CONFIGURATION GUIDE

Ruckus FastIron Layer 2 Switching Configuration Guide, 08.0.61

Supporting FastIron Software Release 08.0.61

Part Number: 53-1005213-04
Publication Date: 25 January 2019

Copyright, Trademark and Proprietary Rights Information

© 2019 ARRIS Enterprises LLC. All rights reserved.

No part of this content may be reproduced in any form or by any means or used to make any derivative work (such as translation, transformation, or adaptation) without written permission from ARRIS International plc and/or its affiliates ("ARRIS"). ARRIS reserves the right to revise or change this content from time to time without obligation on the part of ARRIS to provide notification of such revision or change.

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, ARRIS 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. ARRIS 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. ARRIS 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 ARRIS 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 ARRIS, ARRIS 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 ARRIS 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

ARRIS, the ARRIS logo, Ruckus, Ruckus Wireless, Ruckus Networks, Ruckus logo, the Big Dog design, BeamFlex, ChannelFly, EdgeIron, FastIron, HyperEdge, ICX, IronPoint, OPENG, SmartCell, Unleashed, Xclaim, ZoneFlex are trademarks of ARRIS International plc and/or its affiliates. Wi-Fi Alliance, Wi-Fi, the Wi-Fi logo, the Wi-Fi CERTIFIED logo, Wi-Fi Protected Access (WPA), the Wi-Fi Protected Setup logo, and WMM are registered trademarks of Wi-Fi Alliance. Wi-Fi Protected Setup™, Wi-Fi Multimedia™, and WPA2™ are trademarks of Wi-Fi Alliance. All other trademarks are the property of their respective owners.

Contents

Preface **11**
- Document Conventions 11
  - Notes, Cautions, and Warnings 11
- Command Syntax Conventions 12
- Document Feedback 12
- Ruckus Product Documentation Resources 12
- Online Training Resources 13
- Contacting Ruckus Customer Services and Support 13
  - What Support Do I Need? 13
  - Open a Case 13
  - Self-Service Resources 13

About This Document **15**
- What's new in this document 15
- Supported Hardware 15
- How Command Information is Presented in this Configuration Guide 15

Remote Fault Notification **17**
- Remote Fault Notification on 1Gbps fiber connections 17
- Enabling and disabling remote fault notification 17

Metro Ring Protocol **19**
- Metro Ring Protocol Overview 19
  - Metro Ring Protocol configuration notes 21
- MRP rings with shared interfaces (MRP Phase 2) 21
  - Selection of master node 22
- MRP rings without shared interfaces (MRP Phase 1) 23
- Ring initialization 24
  - RHP processing in MRP Phase 1 26
  - RHP processing in MRP Phase 2 28
- How ring breaks are detected and healed 29
- Master VLANs and customer VLANs 31
- Metro Ring Protocol configuration 32
  - Adding an MRP ring to a VLAN 32
  - Changing the hello and preforwarding times 34
- Metro Ring Protocol diagnostics 34
  - Enabling MRP diagnostics 34
  - Displaying MRP diagnostics 34
- Displaying MRP information 35
- MRP CLI example 37
  - MRP commands on Switch A (master node) 38
  - MRP commands on Switch B 38
  - MRP commands on Switch C 39
  - MRP commands on Switch D 39

Virtual Switch Redundancy Protocol (VSRP) **41**
- VSRP overview 41
- VSRP configuration notes and feature limitations 43

VSRP redundancy ... 43
Master election and failover ... 43
VSRP failover ... 43
VSRP priority calculation ... 44
MAC address failover on VSRP-aware devices ... 47
VSRP interval timers ... 48
Configuring device redundancy using VSRP ... 48
Configuring optional VSRP parameters ... 50
Configuring authentication on VSRP interfaces ... 51
Tracking ports and setting the VSRP priority ... 51
Disabling backup pre-emption setting ... 52
Disabling VSRP backup preemption ... 52
VSRP-aware security features ... 53
Configuring security parameters on a VSRP-aware device ... 53
VSRP fast start ... 54
Special considerations when configuring VSRP fast start ... 54
Recommendations for configuring VSRP fast start ... 54
Configuring VSRP fast start globally ... 55
VSRP and MRP signaling ... 55

UDLD and Protected Link Groups ... 59
UDLD overview ... 59
UDLD for tagged ports ... 60
Configuration notes and feature limitations for UDLD ... 60
Enabling UDLD ... 60
Enabling UDLD for tagged ports ... 61
Changing the Keepalive interval ... 61
Changing the keepalive retries ... 61
Displaying information for all ports ... 61
Displaying information for a single port ... 62
Clearing UDLD statistics ... 63

Link Aggregation Group ... 65
Overview of link aggregation ... 65
LAG virtual interface ... 66
LAG virtual interface MAC ... 66
LAG formation rules ... 66
Configuration notes for FastIron devices in a traditional stack ... 68
Maximum number of LAGs ... 69
Upgrade and downgrade notes ... 70
LAG Load Sharing ... 70
LAG hashing on stacking products ... 71
Symmetric load balancing ... 71
Use case: Deploying Ruckus ICX 7750 as a traffic splitter in a DPI solution ... 73
Resilient hashing ... 74
Resilient hashing limitations ... 75
Configuring resilient hashing ... 75
Configuring a LAG ... 75
Creating a Link Aggregation Group (LAG) ... 76
Configuring LAG virtual interface ... 77
Configuring an LACP timeout ... 78

Specifying the LAG threshold for a LAG group ... 78
Disabling ports within a LAG ... 79
Enabling ports within a LAG ... 79
Deleting a port from a currently operational LAG ... 79
Monitoring LAG virtual interface and individual LAG port ... 79
Assigning a name to a port within a LAG ... 80
Enabling sFlow forwarding on a port in a LAG ... 81
Setting the sFlow sampling rate for a LAG ... 82
IP assignment within a LAG ... 82
Renaming an existing LAG ... 82
Displaying LAG information ... 82
Enabling LAG hardware failover ... 84
Preboot eXecution Environment boot support ... 85
User-configured peer information per LACP ... 85

Multi-Chassis Trunking ... 87
Multi-Chassis Trunking Overview ... 87
How MCT works ... 87
MCT terminology ... 88
MCT data flow ... 89
MCT and VLANs ... 92
MCT Feature Interaction and Unsupported Features ... 92
Basic MCT configuration ... 93
MCT configuration considerations ... 94
Differences in configuring MCT for the switch and router image ... 95
Configuring MCT ... 95
Forcing a port up in a basic MCT configuration ... 98
Cluster client automatic configuration ... 99
Setting up cluster client automatic configuration ... 100
MCT failover scenarios ... 101
Cluster failover mode ... 102
Client isolation mode ... 102
Shutting down all client interfaces ... 103
Using the keep-alive VLAN ... 103
Setting keep-alive timers and hold-time ... 103
Layer 2 behavior with MCT ... 104
MAC operations ... 104
MAC Database Update ... 104
Cluster MAC types ... 104
MAC aging ... 105
MAC flush ... 105
Syncing router MAC addresses to peer MCT devices ... 105
Dynamic trunks ... 105
Port loop detection ... 106
MCT Layer 2 protocols ... 106
Layer 2 multicast snooping over MCT ... 107
Layer 3 behavior with MCT ... 113
Layer 3 unicast forwarding over MCT ... 114
VRRP or VRRP-E over an MCT-enabled network ... 115
OSPF and BGP over an MCT-enabled network ... 116
Layer 3 with MCT configuration considerations ... 117

MCT configuration for a single-level MCT deployment ... 118
MCT configuration with VRRP-E ... 120
MCT Configuration with OSPF ... 121
MCT Configuration with BGP ... 122
PIM over MCT intermediate router functionality ... 123
Displaying MCT information ... 129
Displaying peer and client states ... 129
Displaying state machine information ... 130
Displaying cluster, peer, and client states ... 130
Displaying information about Ethernet interfaces ... 131
Displaying STP information ... 132
MAC show commands ... 132
Displaying MDUP packet statistics ... 133
Single-level MCT configuration example ... 134
Client 1 - Configuration ... 135
Client 2- Configuration ... 136
AGG-A (R1) - Configuration ... 136
AGG-B (R2) - Configuration ... 137
Two-level MCT configuration example ... 137
AGG-A (R1) - Configuration ... 139
AGG-B (R2) - Configuration ... 140
DIST-A (R3) - Configuration ... 140
DIST-B (R4) - Configuration ... 141
MCT configuration examples using STP ... 142
Router-1 configuration ... 143
AGG-B (R2) - Configuration ... 144
Client-1 - Configuration ... 144
Client-2 - Configuration ... 145
Example 1: Configure the Per-VLAN Spanning Tree on the MCT Clients ... 145
Example 2: Configure Single Spanning Tree (SSTP) on the MCT Clients ... 145
Example 3: Configure Multiple Spanning Tree (MSTP) on the MCT Clients ... 146

GVRP ... 149
GVRP overview ... 149
GVRP application examples ... 149
Dynamic core and fixed edge ... 150
Dynamic core and dynamic edge ... 151
Fixed core and dynamic edge ... 152
Fixed core and fixed edge ... 152
VLAN names created by GVRP ... 152
Configuration notes for GVRP ... 152
Configuring GVRP ... 153
Clearing GVRP statistics ... 155
Configuration example: Implementing the applications of GVRP ... 156
Dynamic core and fixed edge ... 157
Dynamic core and dynamic edge ... 158
Fixed core and dynamic edge ... 158
Fixed core and fixed edge ... 158

Spanning Tree Protocol ... 159
STP overview ... 159

Standard STP parameter configuration....159
STP parameters and defaults....159
Enabling or disabling the Spanning Tree Protocol (STP)....161
Changing STP bridge and port parameters....162
STP protection enhancement....164
Displaying STP information....165
STP feature configuration....168
Fast port span....168
Fast Uplink Span....170
802.1W Rapid Spanning Tree (RSTP)....172
802.1W Draft 3....210
Single Spanning Tree (SSTP)....214
STP per VLAN group....216
Spanning Tree Path Cost Support for 802.1D 2004....219
PVST/PVST+ compatibility....221
Overview of PVST and PVST+....221
VLAN tags and dual mode....222
Configuring PVST+ support....223
Displaying PVST+ support information....223
PVST+ configuration examples....224
PVST+ Protect....226
PVRST compatibility....230
BPDU guard....230
Enabling BPDU protection by port....230
Re-enabling ports disabled by BPDU guard....231
Displaying the BPDU guard status....231
BPDU guard status example console messages....232
Root guard....232
Enabling STP root guard....233
Displaying the STP root guard....233
Displaying the root guard by VLAN....233
Designated Protection....233
Enabling Designated Protection on a port....234
Syslog message for a port in designated inconsistent state....234
Packet InError Detection....234
Configuring Packet InError Detection....235
Syslog message for error-disabled port due to inError packets....235
Error disable recovery....235
Enabling an error-disabled port automatically....236
Enabling an error-disabled port manually....236
Setting the recovery interval....236
Displaying the error disable recovery state by interface....236
Displaying the recovery state for all conditions....237
Displaying the recovery state by port number and cause....237
Errdisable Syslog messages....237
802.1s Multiple Spanning Tree Protocol....237
Multiple spanning-tree regions....237
Configuration notes....239
Configuring MSTP mode and scope....239
Reduced occurrences of MSTP reconvergence....240

