN is configured as the encapsulation type for the EVPN instance. The auto-generation of a route distinguisher (RD) is enabled. The auto-generation of import and export route-target community attributes is enabled. BGP is configured as the mechanism for host reachability advertisement. Duplicate detection for MAC addresses is enabled. ARP suppression is enabled. Unknown unicast traffic is suppressed.

```
device# configure terminal
device(config)# l2vpn evpn
device(config-evpn)# evpn-instance 1 vlan-based
device(config-evpn-evi)# vni 1000 L2
device(config-evpn-evi)# encapsulation vxlan
device(config-evpn-evi)# rd auto
device(config-evpn-evi)# route-target export auto
device(config-evpn-evi)# route-target import auto
device(config-evpn-evi)# host-reachability protocol bgp
device(config-evpn-evi)# arp-suppression
device(config-evpn-evi)# unknown-unicast-suppress
```

Creating a Range of EVPN Instances

A range of Layer 2 Ethernet Virtual Private Network (EVPN) instances can be created. Perform the following steps to create a range of EVPN instances.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter EVPN configuration mode.

```
device(config)# l2vpn evpn
```

3. Create a range of EVPN instances.

```
device(config-evpn)# evpn-instance range 100 to 200 vni-start 1000 vlan-based
device(config-evpn-evi-range-100-200)#
```

   A range of EVPN instances is created, all of which utilize the VLAN-based VXLAN service model, and the first of which has its associated VXLAN network identifier (VNI) defined as 1000.

The following example configures a range of Layer 2 EVPN instances and specifies that a VNI is started.

```
device# configure terminal
device(config)# l2vpn evpn
device(config-evpn)# evpn-instance range 100 to 200 vni-start 1000 vlan-based
device(config-evpn-evi-range-100-200)#
```

ARP Suppression for BGP EVPN Overview

Address Resolution Protocol (ARP) suppression reduces the flooding of ARP messages in the EVPN network, leading to a more efficient use of core bandwidth.

ARP is responsible for mapping IPv4 addresses to MAC addresses in IP networks. ARP obtains endpoint MAC address information by utilizing the IP address information sent out in ARP broadcast requests. These broadcast requests are treated as multi-destination traffic, typically VXLAN-encapsulated, and sent to every VXLAN Tunnel End Point (VTEP) or edge device that is a member of the corresponding Layer 2 VXLAN network identifier (VNI). Responses to ARP broadcast requests are typically sent in the form of unicast packets to the requester, and the ARP traffic is scoped within the bounds of the broadcast domain represented by the Layer 2 VNI.

ARP broadcast requests are sent to the configured default gateway when a host needs to resolve the default gateway. This process facilitates the population of the MAC IP table for the edge device, ensuring that all MAC information is available in the BGP EVPN control plane protocol. Additionally, in a Route Type 2 advertisement, Layer 2, Layer 2 VNI, Route Distinguisher (RD), Route Target (RT), hosting VTEP, and associated IP information are populated. This is then distributed to all remote VTEPs.

Regular ARP broadcast requests are sent out to determine the IP-to-MAC binding of a destination endpoint. This ensures that a host device can learn information about an endpoint in the same subnet. ARP broadcast requests are flooded to all endpoints that are part of that Layer 2 VNI.

ARP snooping prevents flooding for known endpoints. When ARP snooping is employed, all ARP requests from an endpoint are redirected to the locally attached edge device. This allows for the determination of known endpoints. If the destination is known, the IP-to-MAC binding information is returned, and the local edge device performs an ARP proxy on behalf of the destination endpoint by sending a unicast ARP response toward the requester with the resolved MAC address of the known destination endpoint.

All ARP requests to known endpoints are terminated at the earliest possible point, typically the locally attached edge device, VTEP, or leaf. ARP suppression is possible because all information is known on all VXLAN VTEPs. ARP suppression differs from regular proxy ARP behavior; the router does not act as the proxy ARP endpoint in the response. Instead, the remote endpoint or host is used as the proxy, and the information is learned via either ARP snooping or the BGP EVPN control plane.

Configuring ARP Suppression for BGP EVPN

You can enable ARP suppression for an EVPN instance. For BGP EVPN, when a VLAN is mapped to a VXLAN VNI, ARP suppression extends to corresponding EVPNs, VNIs, and EVIs. When ARP suppression is enabled, snooping MAC-IP bindings of any local endpoints for that VLAN is enabled. The information is maintained in a per-VLAN MAC-IP table. In addition to MAC-IP bindings learned through ARP snooping, any MAC-IP bindings learned via the BGP EVPN control plane are also stored in the same MAC-IP table.

The following task enables ARP suppression for a a Layer 2 EVPN.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter EVPN configuration mode.

```
device(config)# l2vpn evpn
```

3. Create an EVPN instance.

```
device(config-evpn)# evpn-instance 1 vlan-based
device(config-evpn-evi)#
```

An EVPN instance with ID 1 is created and a VXLAN VLAN-based service model is configured.

4. Enable ARP suppression for the EVPN instance.

```
device(config-evpn-evi)# arp-suppression
```

The following example enables ARP suppression for an EVPN instance.

```
device# configure terminal
device(config)# l2vpn evpn
device(config-evpn)# evpn-instance 1 vlan-based
device(config-evpn-evi)# arp-suppression
```

Displaying BGP EVPN Information

You can use various **show** commands to view information about BGP EVPN.

Use one of the following commands to view BGP EVPN information. Using these commands is optional and they can be entered in any order. For more information on the full list of **show bgp evpn** commands, refer to the *RUCKUS FastIron Command Reference Guide*.

1. Enter the **show ip bgp evpn attribute-entries** command to display information about AS-path attribute entries for the L2VPN EVPN address family.

```
device> show ip bgp evpn attribute-entries

 Total number of BGP Attribute Entries: 14
1       Next Hop   : 2.2.2.8           MED      :0              Origin:IGP
        Originator:0.0.0.0             Cluster List:None
        Aggregator:AS Number :0        Router-ID:0.0.0.0       Atomic:None
        Local Pref:100                 Communities:Internet
        Extended Community: RT 1000:30 Bgp-Encap: Tunnel-Type: 8
        AS Path    : (length 0)
        AsPathLen: 0  AsNum: 0, SegmentNum: 0, Neighboring As: 0, Source As 0
        Address: 0xd777a4e8  Hash:3730 (0x1000000)
        Links: 0x0, 0x0
        Reference Counts: 1:0:1, Magic: 1
2       Next Hop   : 2.2.2.8            MED      :0              Origin:IGP
        Originator:0.0.0.0             Cluster List:None
        Aggregator:AS Number :0        Router-ID:0.0.0.0       Atomic:None
        Local Pref:100                 Communities:Internet
        Extended Community: RT 1000:31 Bgp-Encap: Tunnel-Type: 8
        AS Path    : (length 0)
        AsPathLen: 0  AsNum: 0, SegmentNum: 0, Neighboring As: 0, Source As 0
        Address: 0xd777a553  Hash:3731 (0x1000000)
        Links: 0x0, 0x0
        Reference Counts: 1:0:1, Magic: 2
```

2. Enter the **show ip bgp evpn neighbors** command to display configuration information and statistics for BGP EVPN neighbors.

```
device> show ip bgp evpn neighbors

    Total number of BGP Neighbors:1
1   IP Address: 1.1.1.11, AS: 1000 (IBGP), RouterID: 1.1.1.11, VRF: default-vrf
    State: ESTABLISHED, Time: 0h2m30s, KeepAliveTime: 60, HoldTime: 180
       KeepAliveTimer Expire in 3 seconds, HoldTimer Expire in 140 seconds
    Minimal Route Advertisement Interval: 0 seconds
       UpdateSource: Loopback 1
       RefreshCapability: Received
    Address Family : L2VPN EVPN
       SendCommunity: yes
       SendExtenedCommunity: yes
    Messages:    Open    Update  KeepAlive Notification Refresh-Req
       Sent    : 1       2       3         0            0
       Received: 1       12      3         0            0
    Last Update Time: NLRI       Withdraw          NLRI       Withdraw
                  Tx: 0h2m30s    ---           Rx: 0h2m30s    ---
    Last Connection Reset Reason:Unknown
    Notification Sent:     Unspecified
    Notification Received: Unspecified
    Neighbor NLRI Negotiation:
      Peer Negotiated IPV4  unicast  capability
      Peer Negotiated L2VPN EVPN address family
      Peer configured for IPV4 unicast  Routes
      Peer configured for L2VPN EVPN address family
    Neighbor AS4 Capability Negotiation:
    Outbound Policy Group:
       ID: 2, Use Count: 1
    TCP Connection state: ESTABLISHED
     Maximum segment size: 1436
    TTL check: value: 0
       Byte Sent:   322, Received: 1394
       Local host:  1.1.1.8, Local  Port: 179
       Remote host: 1.1.1.11, Remote Port: 8107
       ISentSeq:   44010496  SendNext:   44010819  TotUnAck:          0
       TotSent:         323  ReTrans:           0  UnAckSeq:   44010819
       IRcvSeq:  1849175076  RcvNext:  1849176471  SendWnd:       16384
       TotalRcv:       1395  DupliRcv:          0  RcvWnd:        16384
       SendQue:           0  RcvQue:            0  CngstWnd:       1436
```

3. Enter the **show ip bgp evpn routes** command to display information for the routes in the BGP route table for the L2VPN EVPN address family.

```
device> show ip bgp evpn routes

Total number of BGP EVPN Routes: 5 (R: 3, L: 2)
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
       S:SUPPRESSED F:FILTERED s:STALE
       Prefix             Next Hop        MED        LocPrf     Weight Status
Route Distinguisher: 2.2.2.8:30
1      [3]:[0]:[32]:[2.2.2.8]/72
                          2.2.2.8         0          100        0      BL
         AS_PATH: Not Applicable
2      [3]:[0]:[32]:[2.2.2.11]/72
                          2.2.2.11        0          100        0      BI
         AS_PATH: Not Applicable
Route Distinguisher: 2.2.2.8:31
3      [2]:[0]:[48]:[0001.0001.0001]/128
                          2.2.2.11        0          100        0      BI
         AS_PATH: Not Applicable
4      [3]:[0]:[32]:[2.2.2.8]/72
                          2.2.2.8         0          100        0      BL
         AS_PATH: Not Applicable
5      [3]:[0]:[32]:[2.2.2.11]/72
                          2.2.2.11        0          100        0      BI
         AS_PATH: Not Applicable
```

4. Enter the **show ip bgp evpn routes mac** command to display BGP EVPN information for MAC routes.

```
device> show ip bgp evpn routes mac

Searching for matching routes, use ^C to quit...
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
       S:SUPPRESSED F:FILTERED s:STALE
       Prefix             Next Hop        MED        LocPrf     Weight Status
Route Distinguisher: 2.2.2.8:31
1      [2]:[0]:[48]:[0001.0001.0001]/128
                          2.2.2.11        0          100        0      BI
         AS_PATH: Not Applicable
```

5. Enter the **show ip bgp evpn routes imet** command to display BGP EVPN information for P-Multicast Service Interface Tunnel (PMSI Tunnel) tunnel path attributes associated with a route.

```
device> show ip bgp evpn routes imet

Searching for matching routes, use ^C to quit...
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
       S:SUPPRESSED F:FILTERED s:STALE
       Prefix             Next Hop        MED        LocPrf     Weight Status
Route Distinguisher: 2.2.2.8:30
1      [3]:[0]:[32]:[2.2.2.8]/72
                          2.2.2.8         0          100        0      BL
         AS_PATH: Not Applicable
2      [3]:[0]:[32]:[2.2.2.11]/72
                          2.2.2.11        0          100        0      BI
         AS_PATH: Not Applicable
Route Distinguisher: 2.2.2.8:31
3      [3]:[0]:[32]:[2.2.2.8]/72
                          2.2.2.8         0          100        0      BL
         AS_PATH: Not Applicable
4      [3]:[0]:[32]:[2.2.2.11]/72
                          2.2.2.11        0          100        0      BI
         AS_PATH: Not Applicable
```

6. Enter the **show ip bgp evpn routes detail** command to display detailed information for the routes in the BGP route table for the L2VPN EVPN address family.

```
device> show ip bgp evpn routes detail

Total number of BGP EVPN Routes: 5 (R: 3, L: 2)
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
       S:SUPPRESSED F:FILTERED s:STALE
Route Distinguisher: 2.2.2.8:30
1       Prefix: [3]:[0]:[32]:[2.2.2.8]/72,  Status: BL,  Age: 18h12m30s
         NEXT_HOP: 2.2.2.8, Learned from Peer: Local Router
          LOCAL_PREF: 100,  MED: 0,  ORIGIN: igp,  Weight: 0
                  AS_PATH: Not Applicable
           PMSI Tunnel Type: Ingress-Replication, L2VNI: 10030, Endpoint: 2.2.2.8
            Extended Community: RT 1000:30 Bgp-Encap: Tunnel-Type: 8
            Adj_RIB_out count: 1,  Admin distance 1
2       Prefix: [3]:[0]:[32]:[2.2.2.11]/72,  Status: BI,  Age: 18h11m14s
         NEXT_HOP: 2.2.2.11, Learned from Peer: 1.1.1.11 (1000)
          LOCAL_PREF: 100,  MED: 0,  ORIGIN: igp,  Weight: 0
                  AS_PATH: Not Applicable
           PMSI Tunnel Type: Ingress-Replication, L2VNI: 10030, Endpoint: 2.2.2.11
            Extended Community: RT 1000:30 Bgp-Encap: Tunnel-Type: 8
Route Distinguisher: 2.2.2.8:31
3       Prefix: [2]:[0]:[48]:[0001.0001.0001]/128,  Status: BI,  Age: 18h11m14s
         NEXT_HOP: 2.2.2.11, Learned from Peer: 1.1.1.11 (1000)
          LOCAL_PREF: 100,  MED: 0,  ORIGIN: igp,  Weight: 0
                  AS_PATH: Not Applicable
           L2VNI: 10031
            Extended Community: RT 1000:31 Bgp-Encap: Tunnel-Type: 8 MAC-Mobility: Static-flag: 1
Seq-no: 0
4       Prefix: [3]:[0]:[32]:[2.2.2.8]/72,  Status: BL,  Age: 18h12m30s
         NEXT_HOP: 2.2.2.8, Learned from Peer: Local Router
          LOCAL_PREF: 100,  MED: 0,  ORIGIN: igp,  Weight: 0
                  AS_PATH: Not Applicable
           PMSI Tunnel Type: Ingress-Replication, L2VNI: 10031, Endpoint: 2.2.2.8
            Extended Community: RT 1000:31 Bgp-Encap: Tunnel-Type: 8
            Adj_RIB_out count: 1,  Admin distance 1
5       Prefix: [3]:[0]:[32]:[2.2.2.11]/72,  Status: BI,  Age: 18h11m14s
         NEXT_HOP: 2.2.2.11, Learned from Peer: 1.1.1.11 (1000)
          LOCAL_PREF: 100,  MED: 0,  ORIGIN: igp,  Weight: 0
                  AS_PATH: Not Applicable
           PMSI Tunnel Type: Ingress-Replication, L2VNI: 10031, Endpoint: 2.2.2.11
            Extended Community: RT 1000:31 Bgp-Encap: Tunnel-Type:
```

7. Enter the **show ip bgp evpn summary** command to display summarized information for the L2VPN EVPN address family.

```
device> show ip bgp evpn summary
  BGP Summary
  Router ID: 1.1.1.8   Local AS Number: 1000
  Confederation Identifier: not configured
  Confederation Peers:
  Maximum Number of IP ECMP Paths Supported for Load Sharing: 1
  Number of Neighbors Configured: 1, UP: 1
  Number of Routes Installed: 17, Uses 2210 bytes
  Number of Routes Advertising to All Neighbors: 2 (2 entries), Uses 192 bytes
  Number of Attribute Entries Installed: 14, Uses 1386 bytes
  Neighbor Address  AS#         State   Time          Rt:Accepted Filtered Sent     ToSend
  1.1.1.11          1000        ESTAB   0h16m37s      15          0        2        0
```