Configuring additional MSTP parameters....241
MSTP root guard....249

VLANs....253
VLAN overview....253
VLAN support on FastIron devices....253
Layer 2 port-based VLANs....253
Configuring port-based VLANs on Device-A....257
Configuring port-based VLANs on Device-B....258
Configuring port-based VLANs on Device-C....258
Modifying a port-based VLAN....258
Default VLAN....265
802.1Q tagging....267
Spanning Tree Protocol (STP)....269
Virtual routing interfaces....270
VLAN and virtual routing interface groups....270
Super aggregated VLANs....271
Trunk group ports and VLAN membership....271
Summary of VLAN configuration rules....271
Routing between VLANs....272
Virtual routing interfaces (Layer 2 Switches only)....272
Routing between VLANs using virtual routing interfaces (Layer 3 Switches only)....272
Dynamic port assignment (Layer 2 Switches and Layer 3 Switches)....273
Assigning a different VLAN ID to the default VLAN....273
Assigning different VLAN IDs to reserved VLANs 4091 and 4092....273
Assigning trunk group ports....274
Enable spanning tree on a VLAN....274
Enabling port-based VLANs....276
Assigning IEEE 802.1Q tagging to a port....276
VLAN-based static MAC entries configuration....277
Configuring a VLAN to drop static MAC entries....277
Routing between VLANs using virtual routing interfaces (Layer 3 Switches only)....277
Configuring Layer 3 VLANs and virtual routing interfaces on the Device-A....278
Configuring Layer 3 VLANs and virtual routing interfaces for Device-B....280
Configuring Layer 3 VLANs and virtual routing interfaces for Device-C....281
IP subnet address on multiple port-based VLAN configuration....282
VLAN groups and virtual routing interface group....285
Configuring a VLAN group....286
Configuring a virtual routing interface group....287
Displaying the VLAN group and virtual routing interface group information....288
Allocating memory for more VLANs, more associated ports, or more virtual routing interfaces....288
Topology groups....290
Master VLAN and member VLANs....290
Control ports and free ports....290
Topology group configuration considerations....291
Configuring a topology group....291
Displaying STP information....292
Displaying topology group information....292
Super-aggregated VLAN configuration....293
Configuration notes for aggregated VLANs....296
Configuring aggregated VLANs....296

Verifying the aggregated VLAN configuration ... 298
Complete CLI examples for aggregated VLANs ... 298
802.1ad tagging configuration ... 300
Configuration rules for 802.1ad tagging ... 301
Enabling 802.1ad tagging ... 302
Example 802.1ad configuration ... 302
Configuring 802.1ad tag profiles ... 303
Dual-mode VLAN ports ... 304
Private VLAN configuration ... 307
Multiple tagged and untagged support for PVLANs ... 311
Configuration notes for PVLANs and standard VLANs ... 311
CLI example for a general PVLAN network ... 315
Configuration example for dual-mode PVLAN network ... 315
Multiple promiscuous ports support in private VLANs ... 316
Displaying VLAN information ... 316
Displaying VLANs in alphanumeric order ... 317
Displaying system-wide VLAN information ... 317
Displaying global VLAN information ... 318
Displaying VLAN information for specific ports ... 318
Displaying a port VLAN membership ... 318
Displaying a port dual-mode VLAN membership ... 319
Displaying port default VLAN IDs (PVIDs) ... 319
Displaying PVLAN information ... 320
Protected Port ... 321
Protected port ... 321
Configuring protected port ... 323

Preface

- Document Conventions....11
- Command Syntax Conventions....12
- Document Feedback....12
- Ruckus Product Documentation Resources....12
- Online Training Resources....13
- Contacting Ruckus Customer Services and Support....13

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 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.

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@arris.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.ruckuswireless.com.

Online Training Resources

To access a variety of online Ruckus training modules, including free introductory courses to wireless networking essentials, site surveys, and Ruckus products, visit the Ruckus Training Portal at https://training.ruckuswireless.com.

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 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.ruckuswireless.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 **Open 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 **Open a Case** section.
- Priority 3 (P3)—Medium. Network or service is moderately impacted, but most business remains functional. Go to the **Self-Service Resources** section.
- Priority 4 (P4)—Low. Requests for information, product documentation, or product enhancements. Go to the **Self-Service Resources** section.

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://forums.ruckuswireless.com/ruckuswireless/categories
- 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 TAC with additional data from your troubleshooting analysis if you still require assistance through a support case or RMA. If you still require help, open and manage your case at https://support.ruckuswireless.com/case_management.

About This Document

- What's new in this document....15
- Supported Hardware....15
- How Command Information is Presented in this Configuration Guide....15

What's new in this document

The following table describes the changes to this guide for the FastIron 08.0.61 release.

TABLE 2 Summary of changes in FastIron release 08.0.61

Feature	Description	Location
LAG virtual interface	Enhancement to LAG with new method of LAG configuration where primary port is no longer required. LAG virtual interface is a logical interface that represents a bundle of physical interfaces that form a LAG. All features running over LAG will be anchored on the LAG virtual interface.	LAG virtual interface on page 66
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.	MSTP root guard on page 249
Protected port	Protected ports restrict communication among such ports at the system level, providing isolation to end hosts.	Protected port on page 321

Supported Hardware

This guide supports the following product families from Ruckus:

- ICX 7750 Series
- ICX 7450 Series
- ICX 7250 Series
- ICX 7150 Series

For information about the specific models and modules supported in a product family, refer to the hardware installation guide for that product family.

How Command Information is Presented in this Configuration Guide

For all new content supported in FastIron release 08.0.20 and later, command information is documented in a standalone command reference guide.

In the *Ruckus FastIron Command Reference*, the command pages are in alphabetical order and follow a standard format to present syntax, parameters, mode, usage guidelines, examples, and command history.

NOTE
Many commands introduced before FastIron release 08.0.20 are also included in the guide.

Remote Fault Notification

- Remote Fault Notification on 1Gbps fiber connections..... 17
- Enabling and disabling remote fault notification..... 17

Remote Fault Notification on 1Gbps fiber connections

NOTE
Remote Fault Notification (RFN) is only available for 1 Gbps Ethernet Fiber ports. It is not available for 10G/100G ports and 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.

By default, RFN is enabled.

You can configure RFN as follows:

- Globally, on the entire device
- On a trunk group
- On an individual interface

Enabling and disabling 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 set to **auto-gig** by default. To disable RFN, use the following command.

```
device(config)# interface ethernet 1/1/1
device(config-if-e1000-1/1/1)# gig-default neg-off
```

To re-enable RFN, use the following command.

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

Syntax: [no] gig-default { **neg-full-auto** | **neg-off** | **auto-gig** }

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* .

Metro Ring Protocol

- Metro Ring Protocol Overview 19
- MRP rings with shared interfaces (MRP Phase 2) 21
- MRP rings without shared interfaces (MRP Phase 1) 23
- Ring initialization 24
- How ring breaks are detected and healed 29
- Master VLANs and customer VLANs 31
- Metro Ring Protocol configuration 32
- Metro Ring Protocol diagnostics 34
- Displaying MRP information 35
- MRP CLI example 37

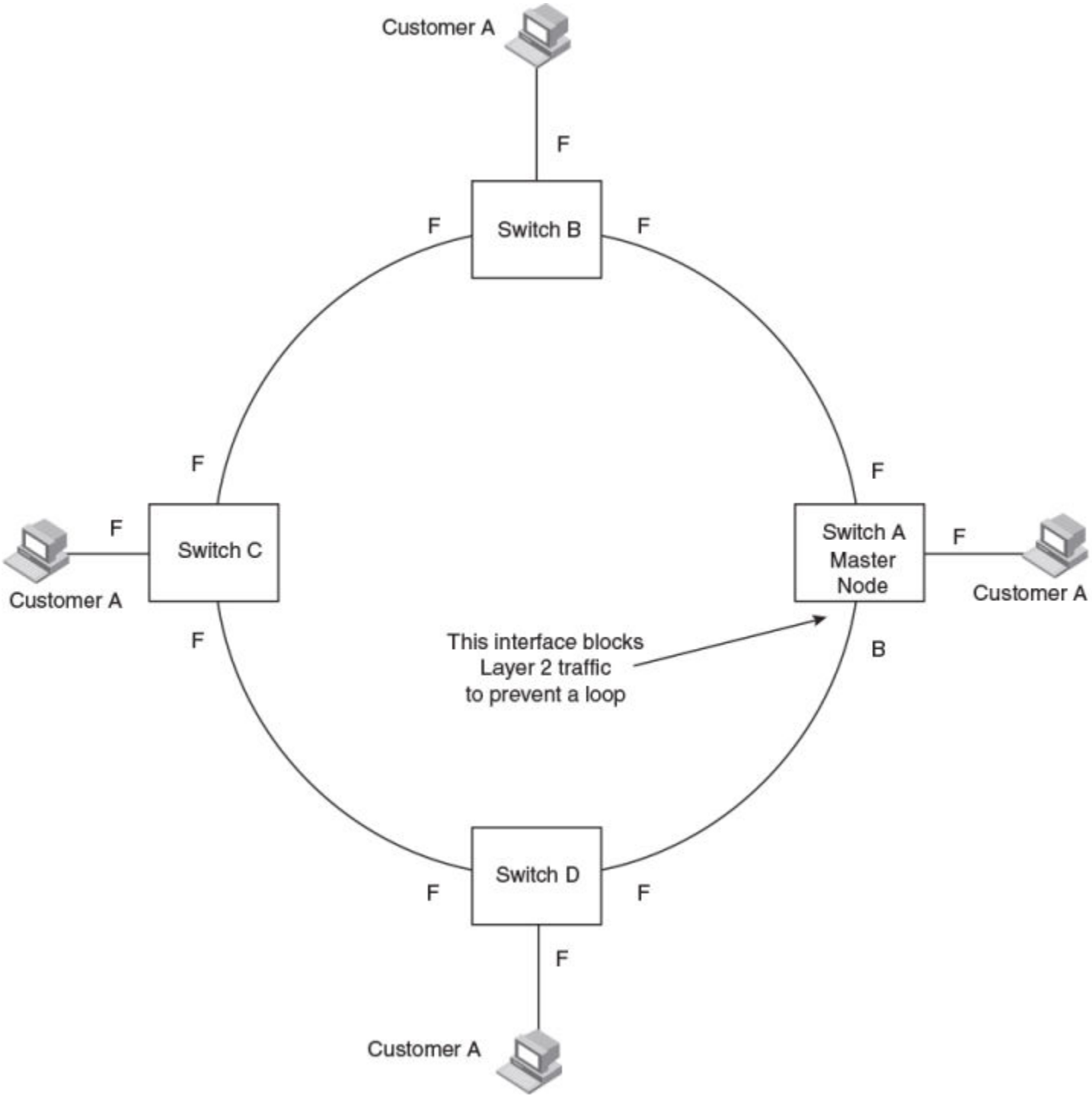
Metro Ring Protocol Overview

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 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.

The following figure shows an example of an MRP metro ring.

FIGURE 1 Metro ring - normal state



The ring in this example 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 the ring. The ring interfaces are all in one 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 MRP 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.