8. Enter the **show l2vpn evpn evi all** command to display information for all Layer 2 EVPN instances (EVIs).

```
device> show l2vpn evpn evi all
Total number of Evpn Instances: 1
EVI         State  VNI       Vlan  Tunnel-Ip        RT-export        RT-import        RD
100         UP     1000      10    1.1.1.1          10000:1000       10000:1000       1.1.1.1:10
200         UP     2000      20    1.1.1.1          10000:2000       10000:2000       1.1.1.1:20
```

9. Enter the **show l2vpn evpn evi** command, and specify an EVI, to display information for the specified EVI.

```
device> show l2vpn evpn evi 100

RD               : 159.159.159.159:100(auto)
RT import list   : 100:100
RT export        : 100:100
Service model    : vlan-based
State            : UP
Replication Type : Ingress-replication
Encapsulation    :
ARP suppression  : Yes
Unknown unicast suppress: No
Overlay          : loopback
L2 VNI           : 100
L3 VNI           : None
Tunnel Ip        : 159.159.159.159
Mac move limit   : 5
Mac move expiry  : 180
MAC re-advertise : No
```

10. Enter the **show l2vpn evpn evi** command with the **detail** keyword, and specify an EVI, to display detailed information for the EVI.

```
device> show l2vpn evpn evi 100 detail
RD               : 159.159.159.159:100(auto)
RT import list   : 100:100
RT export        : 100:100
Service model    : vlan-based
State            : UP
BGP              : Advertised
Replication Type : Ingress-replication
BUM Protocol     : BGP
Encapsulation    :
ARP suppression  : Yes
Unknown unicast suppress: No
Overlay          : loopback
Tunnel Ip        : 159.159.159.159
Mac move limit   : 5
Mac move expiry  : 180
MAC re-advertise : No
Debug info flag  : Off
Debug itc flag   : Off
Debug trace flag : Off
Debug error flag : Off
L2 VNI           : 100
L3 VNI           : None
VRF              : Tenant VRF
```

11. Enter the **show l2vpn evpn mac-ip-table vlan** command, and specify a VLAN, to display information for an EVPN MAC IP table for the VLAN.

```
device> show l2vpn evpn mac-ip-table vlan 10

No.   IP Address         MAC Address    Flags     Port              VLAN
1     192.168.10.100     d4c1.9e17.8d27 DL        e 1/1/1            10
2     192.168.10.200     d4c1.9e17.91f7 DR        VxL-200.1.1.1      10
Flags: D (Dynamic) S (Static) L(Local) R(Remote) Ad(Advertised)
```

BGP EVPN Multihoming

BGP EVPN Multihoming Overview

BGP-EVPN enhances redundancy for Virtual Extensible LAN (VXLAN) by providing multiple, distinct uplink (or Provider Edge (PE)) connections for customer-edge (CE) devices. This multi-attachment capability offers redundant, all-active connectivity to the EVPN core network and extends this redundancy across the entire network connected to the core.

BGP EVPN MH provides a solution that helps to address common challenges associated with all-active redundancy, such as duplication and looping of Broadcast, Unknown unicast, and Multicast (BUM) traffic.

Important Terminology for BGP EVPN Multihoming

The below are common terms used in reference to BGP EVPN Multihoming:

- **BUM**: Broadcast, Unknown unicast, and Multicast
- **CE**: Customer Edge
- **DF**: Designated Forwarder
- **ES**: Ethernet Segment
- **EVPN**: Ethernet Virtual Private Network. A common implementation is Ethernet over VXLAN. This is the implementation outlined in this chapter.
- **EVI**: Ethernet Virtual Instance
- **LAG**: Link Aggregation Group
- **MAC**: Media Access Control
- **PE**: Provider Edge
- **RD**: Route Distinguisher

- **SH**: Split Horizon
- **VE**Virtual Entity: A set of virtualized network resources or devices that are connected to multiple uplinks or network paths.
- **VNI**: Virtual Network Identifier
- **VTEP**: Virtual Tunnel End Point. In BGP EVPN terms, a VTEP can also be referred to as a leaf.
- **VXLAN**: Virtual Extensible Local Area Network

  NOTE
  For more information on VXLAN-related terminology, refer to VXLAN on page 331.

  NOTE
  For more information on general BGP EVPN terminology, refer to BGP EVPN on page 363.

Requirements

- BGP EVPN Multihoming is supported for RUCKUS ICX 7850 and ICX 7550 devices only.

BGP EVPN Multihoming Considerations

- All BGP EVPN singlehoming configurations are supported for BGP EVPN Multihoming. Refer to BGP EVPN on page 363 for more information.
- In a BGP EVPN singlehomed environment, when a port goes down, all locally learned Media Access Control (MAC) addresses on that port become unreachable due to the absence of alternative paths. As a result, the Virtual Tunnel Endpoint (VTEP) initiates the withdrawal of these MAC addresses to other VTEPs in the network. However, in a BGP EVPN multihomed environment, if an Ethernet Segment (ES) becomes unavailable on one VTEP, the local MAC addresses associated with that ES may still be reachable through another local or multihomed member VTEP where the ES is still active. This redundancy ensures continued access to those MAC addresses through the remaining operational VTEP.
- A maximum of two Virtual Tunnel End Points (VTEPs) are supported to participate in multihoming (Dualhoming).
- It is recommended to configure anycast gateways on multihomed (MH) Virtual Entities (VEs) before configuring BGP EVPN MH. For more information on anycast gateways refer to Anycast Gateway with VXLAN on page 359

Prerequisites

- Devices must support BGP EVPN and VXLAN protocols. For more details on BGP EVPN and VXLAN device support for RUCKUS ICX devices, refer to the *RUCKUS FastIron Features and Standards Support Matrix*.
- While an Interior Gateway Protocol (IGP) such as OSPF or RIP can be enabled for underlay routing, it is recommended to employ OSPF for optimal underlay routing.
- VLANs and VXLANs must be configured in advance.
- Define the VTEPs (VXLAN Tunnel Endpoints) within the VXLAN topology.

BGP EVPN Multihoming Limitations

- The maximum number of Ethernet Segment LAGs that can be locally configured on one device is 16. This includes both static and dynamic LAGs.
- The maximum number of dual-homed sites is eight. The maximum number of VTEPs is sixteen so that a maximum of eight dualhomed sites can be configured..

- The maximum number of member EVIs for one ES-LAG is capped at 100. This limit is implemented to maintain control over the number of Ethernet Auto-Discovery (EAD) routes generated in the network.
- IPv6 is not supported.
- eBGP is not supported.

BGP EVPN Multihoming Configuration

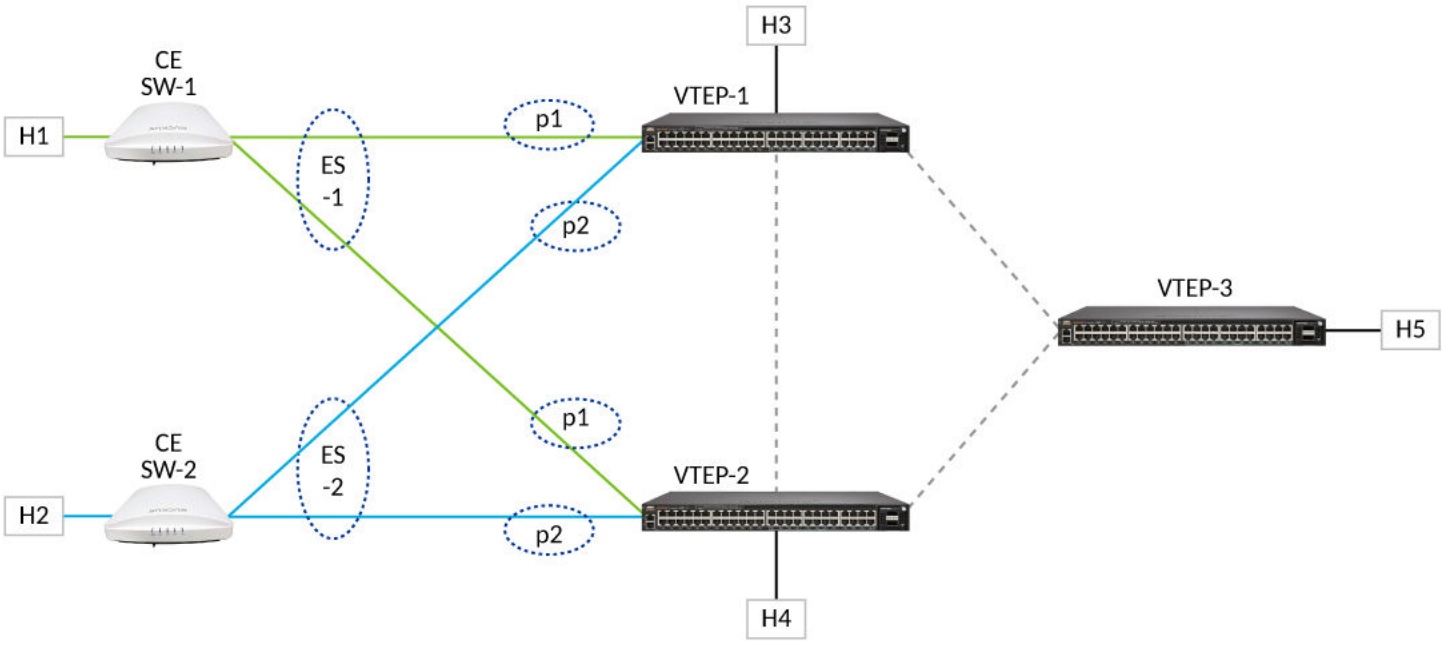
There are four steps involved for configuring Border Gateway Protocol (BGP) Ethernet Virtual Private Network (EVPN) multihoming. These steps are similar to the steps for BGP EVPN singlehoming. Refer to these sections for more information:

- Configuring the Underlay Network on page 366
- Configuring iBGP EVPN Peering Between VTEPs on page 368
- Configuring EVPN Instances for the Overlay Network on page 369 (this step is not required for Route Reflectors (RR)).
- Configuring the Overlay for VXLAN Routing on page 371

BGP EVPN Multihoming Network Topology Overview

The illustration below shows a BGP EVPN multihoming typical network topology where a Customer Edge (CE) switch connects to two Virtual Tunnel End Points (VTEPs) for redundancy and load balancing.

FIGURE 114 BGP EVPN Mutlihoming Network Typology



In this BGP EVPN multihoming network topology, the key components are:

- **CE SW-1**: Multihomed to both *VTEP-1* and *VTEP-2* on Ethernet Segment *ES 1*.
- **VTEP-1 and VTEP-2**: Connected to CE SW-1 via ES-LAG (Ethernet Segment Link Aggregation Group).

- **Ports**:
  - Port *p1* on both VTEPs is part of *ES 1*.
  - Port *p2* on both VTEPs is part of *ES 2*
- **Single-Homed Hosts**: Host *H3* is connected to *VTEP-1*, and Host *H4* is connected to *VTEP-2*.
- **Remote Peer**: *VTEP-3*, which may have its own set of ESs and multihomed connectivity to CE devices. In this example, only a single-homed *H5* connection is shown.

Network Behavior

The key points to note about this BGP EVPN multihoming network topology are:

- **CE SW-1/CE SW-2**: These see the connection as a standard LAG and are unaware of the ES on the other side.
- **VTEP-1, VTEP-2, and VTEP-3**: These run BGP EVPN among themselves. BGP peering occurs in the underlay network, either as a full mesh or through a Route Reflector (RR).

Ethernet Segments

When a customer site is connected to one or more Provider Edge (PE) routers via a set of Ethernet links, this set of links constitutes an Ethernet segment (ES).

For a multihomed site, each Ethernet segment (ES) is identified by a unique, non-zero identifier called an Ethernet Segment Identifier (ESI). All-Active multihoming over an ES allows full utilization of all Customer Edge (CE) to PE links for traffic to and from that CE.

Unique Identifiers Assignment for Ethernet Segment LAGs

For multihomed BGP EVPN sites, Ethernet segments need a unique 10-byte identifier called the Ethernet Segment Identifier (ESI).

The ESI is structured as follows:

- Type (1 byte): Specifies the format of the ESI Value.
- Value (9 bytes): Varies based on the ESI Type.

The following ESI types are supported:

- **Type 0 (T=0x01)**: Indicates an arbitrary 9-byte ESI value, which is manually configured and managed by the operator.
- **Type 1 T=0x00**: Auto-generated using Link Aggregation Control Protocol (LACP) parameters:
  - CE LACP System MAC address (6 bytes): Encoded in the high-order 6 bytes of the ESI Value field.
  - CE LACP Port Key (2 bytes): Encoded in the 2 bytes next to the System MAC address.
  - Remaining byte set to 0x00
- **Type 3 T=0x03**: Based on a MAC address and a local discriminator value:
  - System MAC address (6 bytes): Must be the same across all multihomed PEs.
  - Local Discriminator value (3 bytes): Configured to form the complete ESI value.

Ethernet Segment Configuration and Management

Ethernet Segments (ES) local to a device are configured through the command line interface (CLI) and stored in the ES Manager (ES-Mgr) module. The ES-Mgr handles multihoming configuration information, including an ES identified by a unique ID, the ES Type, associated Link Aggregation Group (LAG) ID, and the redundancy mode.

The following configurations are supported:

- **Redundancy Modes**: All-Active and Single-Active. In All-Active mode, all member links of the LAG are operational. In Single-Active mode, only one member link is operational. Refer to the Ethernet Segment Redundancy Mode on page 393 section for more information.
- **LAG Types**: Static and Dynamic.
- **ES Types**: Type-0, Type-1, and Type-3. All three types support Dynamic LAG. Static LAG supports only Type-0 and Type-3. Refer to the Unique Identifiers Assignment for Ethernet Segment LAGs on page 392 section for more information.

The ES-Mgr performs the following activities:

- For Dynamic LAGs:
  - For Type-1 Ethernet Segment Identifiers (ESIs), the ES-Mgr exchanges Media Access Control (MAC) address and Port Key values with the Customer Edge (CE) device via Link Aggregation Control Protocol (LACP). It constructs the ESI by combining the ES Type with the received MAC address and Port Key values. It stores and shares the constructed ESI with Border Gateway Protocol (BGP) once the dynamic LAG is up.
  - For Type-0 or Type-3 ESIs, constructs the ESI as described in the Unique Identifiers Assignment for Ethernet Segment LAGs on page 392 section.
- For Static LAGs:
  - Computes the unique 10-byte ESI by combining the local ES Type, MAC address, and local discriminator value.
  - Shares the computed ESI with BGP.
- Maintains two Finite State Machines (FSMs) to manage multihoming configurations, including the exchange of MAC and Port Key values and ESI construction.
- Associates each ES-LAG with a physical LAG interface. When the LAG interface joins a Virtual Local Area Network (VLAN), the ES becomes a member of the corresponding Ethernet Virtual Instance (EVI).
- Constructs and sends (ES, EVI) mappings to BGP based on the LAG interface's VLAN membership.
- Keeps VLAN mappings without EVI outside the EVPN domain.
- Sends all local ES information, including ES parameters, operational state, and (ES, EVI) mappings, to BGP and notifies BGP of any changes.

Ethernet Segment Redundancy Mode

Ethernet Segments (ES) can be configured to operate in all-active or single-active mode for a Layer 2 Virtual Private Network (L2VPN) using Ethernet Virtual Private Network (EVPN).

When all-active mode is configured for an ES, the ES is simultaneously UP on all member VTEPs. Traffic is distributed across all VTEPs, providing load balancing and redundancy. All-active mode is most commonly used in a multihoming environment.

When single-active mode is configured for an ES, the ES is physically UP on only one member VTEP at a time and is DOWN on the other member VTEPs. If the ES goes down on the current active VTEP, another member VTEP becomes active and brings the ES UP. This operational state applies to the entire ES for all member VLANs.