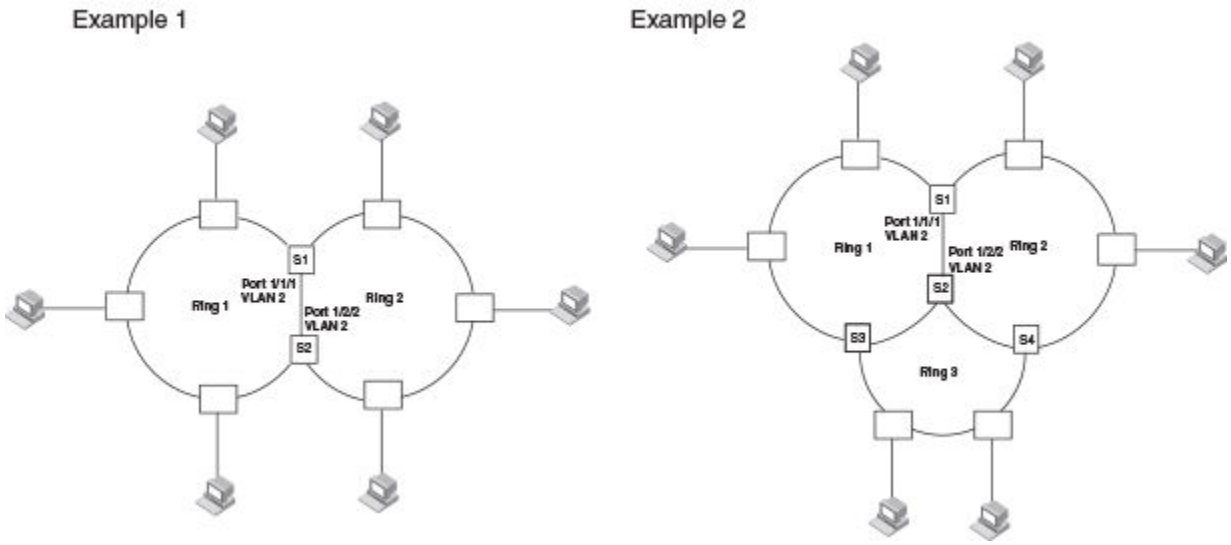
Metro Ring Protocol configuration notes

- 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 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.

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.

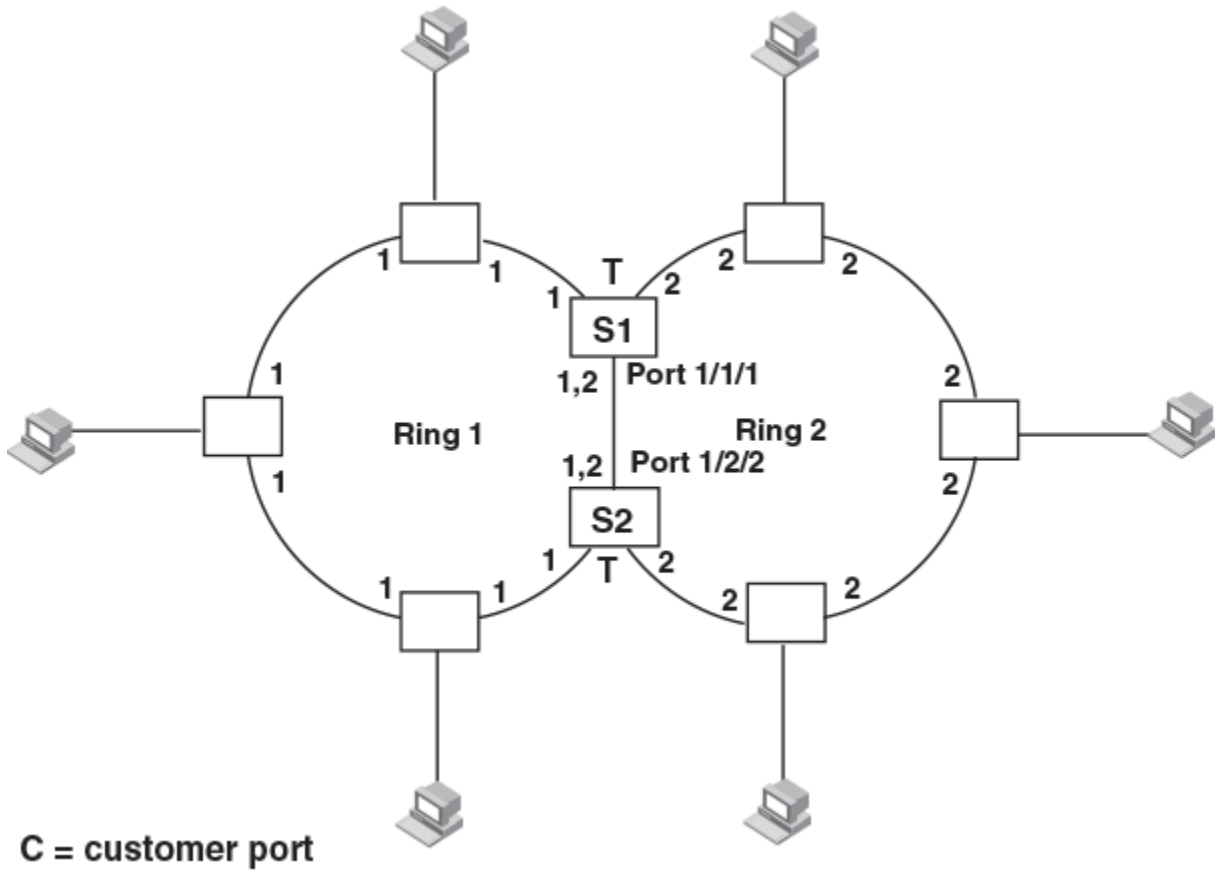
FIGURE 2 Examples of multiple rings sharing the same interface - MRP Phase 2



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 ring 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.

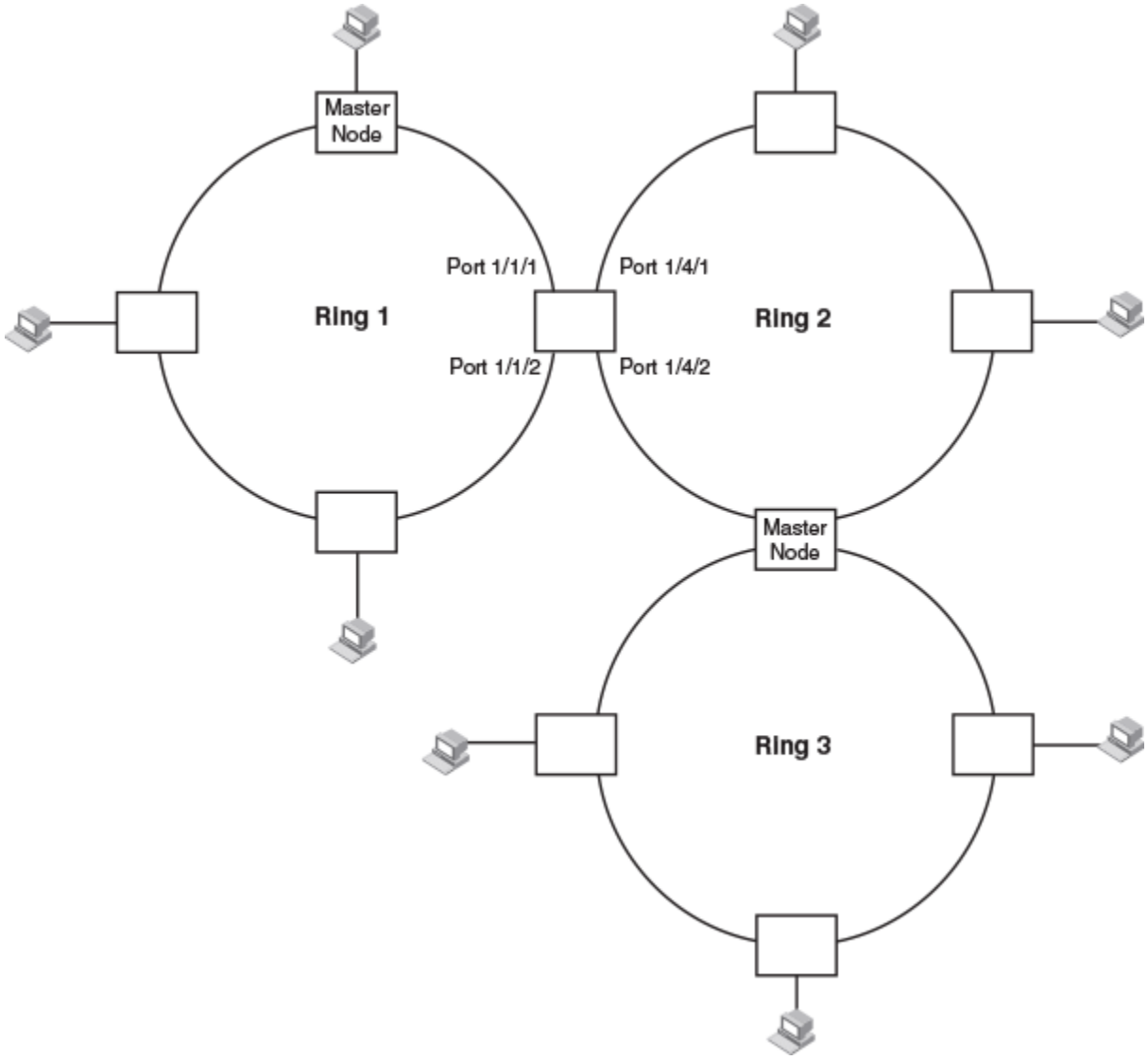
In Figure 3 on page 22, any of the nodes on Ring 1, even S1 or S2, can be a master node since none of its interfaces are tunnel ports. However in Ring 2, neither S1 nor S2 can be a master node since these nodes contain tunnel ports.

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

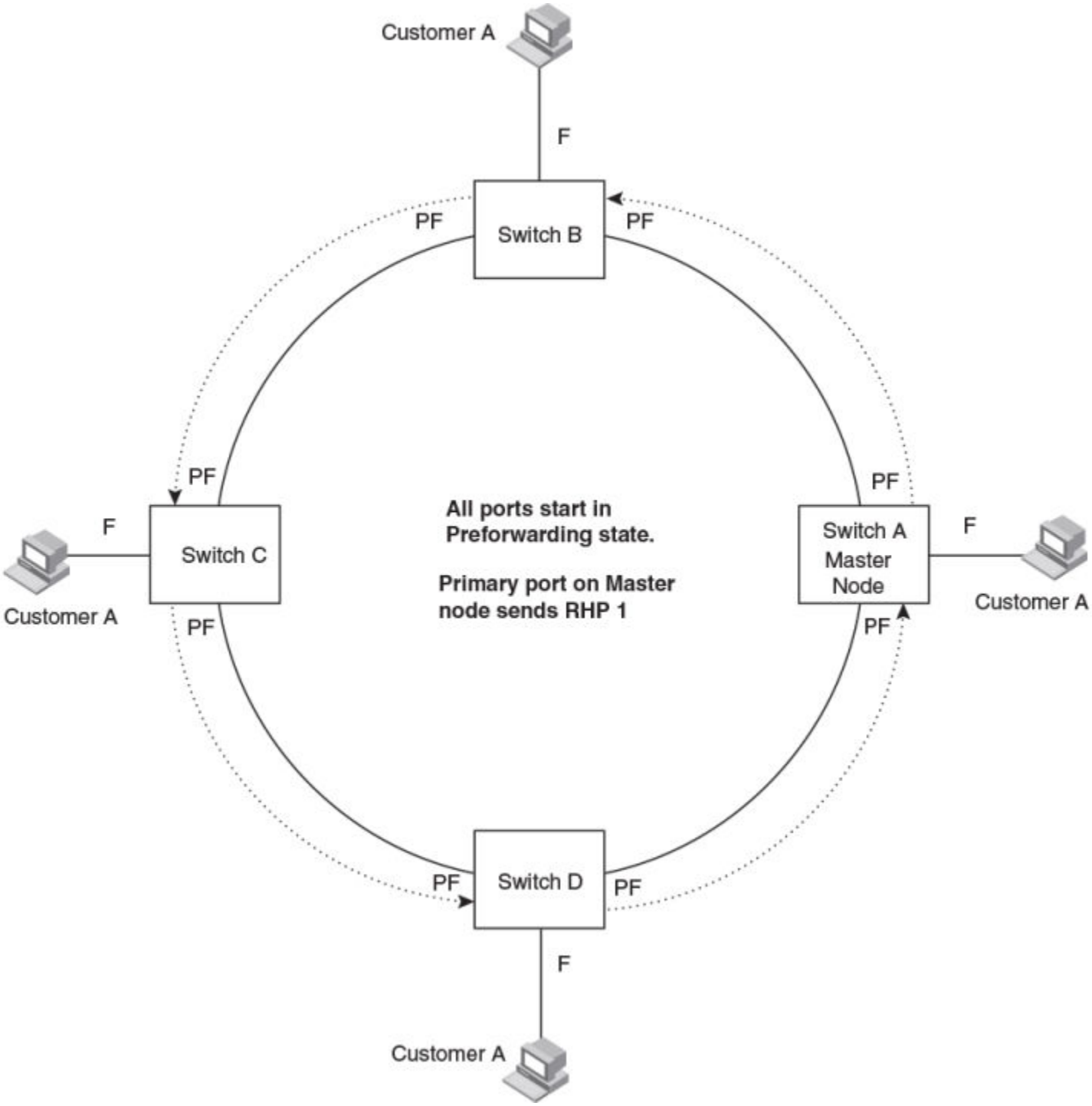


In this example, two nodes are each configured with two MRP rings. Any node in a ring can be the master for its ring. A node also can be the master for more than one ring.