Configuring an Ethernet Segment

Ethernet Segment (ES) type, redundancy mode, and a LAG type can be configured for an ES for a Layer 2 Virtual Private Network (L2VPN) using Ethernet Virtual Private Network (EVPN).

The following task configures an ES as type 0, with a dynamic LAG, to operate in all-active mode (so that all links within the ES are operational).

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Configure a dynamic LAG.

   ```
   device(config)# lag 100 dynamic id 100
   ```

   A dynamic LAG with an ID of 100 is created.

3. Return to global configuration mode.

   ```
   device(config-lag-100)# exit
   ```

4. Enter EVPN configuration mode.

   ```
   device(config)# l2vpn evpn
   ```

5. Configure an ES.

   ```
   device(config-evpn)# ethernet-segment 1
   device(config-evpn-es-1)#
   ```

   An ES is configured and ES configuration mode is enabled.

6. Configure an ES type.

   ```
   device(config-evpn-es-1)# es-type type-0 11:22:33:44:55:66:77:88:99 4444.5555.6666 100
   ```

   An ES type 0 is configured. A local discriminator (11:22:33:44:55:66:77:88:99) is configured, and a MAC address (4444.5555.6666) is specified. The local discriminator value for ES value is set to 100. These identifiers uniquely identify the ES type 0 within the network configuration.

7. Associate a LAG with the ES.

   ```
   device(config-evpn-es-1)# lag 100
   ```

   Associates the ES with the previously configured dynamic LAG.

8. Configure the redundancy mode for the ES.

   ```
   device(config-evpn-es-1)# redundancy-mode all-active
   ```

   The ES is configured to operate in all-active mode so that all links within the ES are operational.

The following example configures ES as ES type 0. A dynamic LAG is configured for the ES and the ES is configured to operate in all-active mode so that all links within the ES are operational.

```
device# configure terminal
device(config)# lag 100 dynamic id 100
device(config-lag-100)# exit
device(config)# l2vpn evpn
device(config-evpn)# ethernet-segment 1
device(config-evpn-es-1)# es-type type-0 11:22:33:44:55:66:77:88:99 4444.5555.6666 100
device(config-evpn-es-1)# lag 100
device(config-evpn-es-1)# redundancy-mode all-active
```

Aliasing and Backup Paths

Aliasing and backup paths are important for BGP EVPN multihoming. With aliasing, a remote Provider Edge (PE) device can utilize all multihomed PE devices as the next-hop for a Customer Edge (CE) device, even if it has learned the CE Media Access Control (MAC) address from only one of those PE devices. This is possible because multihomed PE devices can signal reachability to that MAC address's Ethernet Virtual Private Network (VPN) Instance (EVI) or Ethernet Segment (ES) via Ethernet VPN (EVPN) routes. Aliasing ensures that multiple paths to a destination are utilized effectively, providing redundancy.

Backup Path is a closely related functionality applicable in single-active redundancy mode. Aliasing and Backup Path functionality can also be utilized by singlehomed PE devices that do not support all multihoming features.

Fast Convergence and Mass Withdrawal

When a link failure occurs between a Customer Edge (CE) device and a Provider Edge (PE) device, all remote PE devices are notified through a BGP (Border Gateway Protocol) withdrawal message of a single Ethernet Virtual Private Network (EVPN) route.

This mechanism enables the remote PE devices to reprogram the next-hop for every Media Access Control (MAC) address associated with the failed link without waiting for individual MAC withdrawal messages.

Designated Forwarder

When a Customer Edge (CE) device is multihomed to a set of Provider Edge (PE) devices on an Ethernet Segment (ES) operating in all-active redundancy mode, one of these PE devices is selected as the Designated Forwarder (DF) for a given ES and Ethernet VPN Instance (EVI). The DF is responsible for sending Broadcast, Unknown Unicast, and Multicast (BUM) traffic to the CE device. The DF eliminates duplication of BUM packets in the network-to-access direction by preventing the other PE devices on the ES from delivering the same multi-destination traffic to the CE device.

Split Horizon

In a multihomed network with a set of Provider Edge (PE) devices on an all-active Ethernet Segment (ES), when a Customer Edge (CE) device sends Broadcast, Unknown Unicast, and Multicast (BUM) packets to a non-Designated Forwarder (non-DF) PE, that non-DF PE forwards the packet to all other PEs in the Ethernet Virtual Private Network (EVI), including the Designated Forwarder (DF) PE for that ES.

To prevent duplication of packets, the DF PE must drop the BUM packet rather than forwarding it back to the CE. This process is known as split-horizon filtering.

For an Ethernet Virtual Private Network Instance (EVI), if a BUM packet is received from a CE, the PE acting as the DF must send the packet to every ES for which it is the DF, except the one from which it originated. Additionally, the PE must send the packet to all other PEs in the EVI. Conversely, if the BUM packet is received from another PE (for example, PE2), the PE (for example, PE1) must only send the packet on the ESs where it is the DF and should avoid sending it to other PEs, as the other PE (PE2) has already handled that. Split Horizon prevents forwarding loops for BUM packets.

Local Bias

Local bias involves replicating multi-destination traffic received from the access side to all directly attached Ethernet Segments (ESs), regardless of the Designated Forwarder (DF) election state.

This behavior differs from the default split-horizon (SH) rule and is specifically applicable to Border Gateway Protocol (BGP) Ethernet Virtual Private Network (EVPN) with Virtual Extensible LAN (VXLAN) tunnel encapsulation.

Protocol Extension

In Protocol Extension encapsulation, the ingress Provider Edge (PE) device can use the Broadcast, Unknown Unicast, and Multicast (BUM) bit to indicate that BUM traffic has been replicated at the ingress.

This practice is essential because the MAC tables between the ingress and egress PE devices may sometimes be out of sync, leading to discrepancies in the handling of unknown unicast traffic.

Ethernet Segment Routes (Type 4)

In a Border Gateway Protocol (BGP) Ethernet Virtual Private Network (EVPN) multi-homing environment, Type 4 routes, or Ethernet Segment routes (ESRs), are crucial for the Designated Forwarder (DF) Election process, ensuring efficient data delivery.

The Ethernet Segment Identifier (ESI) and Route Distinguisher (RD) are key components that ensure the uniqueness and stability of these routes.

The table below outlines the fundamental elements of ESRs, including the message fields and their descriptions, to clarify each component's role within the ES route framework.

TABLE 29 Ethernet Segment Routes

Message	Description
RD (8 Bytes)	A unique type-1 Route Distinguisher that identifies network segments.
Key: ESI (10 Bytes)	The identifier for the Ethernet Segment, unique within the network but shared across two VTEPs.
Key: IP Length (1 Byte)	Indicates whether the IP address is 4 or 16 bytes long, important for network configuration.
Key: Originator IP	The IP address of the tunnel's origin, crucial for routing and network management.
Tunnel Encapsulation Extended Community	Defines the tunnel's encapsulation protocol, ensuring all devices use a consistent method for data transport.
ES Import Extended Community	A Route Target (/RT) that helps in the initial network segment filtering, though it may not uniquely identify each segment.

Ethernet Auto Discovery Route (Type 1)

Ethernet Auto-Discovery (EAD) Routes are essential for simplifying network configurations and ensuring stable connectivity within Border Gateway Protocol (BGP) Ethernet Virtual Private Network (EVPN) multi-homing architectures. They facilitate the identification and association of Ethernet Segments (ESs), making network management more intuitive.

The table below outlines the fundamental elements of EAD-ES routes, including the message fields and their descriptions, to clarify each component's role within the ES route framework.

TABLE 30 EAD Sub-type 1 (EAD-ES) Routes

Message	Description
RD (8 Bytes)	A unique identifier that differentiates network segments.
Key: ESI (10 Bytes)	The identifier for the ES, unique within the network and shared across VTEPs.
Key: Ethernet Tag ID (4 Bytes)	Set to the maximum value to indicate specific tagging behavior.
Label/L2VNI	Used for routing; typically set to 0 in this context.
Tunnel Encapsulation Extended Community	Defines the tunnel's encapsulation protocol, ensuring consistent data transport.