Ring initialization

The ring shown in Figure 1 on page 20 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:

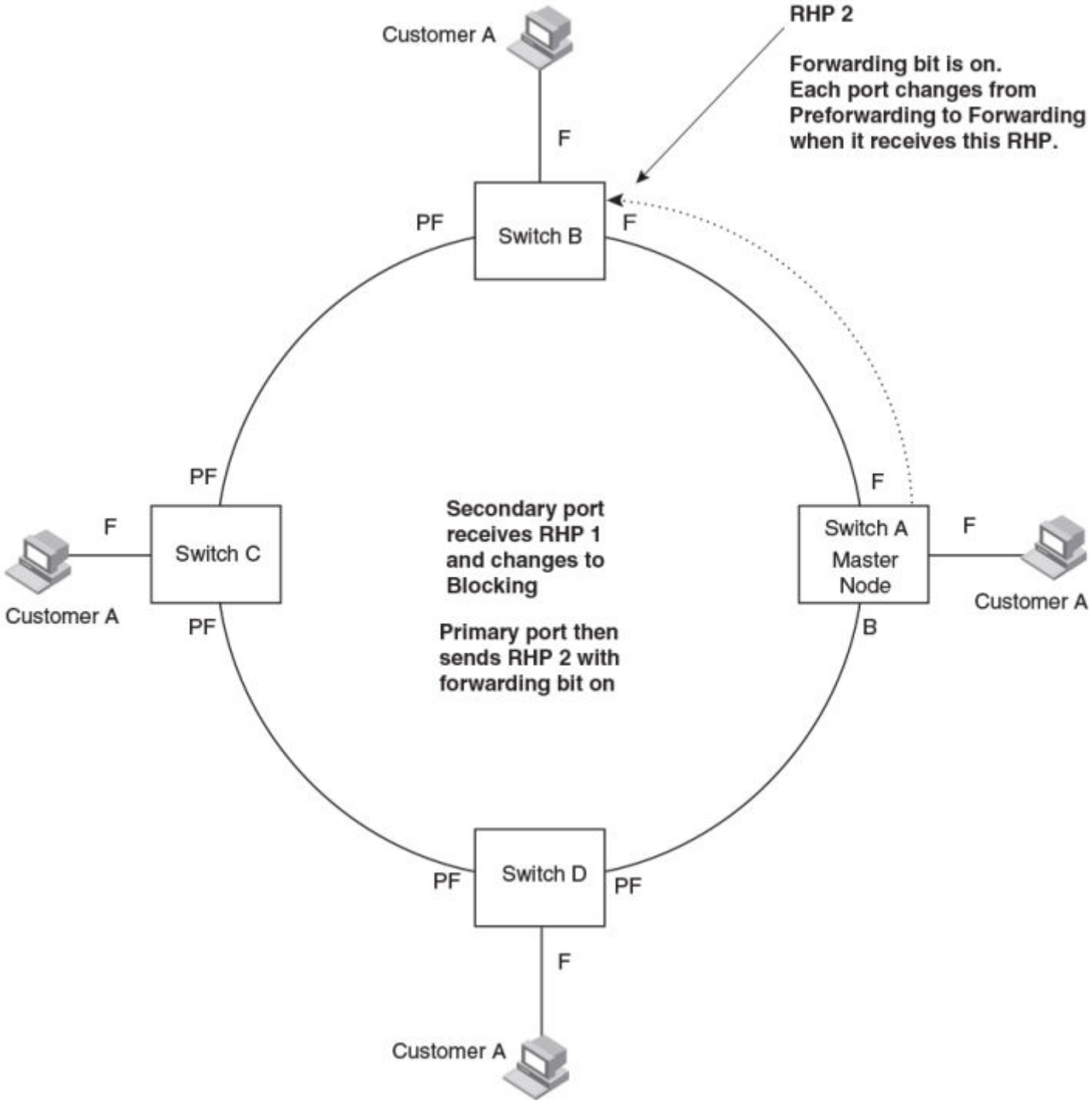
- Preforwarding (PF) - The interface can forward RHPS but cannot forward data. All ring ports begin in this state when you enable MRP.
- Forwarding (F) - The interface can forward data as well as RHPs. An interface changes from Preforwarding to Forwarding when the port preforwarding time expires. This occurs if the port does not receive an RHP from the Master, 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.

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 port 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 MRP 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.

The following figure shows an example.

FIGURE 6 Metro ring - from preforwarding to forwarding

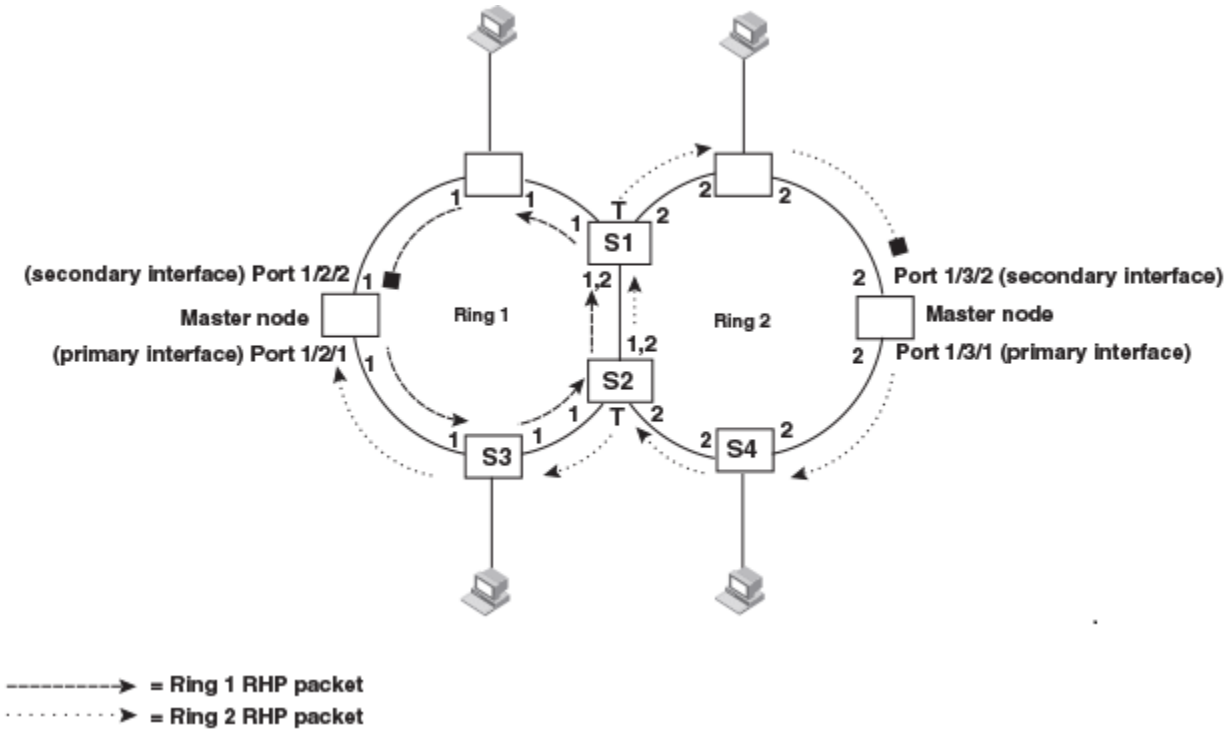


Each RHP also has a sequence number. MRP can use the sequence number to determine the round-trip time for RHPs in the ring. Refer to Metro Ring Protocol diagnostics on page 34.

RHP processing in MRP Phase 2

Figure 7 shows an example of how RHP packets are processed normally in MRP rings with shared interfaces.

FIGURE 7 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.

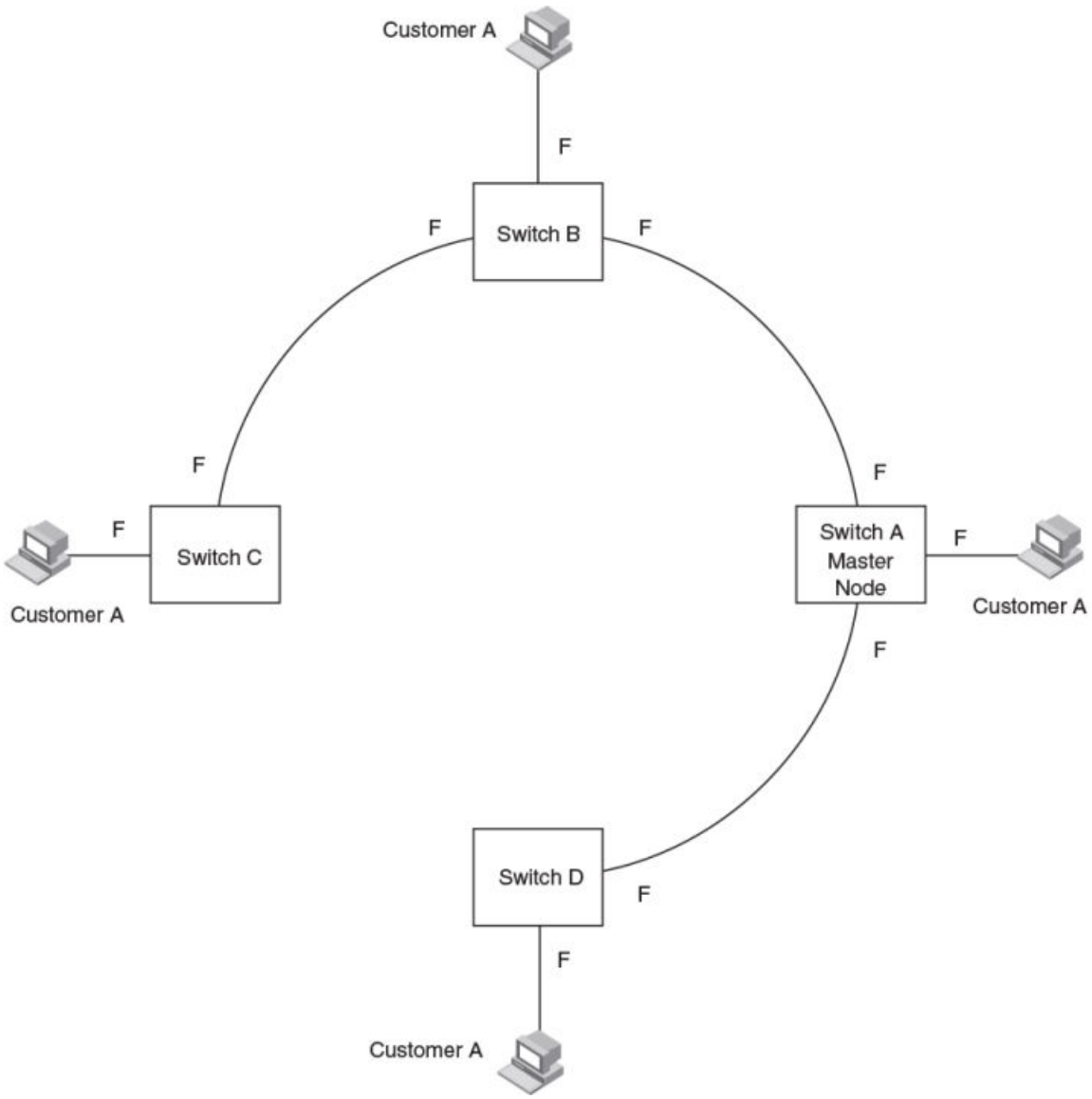
When the RHP packet reaches the interface on node S2 shared by Rings 1 and 2, the packet is forwarded since its priority is less than the interface priority. The packet continues to be forwarded to node S1 until it reaches the tunnel port on S1. That tunnel port determines that the RHP packet priority 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 packets 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 8 shows ring interface states following a link break. MRP quickly heals the ring and preserves connectivity among the customer networks.

FIGURE 8 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 8.

- **Forwarding interfaces** - Each member interface remains in the Forwarding state.

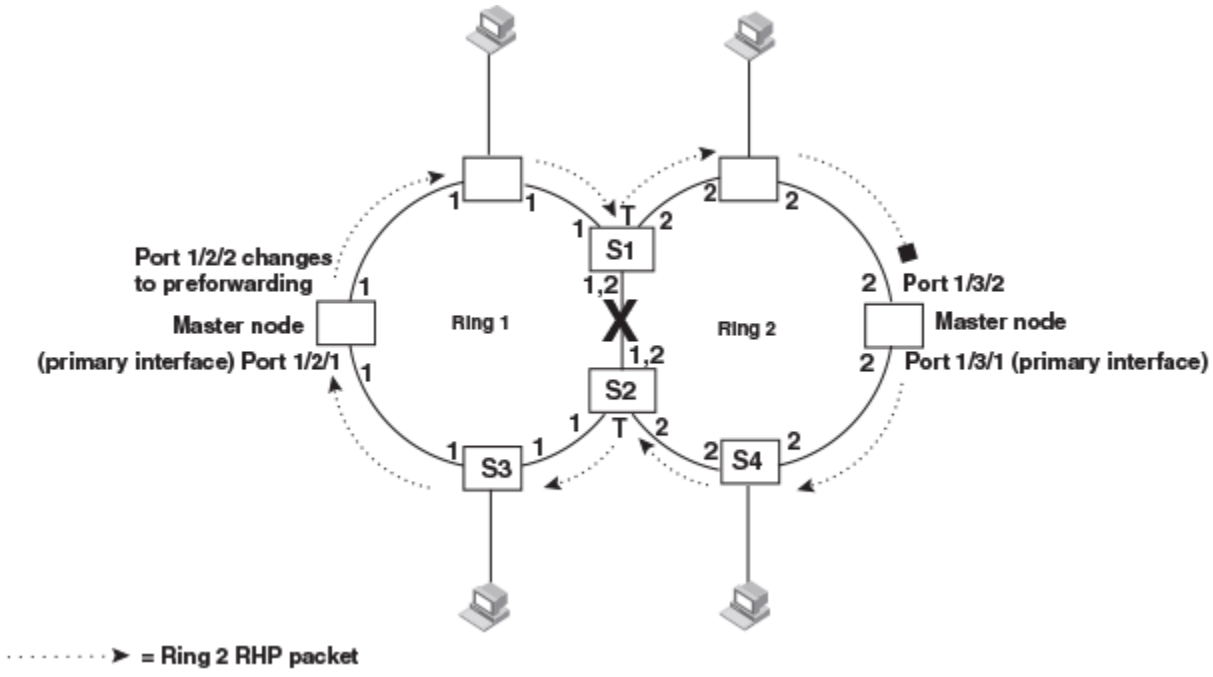
When the broken link is repaired, the link interfaces come up in the Preforwarding state, which allows RHPs to travel through the restored interfaces and reach the secondary interface on the Master node:

- 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 9), 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 mode 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 9 Flow of RHP packets when a link for shared interfaces breaks

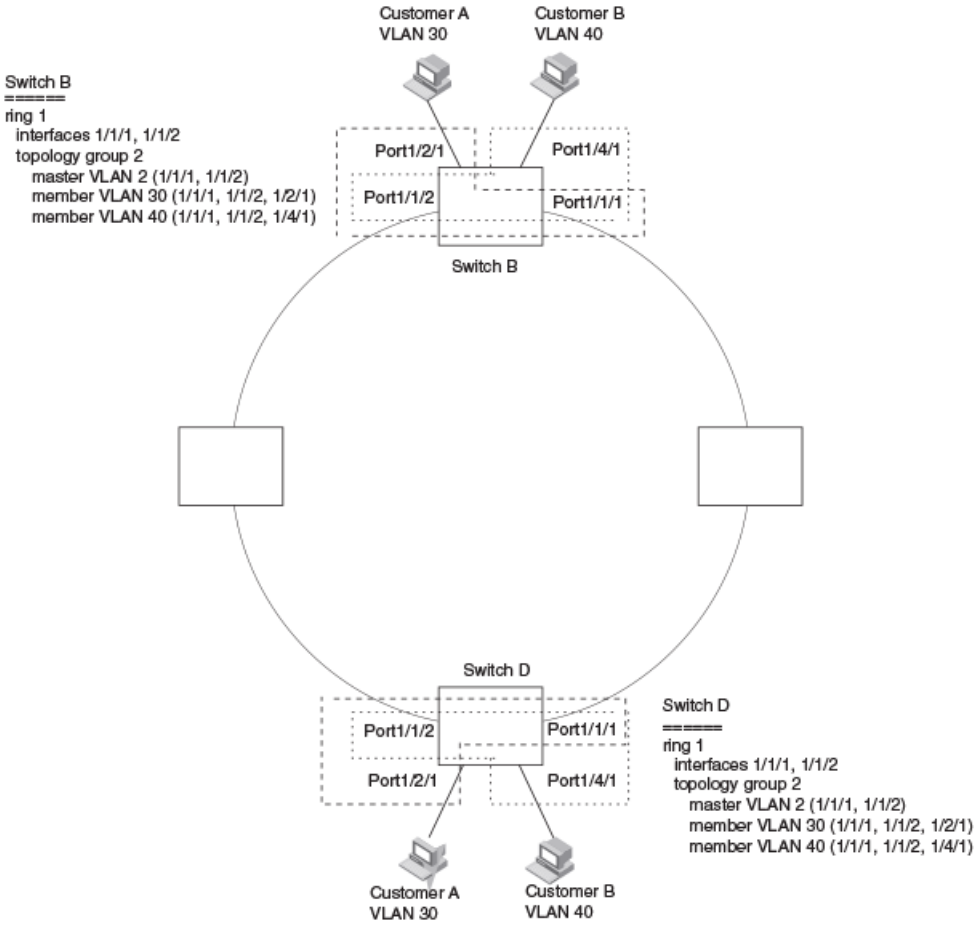


RHP packets follow this flow until the link is restored; then the RHP packet returns to it normal flow as shown in Figure 7 on page 28.

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 10 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 10, 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 290.

Refer to MRP CLI example on page 37 for the configuration commands required to implement the MRP configuration shown in Figure 10.

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.

NOTE
There are no new commands or parameters to configure MRP with shared interfaces (MRP Phase 2).

- Disable one of the ring interfaces. This prevents a Layer 2 loop from occurring while you are configuring the devices for MRP.
- 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.
  - On the master node only, 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.
  - 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.
- 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 290.
- Re-enable the interface you disabled to prevent a Layer 2 loop. Once MRP is enabled, MRP will prevent the Layer 2 loop.

Adding an MRP ring to a VLAN

To add an MRP ring to a VLAN, enter commands such as the following.

NOTE
If you plan to use a topology group to add VLANs to the ring, make sure you configure MRP on the topology group master VLAN.

```
device(config)#vlan 2
device(config-vlan-2)#metro-ring 1
device(config-vlan-2-mrp-1)#name CustomerA
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
```

These commands configure an MRP ring on VLAN 2. The ring ID is 1, the ring name is CustomerA, and this node (this Ruckus device) is the master for the ring. The ring interfaces are 1/1/1 and 1/1/2. Interface 1/1/1 is the primary interface and 1/1/2 is the secondary interface. The primary interface will initiate RHPs by default. The ring takes effect in VLAN 2.

```
device(config)#vlan 2
device(config-vlan-2)#metro-ring 1
device(config-vlan-2-mrp-1)#name CustomerA
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)#metro-ring 2
device(config-vlan-2-mrp-2)#name CustomerB
device(config-vlan-2-mrp-2)#ring-interface ethernet 1/1/1 ethernet 1/1/2
device(config-vlan-2-mrp-2)#enable
```

Syntax: [no] metro-ring ring *id*

The *ring-id* parameter specifies the ring ID. The *ring-id* can be from 1 - 1023; ID 256 is reserved for VSRP.

Enter the **metro-rings** in addition to the **metro-ring** command as shown below.

```
device(config)#vlan 2
device(config-vlan-2)#metro-rings 1 2
device(config-vlan-2)#metro-ring 1
device(config-vlan-2-mrp-1)#name CustomerA
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)#metro-ring 2
device(config-vlan-2-mrp-2)#name CustomerB
device(config-vlan-2-mrp-2)#ring-interface ethernet 1/1/1 ethernet 1/1/2
device(config-vlan-2-mrp-2)#enable
```

Syntax: [no] metro-rings *ringid ringid . ..*

The *ring id* variables identify the metro rings you want to configure on the VLAN.

Syntax: [no] name *string*

The *string* parameter specifies a name for the ring. The name is optional, but it can be up to 20 characters long and can include blank spaces. If you use a name that has blank spaces, enclose the name in double quotation marks (for example: "Customer A").

Syntax: [no] master

Configures this node as the master node for the ring. Enter this command only on one node in the ring. The node is a member (non-master) node by default.

Syntax: [no] ring-interface ethernet *primary-if* **ethernet** *secondary-if*

The **ethernet** *primary-if* parameter specifies the primary interface. 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.

The **ethernet** *secondary-if* parameter specifies the secondary interface.

> **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.

Syntax: [no] enable

The **enable** command enables the ring.

Changing the hello and preforwarding times

You also can change the RHP hello time and preforwarding time. To do so, enter commands such as the following.

```
device(config-vlan-2-mrp-1)#hello-time 200
device(config-vlan-2-mrp-1)#preforwarding-time 400
```

These commands change the hello time to 200 ms and change the preforwarding time to 400 ms.

Syntax: [no] hello-time *ms*

Syntax: [no] preforwarding-time *ms*

The *ms* specifies the number of milliseconds. For the hello time, you can specify from 100 - 1000 (one second). The default hello time is 100 ms. The preforwarding time can be from 200 - 5000 ms, but must be at least twice the value of the hello time and must be a multiple of the hello time. The default preforwarding time is 300 ms. A change to the hello time or preforwarding time takes effect as soon as you enter the command.