TABLE 30 EAD Sub-type 1 (EAD-ES) Routes (continued)

Message	Description
ES Label Extended Community	Indicates the operational mode of the ES, important for traffic handling.
List of EVI-RTs	Carries membership information for all EVIs associated with the ES.

The table below outlines the fundamental elements of EAD-EVI routes, including the message fields and their descriptions, to clarify each component's role within the ES route framework.

TABLE 31 EAD Sub-type 2 (EAD-EVI) Routes

Message	Description
RD (8 Bytes)	Similar to EAD-ES, it serves as a unique network segment identifier.
Key: ESI (10 Bytes)	Identical to EAD-ES, it marks the ES within the network.
Key: Ethernet Tag ID (4 Bytes)	Set to 0 to support VLAN-based service models.
Label/L2VNI	In MPLS-EVPN, this value is crucial for packet forwarding.
Tunnel Encapsulation Extended Community	Specifies the encapsulation protocol for creating tunnels in a multi-homing environment.
EVI-RT Extended Community	Targets a specific EVI, providing route targeting information.

Advertising and Learning Multihomed MAC Addresses and Snooped ARP Entries

Multihomed Media Access Control (MAC) and MAC/IP refers to a local MAC address or a snooped Address Resolution Protocol (ARP) entry that has been learned over a local Ethernet Segment (ES). In singlehomed (SH) scenarios, locally learned MAC entries are advertised by Route Type-2 (RT-2). Upon processing the route, the receiver learns about the egress Virtual Tunnel Endpoint (VTEP) for the MAC, but it does not have information about the egress port on the egress VTEP.

In multihomed (MH) scenarios, MH MAC and MAC/IP information is also carried in the same RT-2 message, but it includes the Ethernet Segment Identifier (ESI) for the ES associated with the MAC entry. This means that, in addition to the egress VTEP, the remote VTEP is aware of the outgoing ES on the egress VTEP. This information is then used in features such as aliasing and fast-convergence.

The table below outlines the fundamental elements of MAC/IP routes or RT-2 for MH:

TABLE 32 MAC/IP Route or Type-2 (RT-2) for SH and MH

Message	Description
RD (8B)	Route Distinguisher (RD) for the Ethernet Virtual Instance (EVI). This is the same in both SH and MH scenarios.
ESI (10B)	ESI. For SH MAC, this value is always set to 0. For MH MAC, the ESI value is included.
Key: Ethernet Tag ID (4B)	Identifier for the Ethernet Tag. This is the same in both SH and MH scenarios.
Key: MAC Address Length (1B)	Length of the MAC address. This is the same in both SH and MH scenarios.
Key: MAC Address (6B)	The actual MAC address. This is the same in both SH and MH scenarios.
Key: IP Address Length (1B)	Length of the IP address. This is the same in both SH and MH scenarios.
Key: IP Address	The actual IP address. This is the same in both SH and MH scenarios.
Label1/L2VNI	Label for Layer 2 Virtual Network Identifier (VNI). This is the same in both SH and MH scenarios.
Label2/L3VNI	Label for Layer 3 VNI. This is the same in both SH and MH scenarios.

TABLE 32 MAC/IP Route or Type-2 (RT-2) for SH and MH (continued)

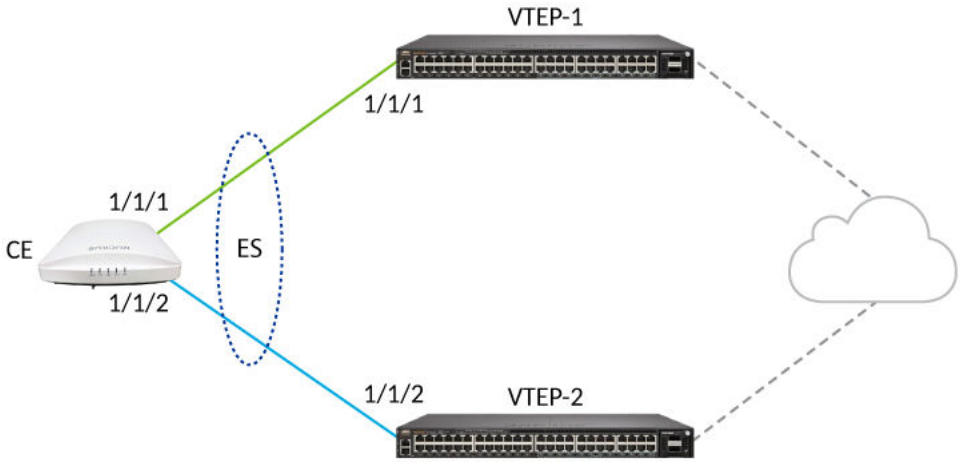
Message	Description
EVI-RT Extended Community	Extended community for the EVI Route Target (RT). This is the same in both SH and MH scenarios.

BGP EVPN Multihoming Configuration Example

The following illustration shows a typical BGP EVPN multihoming configuration for a Customer Edge (CE) device with the multi-homed nodes VXLAN Tunnel Endpoint (VTEP) 1 and VTEP 2 of an Ethernet Segment (ES).

Both VTEPs are configured identically, sharing the same MAC and port-key combination with the CE. This setup enables the CE to aggregate both connected links. The only configuration required on the CE is the creation of a LAG that connects to both VTEP 1 and VTEP 2.

FIGURE 115 BGP EVPN Multihoming Configuration Example



Configuring BGP EVPN Multihoming for a CE Device

The *BGP EVPN Multihoming Configuration Example* figure shows a typical BGP EVPN multihoming configuration at the Customer Edge (CE) device with the multi-homed nodes VXLAN Tunnel Endpoint (VTEP) 1 and VTEP 2 of an Ethernet Segment (ES). Both VTEPs are configured identically to share the same MAC and port-key combination with the CE. This setup allows the CE to aggregate both connected links at its end. The following task sets the configurations for VTEP-1.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Configure a dynamic LAG.

   ```
   device(config)# lag 100 dynamic id 100
   ```

   A dynamic LAG with an ID of 100 is created.

3. Specify an Ethernet port to be part of the LAG.

```
device(config-lag-100)# ports ethernet 1/1/1
```

Ethernet port 1/1/1 is added to the LAG with the identifier 100, allowing it to participate in the aggregated link.

4. Enable the Ethernet port.

```
device(config-lag-100)# enable ethernet 1/1/1
```

Ethernet port 1/1/1 is enabled within the LAG so that it is operational and ready to handle traffic as part of the LAG.

5. Return to global configuration mode.

```
device(config-lag-100)# exit
```

6. Enable EVPN configuration mode.

```
device(config)# l2vpn evpn
```

7. Configure an ES.

```
device(config-evpn)# ethernet-segment 1
device(config-evpn-es-1)#
```

An ES is configured and ES configuration mode is enabled.

8. Configure an ES type.

```
device(config-evpn-es-1)# es-type type-1 1234.1234.1234 1234
```

An ES type 1 is configured with an ESI of 1234.1234.1234 1234.

9. Configure a LAG for the ES.

```
device(config-evpn-es-1)# lag 100
```

Associates the ES with the previously configured dynamic LAG.

10. Configure the redundancy mode for the ES.

```
device(config-evpn-es-1)# redundancy-mode all-active
```

The ES is configured to operate in all-active mode so that all links within the ES are operational.

11. Return to global configuration mode.

```
device(config-evpn-es-1)# exit
device(config-evpn)# exit
device(config)#
```

12. Configure a VLAN.

```
device(config)# vlan 100
```

13. Assign VLAN 100 to LAG 100 as a tagged VLAN.

```
device(config-vlan-100)# tag lag 100
```

Traffic from VLAN 100 is allowed to be transmitted across the LAG while retaining its VLAN tagging information.