Configuration notes for changing the hello and preforwarding times

- 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.
- You can use MRP ring diagnostics to determine whether you need to change the hello time and preforwarding time. Refer to Metro Ring Protocol diagnostics on page 34.

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.

Enabling MRP diagnostics

To enable MRP diagnostics for a ring, enter the following command on the Master node, at the configuration level for the ring.

```
device(config-vlan-2-mrp-1)#diagnostics
```

Syntax: [no] diagnostics

> **NOTE**
> This command is valid only on the master node.

Displaying MRP diagnostics

To display MRP diagnostics results, enter the following command on the Master node.

```
device#show metro 1 diag
Metro Ring 1 - CustomerA
=============
diagnostics results
```

```
Ring       Diag       RHP average     Recommended     Recommended
id         state      time(microsec)  hello time(ms)  Prefwing time(ms)
2          enabled    125             100             300
Diag frame sent     Diag frame lost
1230                0
```

Syntax: show metro *ring-id* **diag**

This display shows the following information.

TABLE 3 CLI display of MRP ring diagnostic information

Field	Description
Ring id	The ring ID.
Diag state	The state of ring diagnostics.
RHP average time	The average round-trip time for an RHP packet on the ring. The calculated time has a granularity of 1 microsecond.
Recommended hello time	The hello time recommended by the software based on the RHP average round-trip time.
Recommended Prefwing time	The preforwarding time recommended by the software based on the RHP average round-trip time.
Diag frame sent	The number of diagnostic RHPs sent for the test.
Diag frame lost	The number of diagnostic RHPs lost during the test.

If the recommended hello time and preforwarding time are different from the actual settings and you want to change them, refer to Metro Ring Protocol configuration on page 32.

Displaying MRP information

You can display the following MRP information:

- Topology group configuration information
- Ring configuration information and statistics

Topology group configuration information

To display topology group information, enter the following command.

Syntax: show topology-group [group-id]

Refer toDisplaying topology group information on page 292 for more information.

Ring configuration information and statistics

To display ring information, enter the following command.

```
device#show metro
Metro Ring 1
=============
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 1/1/1       primary            disabled            none                Regular
ethernet 1/1/2       secondary          forwarding          ethernet 2          Tunnel
RHPs sent          RHPs rcvd          TC RHPs rcvd        State changes
3                    0                  0                   4
```

Syntax: show metro [*ring-id*]

This display shows the following information.

TABLE 4 CLI display of MRP ring information

Field	Description
Ring id	The ring ID
State	The state of MRP. The state can be one of the following: • enabled - MRP is enabled • disabled - MRP is disabled
Ring role	Whether this node is the master for the ring. The role can be one of the following: • master • member
Master vlan	The ID of the master VLAN in the topology group used by this ring. If a topology group is used by MRP, the master VLAN controls the MRP settings for all VLANs in the topology group. **NOTE** The topology group ID is 0 if the MRP VLAN is not the master VLAN in a topology group. Using a topology group for MRP configuration is optional.
Topo group	The topology group ID.
Hello time	The interval, in milliseconds, at which the Forwarding port on the ring master node sends Ring Hello Packets (RHPs).
Prefwing time	The number of milliseconds an MRP interface that has entered the Preforwarding state will wait before changing to the Forwarding state. If a member port in the Preforwarding state does not receive an RHP within the Preforwarding time (Prefwing time), the port assumes that a topology change has occurred and changes to the Forwarding state. The secondary port on the Master node changes to Blocking if it receives an RHP, but changes to Forwarding if the port does not receive an RHP before the preforwarding time expires. **NOTE** A member node Preforwarding interface also changes from Preforwarding to Forwarding if it receives an RHP whose forwarding bit is on.
Ring interfaces	The device two interfaces with the ring. **NOTE** If the interfaces are part of a LAG, only the LAG interfaces are listed.
Interface role	The interface role can be one of the following: • primary – Master node - The interface generates RHPs. – Member node - The interface forwards RHPs received on the other interface (the secondary interface). • secondary - The interface does not generate RHPs. – Master node - The interface listens for RHPs. – Member node - The interface receives RHPs.

TABLE 4 CLI display of MRP ring information (continued)

Field	Description
Forwarding state	Whether MRP Forwarding is enabled on the interface. The forwarding state can be one of the following: • blocking - The interface is blocking Layer 2 data traffic and RHPs • disabled - The interface is down • forwarding - The interface is forwarding Layer 2 data traffic and RHPs • preforwarding - The interface is listening for RHPs but is blocking Layer 2 data traffic
Active interface	The physical interfaces that are sending and receiving RHPs. **NOTE** If a port is disabled, its state is shown as "disabled". **NOTE** If an interface is a trunk group, the member port which comes up first is listed.
Interface Type	Shows if the interface is a regular port or a tunnel port.
RHPs sent	The number of RHPs sent on the interface. **NOTE** This field applies only to the master node. On non-master nodes, this field contains 0. This is because the RHPs are forwarded in hardware on the non-master nodes.
RHPs rcvd	The number of RHPs received on the interface. **NOTE** On most Ruckus devices, this field applies only to the master node. On non-master nodes, this field contains 0. This is because the RHPs are forwarded in hardware on the non-master nodes. However, on the FastIron devices, the RHP received counter on non-master MRP nodes increment. This is because, on FastIron devices, the CPU receives a copy of the RHPs forwarded in hardware.
TC RHPs rcvd	The number of Topology Change RHPs received on the interface. A Topology Change RHP indicates that the ring topology has changed.
State changes	The number of MRP interface state changes that have occurred. The state can be one of the states listed in the Forwarding state field.
Interface Type	Shows if the interface is a regular port or a tunnel port.

MRP CLI example

The following examples show the CLI commands required to implement the MRP configuration shown in Figure 10 on page 31.

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....41
- VSRP configuration notes and feature limitations....43
- VSRP redundancy....43
- Master election and failover....43
- VSRP interval timers....48
- Configuring device redundancy using VSRP....48
- Configuring optional VSRP parameters....50
- Configuring authentication on VSRP interfaces....51
- Tracking ports and setting the VSRP priority....51
- Disabling backup pre-emption setting....52
- VSRP-aware security features....53
- VSRP fast start....54
- VSRP and MRP signaling....55

VSRP overview

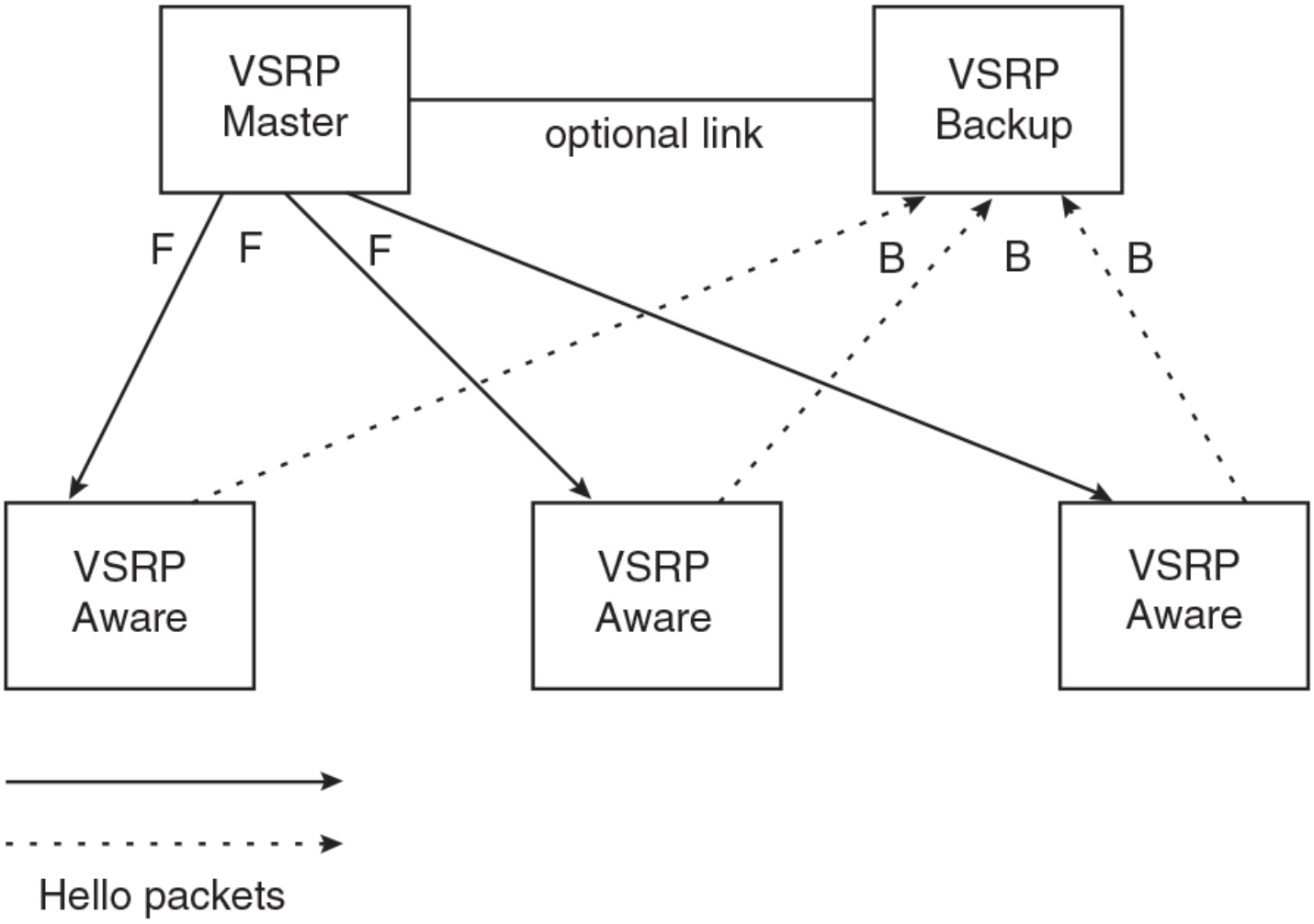
Virtual Switch Redundancy Protocol (VSRP) is a Ruckus proprietary protocol that provides redundancy and sub-second failover in Layer 2 and Layer 3 mesh topologies. Based on the Ruckus Virtual Router Redundancy Protocol Extended (VRRP-E), VSRP provides one or more backups for a device. If the active device 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 11 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 802.1Q-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 Switch with the higher management IP address becomes the Master.
  - Device with management IP addresses are preferred over switches without management IP addresses.
  - If neither of the switches has a management IP address, then the device with the higher MAC address becomes the Master. (VSRP compares the MAC addresses of the ports configured for the VRID, not the base MAC addresses of the devices.)
- Layer 3 devices - The Layer 3 device whose 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.

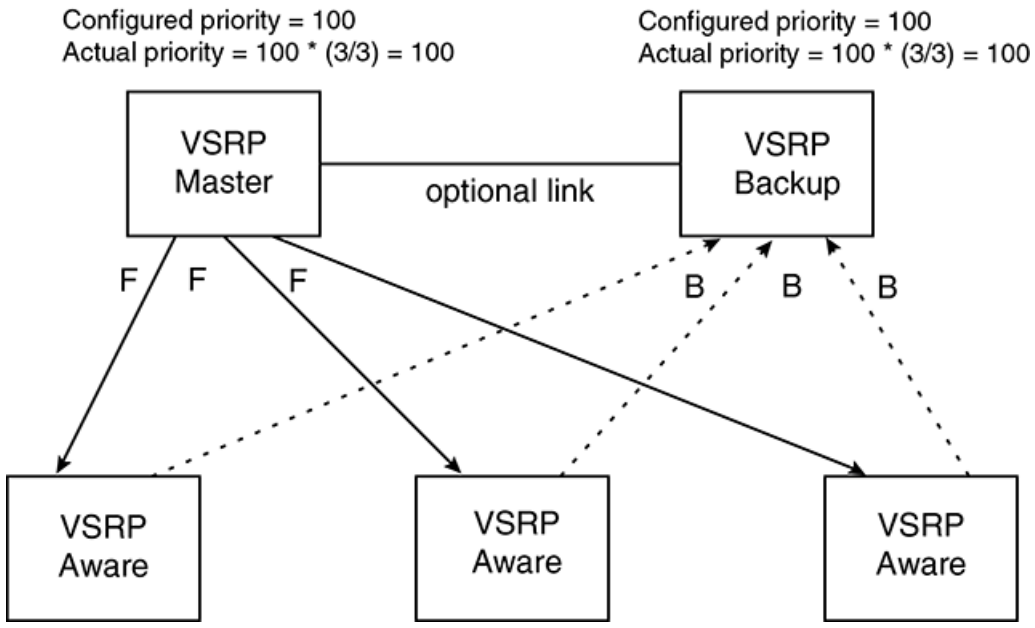
- If the Backup receives a Hello message with a higher priority than its own, the Backup resets its Dead Interval and returns to normal Backup status.
- If the Backup does not receive a Hello message with a higher priority than its own by the time the hold-down timer expires, the Backup becomes the new 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

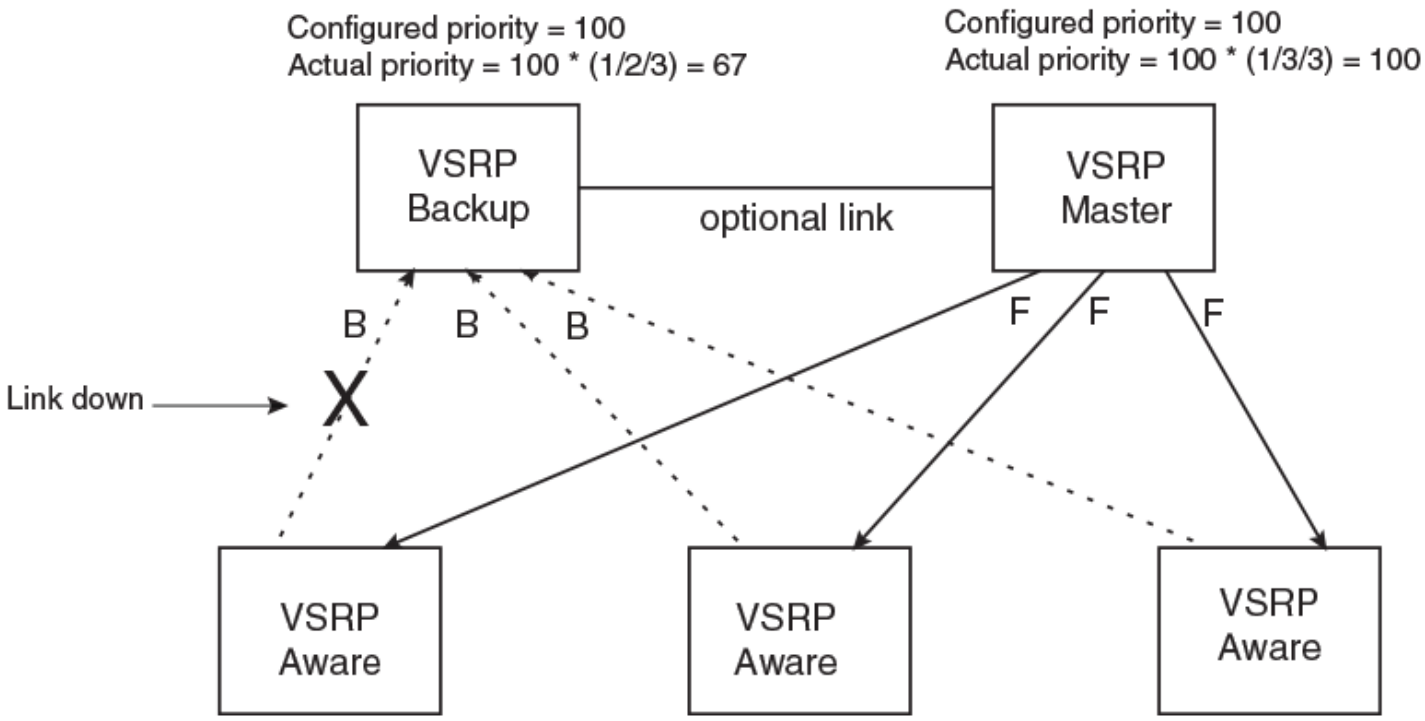
Each VSRP device has a VSRP priority for each VRID and its VLAN. The VRID is used during Master election for the VRID. By default, a device VSRP priority is the value configured on the device (which is 100 by default). However, to ensure that a Backup with a high number of up ports for a given VRID is 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 12 VSRP priority



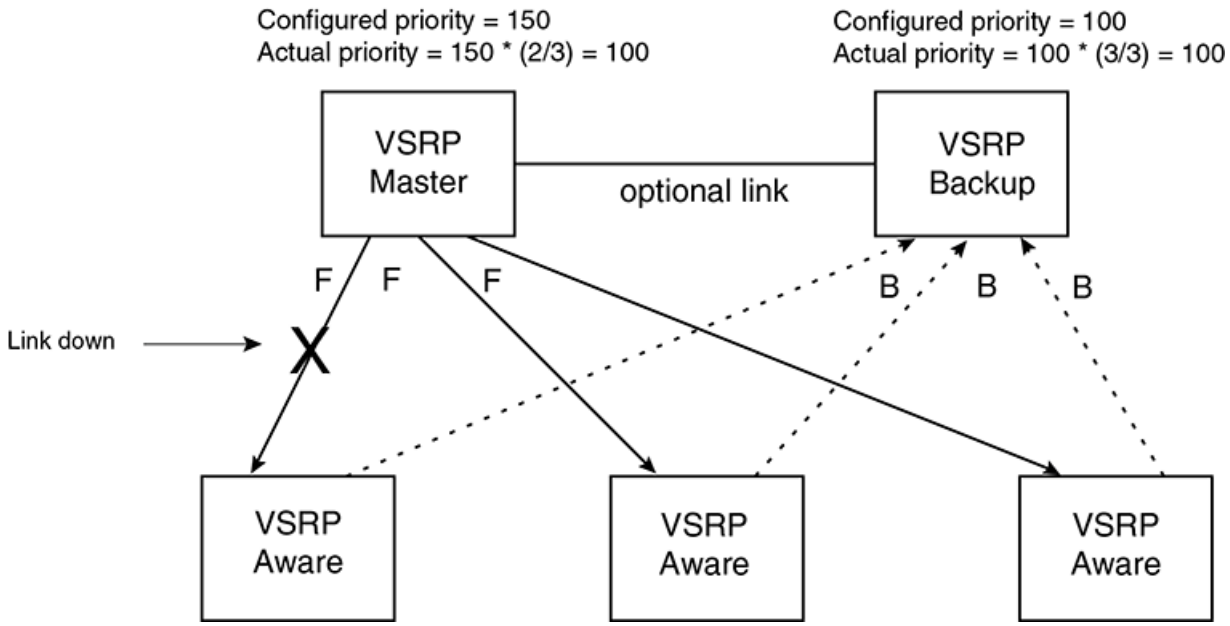
However, if one of the VRID ports goes down on one of the Backups, that Backup priority is reduced. If the Master priority is reduced enough to make the priority lower than a Backup priority, the VRID fails over to the Backup. The following figure shows an example.

FIGURE 13 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 13 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 14 VSRP priority bias

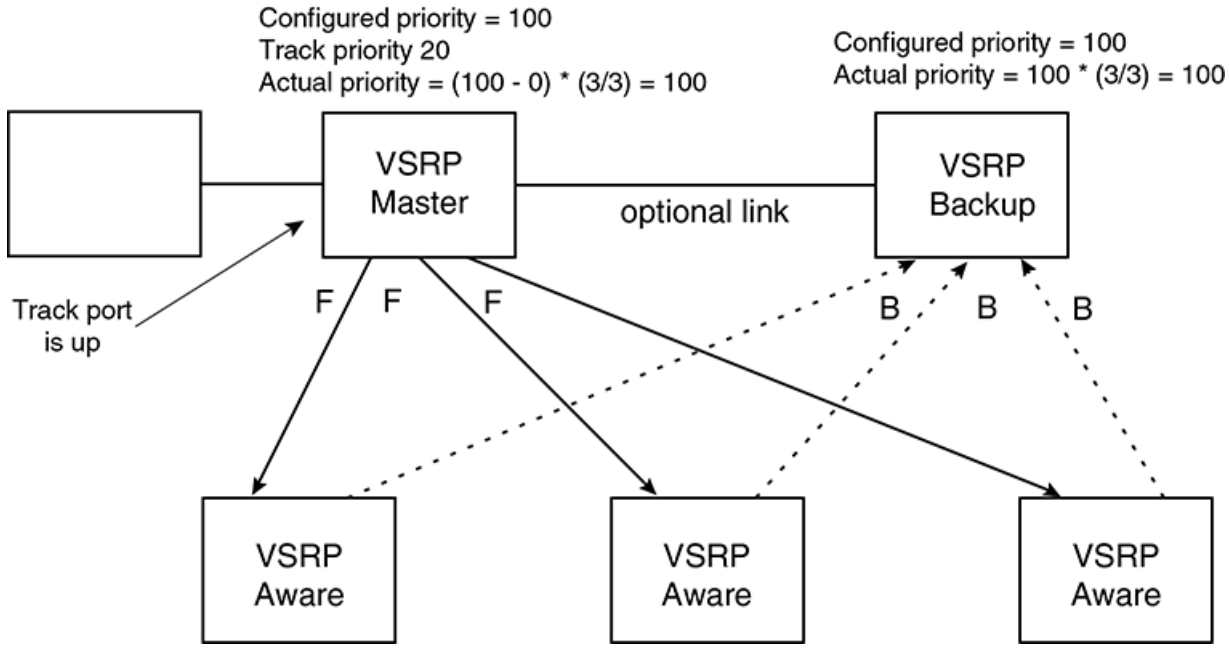


Track ports

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 whose state 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.