Example Configuration for VTEP-1 and VTEP-2

VTEP-1:

```
device# configure terminal
device(config)# lag 100 dynamic id 100
device(config-lag-100)# ports ethernet 1/1/1
device(config-lag-100)# enable ethernet 1/1/1
device(config-lag-100)# exit
device(config)# l2vpn evpn
device(config-evpn)# ethernet-segment 1
device(config-evpn-es-1)# es-type type-1 1234.1234.1234 1234
device(config-evpn-es-1)# lag 100
device(config-evpn-es-1)# redundancy-mode all-active
device(config-evpn-es-1)# exit
device(config-evpn)# exit
device(config)# vlan 100
device(config-vlan-100)# tag lag 100
```

VTEP-2 :

```
device# configure terminal
device(config)# lag 100 dynamic id 100
device(config-lag-100)# ports ethernet 1/1/2
device(config-lag-100)# enable ethernet 1/1/2
device(config-lag-100)# exit
device(config)# l2vpn evpn
device(config-evpn)# ethernet-segment 1
device(config-evpn-es-1)# es-type type-1 1234.1234.1234 1234
device(config-evpn-es-1)# lag 100
device(config-evpn-es-1)# redundancy-mode all-active
device(config-evpn-es-1)# exit
device(config-evpn)# exit
device(config)# vlan 100
device(config-vlan-100)# tag lag 100
```

CE :

```
device# configure terminal
device(config)# lag 100 dynamic id 100
device(config-lag-100)# ports ethernet 1/1/1 ethernet 1/1/2
device(config-lag-100)# enable ethernet 1/1/1
device(config-lag-100)# enable ethernet 1/1/2
```

Displaying BGP EVPN Multihoming Information

You can use various **show** commands to view information about BGP EVPN multihoming.

Use one of the following commands to view BGP EVPN Multihoming information. Using these commands is optional and they can be entered in any order. Commands that are supported for BGP EVPN singlehoming are also supported for BGP EVPN multihoming. For more information on these commands, refer to Displaying BGP EVPN Information on page 383. For more detailed information on any commands in the examples below, refer to the *RUCKUS FastIron Command Reference Guide*.

1. Enter the **show ip bgp evpn es** command to display summarized information about all local and remote Ethernet Segments (ESs).

```
device> show ip bgp evpn es

Id ESI                           L/R State Mode #Member #EVI #MAC
=================================================================
1  01:00:12:00:12:00:12:4e:52:00 R   new   AA   2       3    1
2  01:00:12:00:12:00:12:4e:53:00 R   new   AA   1       4    2
```

2. Enter the **show ip bgp evpn es es-id** command to display configuration information for a specified ES.

```
device> show ip bgp evpn es es-id 01:00:12:00:12:00:12:4e:52:00

Id ESI                           L/R State Mode #Member #EVI #MAC
=================================================================
1  01:00:12:00:12:00:12:4e:52:00 R   new   AA   2       3    1

Last Change: 6d3h ago
ES Members: 2.2.2.12(AA) 2.2.2.13(AA)

EVI Mapping:
VNI       Originator       Properties
---       ----------       ----------
10300     2.2.2.12         ead-es ead-evi
10300     2.2.2.13
10301     2.2.2.12         ead-es ead-evi
10301     2.2.2.13         ead-es ead-evi
10302     2.2.2.12         ead-es ead-evi
10302     2.2.2.13         ead-es ead-evi
```

3. Enter the **show ip bgp evpn es detail** command to display detailed information about all local and remote ESs.

```
device> show ip bgp evpn es detail

Id ESI                           L/R State Mode #Member #EVI #MAC
=================================================================
1  01:00:12:00:12:00:12:4e:52:00 R   new   AA   2       3    1

Last Change: 6d3h ago
ES Members: 2.2.2.12(AA) 2.2.2.13(AA)

EVI Mapping:
VNI       Originator       Properties
---       ----------       ----------
10300     2.2.2.12         ead-es ead-evi
10300     2.2.2.13
10301     2.2.2.12         ead-es ead-evi
10301     2.2.2.13         ead-es ead-evi
10302     2.2.2.12         ead-es ead-evi
10302     2.2.2.13         ead-es ead-evi

Id ESI                           L/R State Mode #Member #EVI #MAC
=================================================================
2  01:00:12:00:12:00:12:4e:53:00 R   new   AA   1       4    2

Last Change: 6h12m ago
ES Members: 2.2.2.13(AA)

EVI Mapping:
VNI       Originator       Properties
---       ----------       ----------
10300     2.2.2.13         ead-es ead-evi
10303     2.2.2.13         ead-es ead-evi
10304     2.2.2.13         ead-es ead-evi
10305     2.2.2.13         ead-es ead-evi
```

4. Enter the **show ip bgp evpn attribute-entries** command to display autonomous system (AS) path attribute entries in a multi-homing environment, including Ethernet Segment Identifier (ESI) Label Extended Community and ES Import Route Target information.

```
device> show ip bgp evpn attribute-entries

Next Hop  : 2.2.2.35           MED      :0              Origin:IGP
Originator:0.0.0.0           Cluster List:None
Aggregator:AS Number :0           Router-ID:0.0.0.0          Atomic:None
Local Pref:100              Communities:Internet
Extended Community: Bgp-Encap: Tunnel-Type: 8 ESI-Label: Redundancy mode ESI_ALL_ACTIVE_MODE
RT 1000:10030 RT 1000:10032
AS Path   : (length 0)
AsPathLen: 0  AsNum: 0, SegmentNum: 0, Neighboring As: 0, Source As 0
Address: 0xcb96e95a  Hash:2264 (0x1000000)
Links: 0x0, 0x0
Reference Counts: 2:0:2, Magic: 7
Next Hop  : 2.2.2.35           MED      :0              Origin:IGP
Originator:0.0.0.0           Cluster List:None
Aggregator:AS Number :0           Router-ID:0.0.0.0          Atomic:None
Local Pref:100              Communities:Internet
Extended Community: Bgp-Encap: Tunnel-Type: 8 ES-Import-RT: 0000.00aa.a012
AS Path   : (length 0)
AsPathLen: 0  AsNum: 0, SegmentNum: 0, Neighboring As: 0, Source As 0
Address: 0xcb96e9c5  Hash:4451 (0x1000000)
Links: 0x0, 0x0
```

5. Enter the **show ip bgp evpn routes** command to display statistics for the routes in the BGP route table for the L2VPN EVPN address family, including information for multi-homing routes.

```
device> show ip bgp evpn routes
Total number of BGP EVPN Routes: - (R: x, L: y)
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
       S:SUPPRESSED F:FILTERED s:STALE
       Prefix              Next Hop        MED        LocPrf     Weight Status
Ethernet Segment Id: 03:00:00:00:00:00:03:01:01:01
1     [4]:[03:00:00:00:00:00:03:01:01:01]:[32]:[2.2.2.35]/120
                          2.2.2.35        0          100        0      BL
         AS_PATH: Not Applicable
2     [4]:[03:00:00:00:00:00:03:01:01:01]:[32]:[2.2.2.38]/120
                          2.2.2.38        0          100        0      BI
         AS_PATH: Not Applicable
Route Distinguisher: 2.2.2.35:30
< ... IMET and MAC route outputs are removed ...>
5      [1]:[03:00:00:00:00:00:03:01:01:01]:[0]/112
                          2.2.2.35        0          100        0      BL
         AS_PATH: Not Applicable
6      [1]:[03:00:00:00:00:00:03:01:01:01]:[0]/112
                          2.2.2.38        0          100        0      BI
         AS_PATH: Not Applicable
7      [1]:[03:00:00:00:00:00:03:01:01:01]:[0xFFFFFFFF]/112
                          2.2.2.35        0          100        0      BL
         AS_PATH: Not Applicable
8      [1]:[03:00:00:00:00:00:03:01:01:01]:[0xFFFFFFFF]/112
                          2.2.2.38        0          100        0      BI
         AS_PATH: Not Applicable
Route Distinguisher: 2.2.2.35:32
< ... IMET and MAC route outputs are removed ...>
9      [1]:[03:00:00:00:00:00:03:01:01:01]:[0]/112
                          2.2.2.35        0          100        0      BL
         AS_PATH: Not Applicable
10     [1]:[03:00:00:00:00:00:03:01:01:01]:[0]/112
                          2.2.2.38        0          100        0      BI
         AS_PATH: Not Applicable
11     [1]:[03:00:00:00:00:00:03:01:01:01]:[0xFFFFFFFF]/112
                          2.2.2.35        0          100        0      BL
         AS_PATH: Not Applicable
12     [1]:[03:00:00:00:00:00:03:01:01:01]:[0xFFFFFFFF]/112
                          2.2.2.38        0          100        0      BI
         AS_PATH: Not Applicable
```

6. Enter the **show l2vpn evpn es all** command to display information for all ES instances.

```
device> show l2vpn evpn es all

Total number of Ethernet-Segment Instances: 1
Index   State  Eth-Seg-Id                      Type   Interface  Redundancy-Mode
2       UP     18c7a153da85620100               1     lg100      All-Active
```

7. Enter the **show l2vpn evpn es id 2** command to display information for a specified ESI.

```
device> show l2vpn evpn es id 2

Eth-Segment Index           : 2
Eth-Segment Identifier      : 18c7a153da85620100
Eth-Segment Type            : 1
PE Lacp Mac-address         : 1234.1234.1234
PE Port-Key                 : 1234
CE Lacp Mac-address         : 8c7a.153d.a856
CE Port-Key                 : 20100
System Mac-address          : NA
Local-discriminator         : 0
Access-Interface            : lg100 (dynamic)
Redundancy-mode             : All-Active
Ethernet-Segment state      : Up
Eth-Segment Configuration   : Complete
EVI mapping                 : 100
```

Protected Port

Protected Port Overview

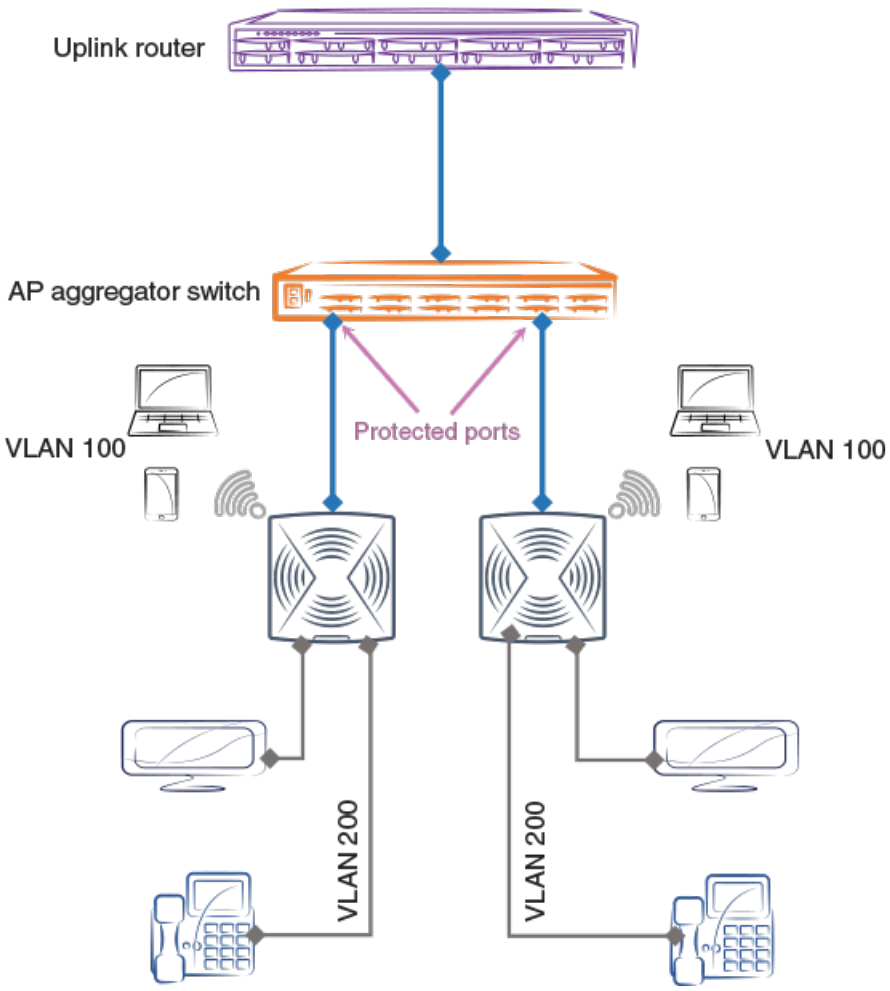
Protected ports restrict all but CPU–bound or –originated traffic, providing isolation to end hosts.

The protected port feature has wide applicability to access point (AP) aggregator switches used for hospitality, public Wi-Fi, campuses, and condominiums.

Protected port is a port-level, per-device/stack only, security feature that restricts communication with a device connected to the port. As a result, even ports that are in identical broadcast domains, once protected, will not communicate with other protected ports, irrespective of their VLAN membership, and instead access the uplink alone. This provides isolation among hosts connected to the ports by restricting all traffic between those hosts.

The following figure illustrates the use of this feature in, for example, the hospitality sector.

FIGURE 116 Protected Port Application



The following configurations are supported with the protected port feature:

- Port MAC security
- 802.1x security
- DHCP snooping
- Control protocols
- Aggregated ports (LAGs)

The following should not be configured as protected ports:

- Uplink ports
- DHCP server ports
- ARP inspection trusted ports
- DHCP snooping trusted ports
- Ports on an active xSTP path in a device
- IGMP/MLD snooping router ports
- IGMP/MLD source ports

Ruckus recommends that multiple interface (MIF) mode is configured when enabling this feature.

The following features are not supported on protected ports:

- Layer 3 interfaces (Port or LAGs with IP addresses are not supported)
- Mirror or monitor ports
- Private VLAN (PVLAN)
- PVLAN extension to protected-port switches
- Virtual Ethernet (VE) and group VE interfaces
- Loopback interfaces
- Management interfaces
- OpenFlow ports
- Multi-Chassis Trunk (MCT) ICL and CCEP ports

Configuring Protected Port

This task configures the protected port feature on a single and multiple interfaces in interface or multiple interface (MIF) mode.

Use the **no** form of the **protected-port** command to disable the feature.

1. Enter global configuration mode.

```
device# configure terminal
```

2. To configure this feature on a single interface, specify an interface. and enter the **protected-port** command.

```
device(config)# interface ethernet 1/1/1
```

3. Enter the **protected-port** command.

```
device(config-if-e1000-1/1/1)# protected-port
```

4. Enter the **show interface ethernet** command to confirm the interface configuration.

```
device# show interface ethernet 1/1/1

GigabitEthernet1/1/1 is down, line protocol is down
  Port down for 3 minute(s) 59 second(s)
  Hardware is GigabitEthernet, address is 748e.f882.f480 (bia 748e.f882.f480)
  Configured speed auto, actual unknown, configured duplex fdx, actual unknown
  Configured mdi mode AUTO, actual unknown
  Member of L2 VLAN ID 1, port is untagged, port state is BLOCKING
 <---output omitted--->
  0 packets output, 0 bytes, 0 underruns
  Transmitted 0 broadcasts, 0 multicasts, 0 unicasts
  0 output errors, 0 collisions
  Relay Agent Information option: Disabled
  Protected: Yes
```

5. Use the **show protected-ports** command to confirm the system-wide configuration.

```
device# show protected-ports
  System-Wide Protected Ports: ethe 1/1/1 ethe 2/1/1 ethe 3/1/1 lag lg1
```

The following example enables this feature for multiple interfaces in MIF mode (recommended).

```
device(config)# interface ethernet 2/1/1 ethernet 3/1/1
device(config-if-e1000-2/1/1,3/1/1)# protected-port
```

The following example uses a range of interfaces before enabling this feature for multiple interfaces.

```
device(config)# interface ethernet 2/1/1 to ethernet 2/1/48
device(config-if-e1000-2/1/1:2/1/48)# protected-port
```

The following example enables this feature on LAG virtual interface.

```
device(config)# interface lag 1
device(config-lag-if-lg1)# protected-port
```

350 West Java Dr., Sunnyvale, CA 94089 USA
https://www.commscope.com