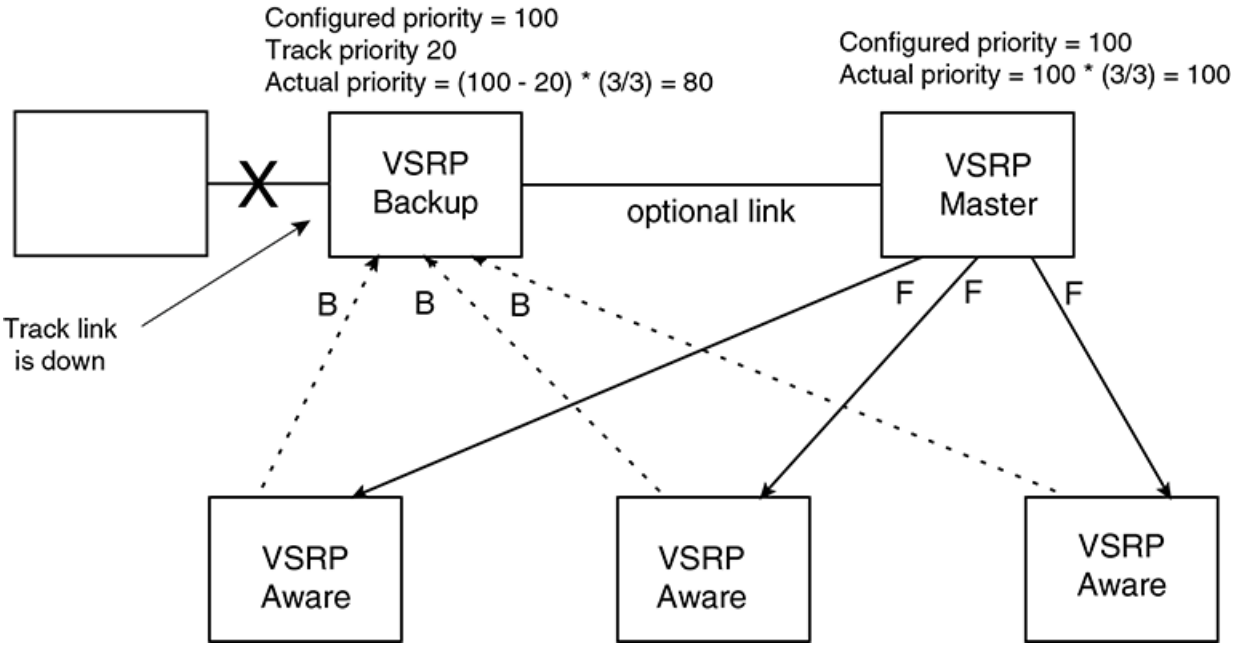
When you configure a track port, you assign a priority value to the port. If the port goes down, VSRP subtracts the track port priority value from the configured VSRP priority. For example, if the you configure a track port with priority 20 and the configured VSRP priority is 100, the software subtracts 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 15 Track port priority



In Figure 15, 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 16 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 also must 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 authenticaiton 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 hte 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 other 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 fats 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 shutdown all the ports, with the specified VRID, that belong to the VLAN when failover occurs.

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

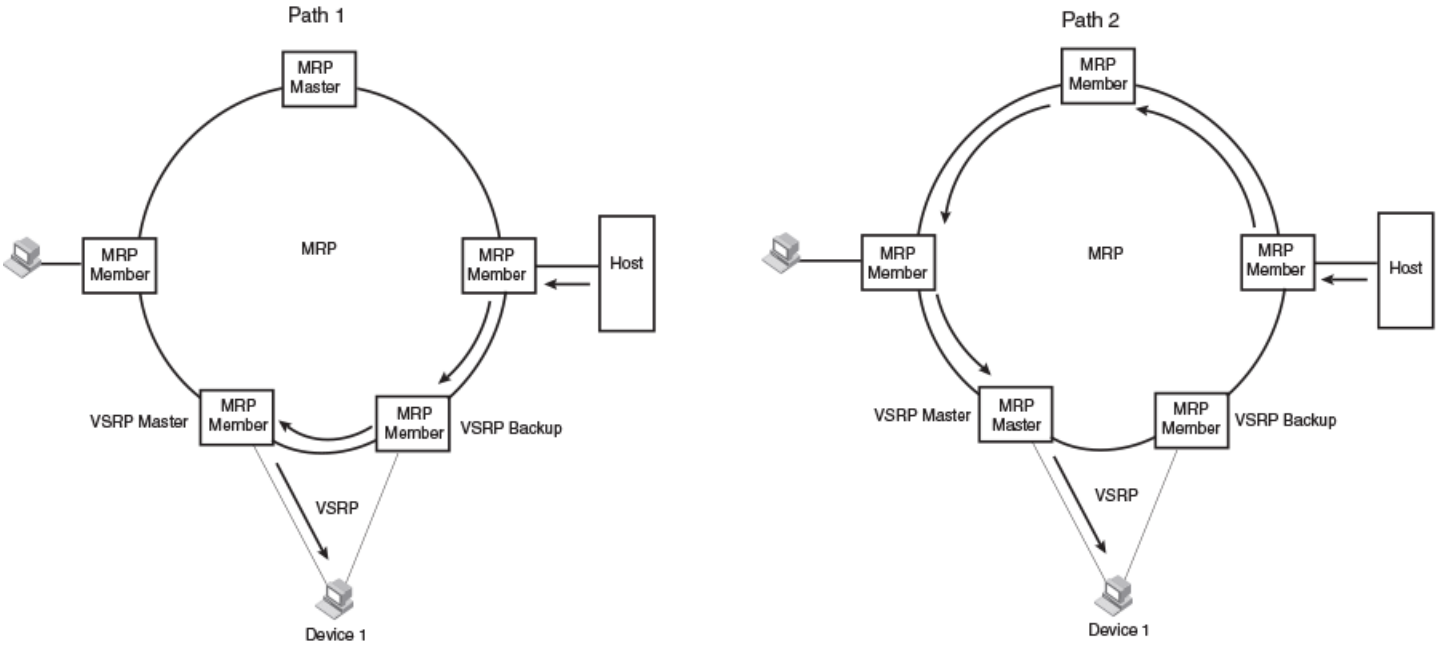
5. Verify the configuration using **show vsrp vrid** command.

```
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

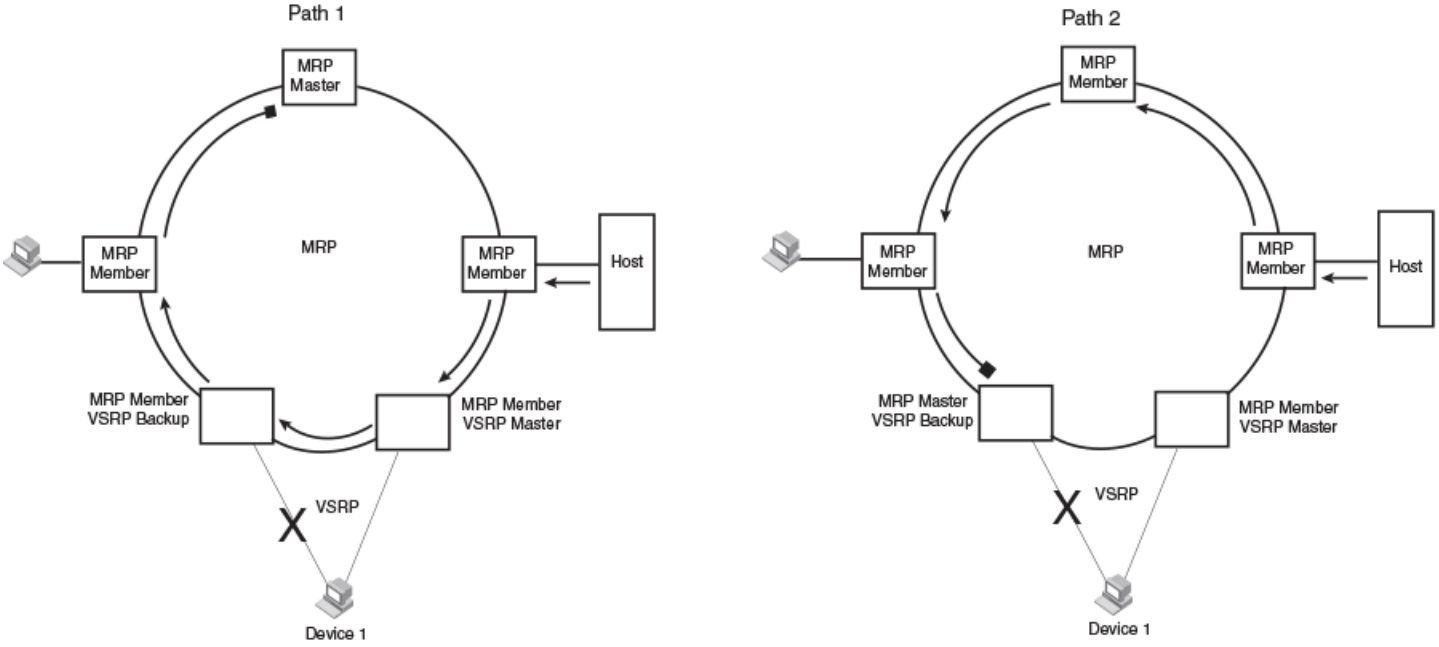
A device may connect to an MRP ring through VSRP to provide a redundant path between the device and the MRP ring. VSRP and MRP signaling ensures rapid failover by flushing MAC addresses appropriately. The host on the MRP ring learns the MAC addresses of all devices on the MRP ring and VSRP link. From 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 17 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 18 VSRP on MRP rings that failed over

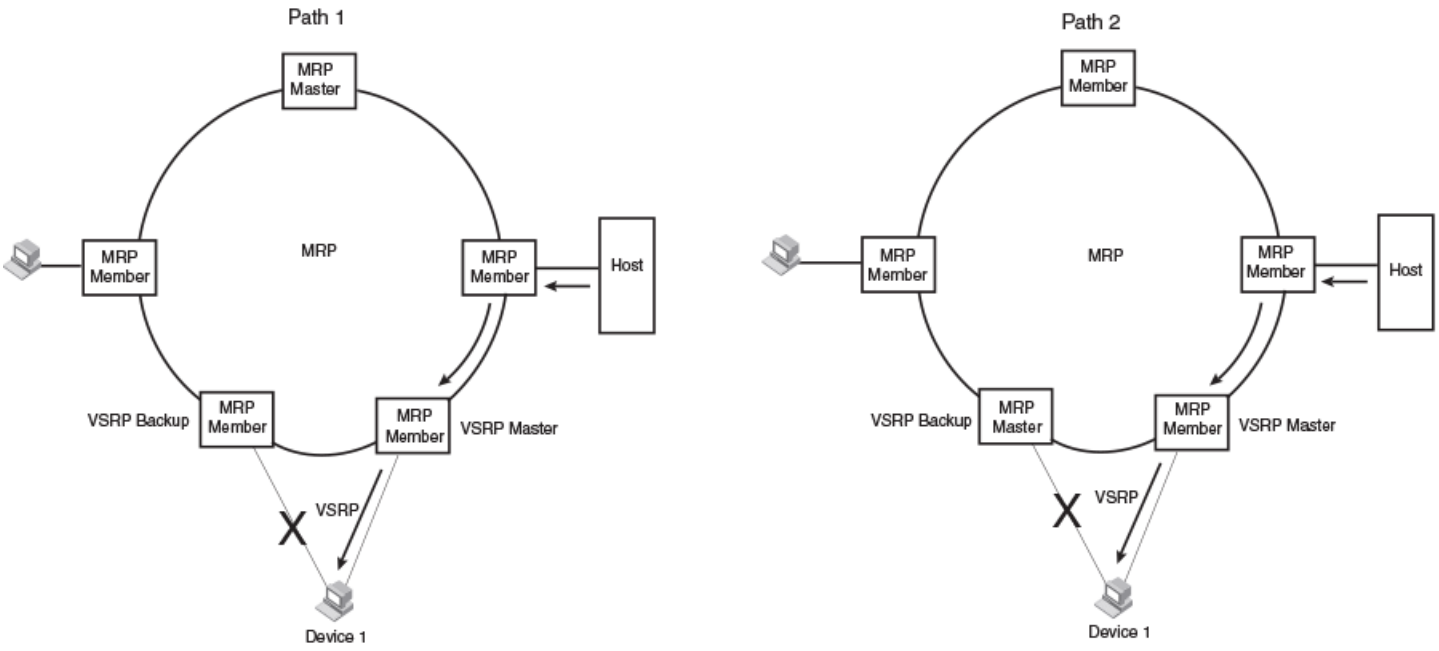


A signaling process for the interaction between VSRP and MRP ensures that MRP is informed of the topology change and achieves convergence rapidly. When a VSRP node fails, a new VSRP master is selected. The new VSRP master finds all MRP instances impacted by the failover. Then each MRP instance does the 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 19 New path established



There are no CLI commands used to configure this process.

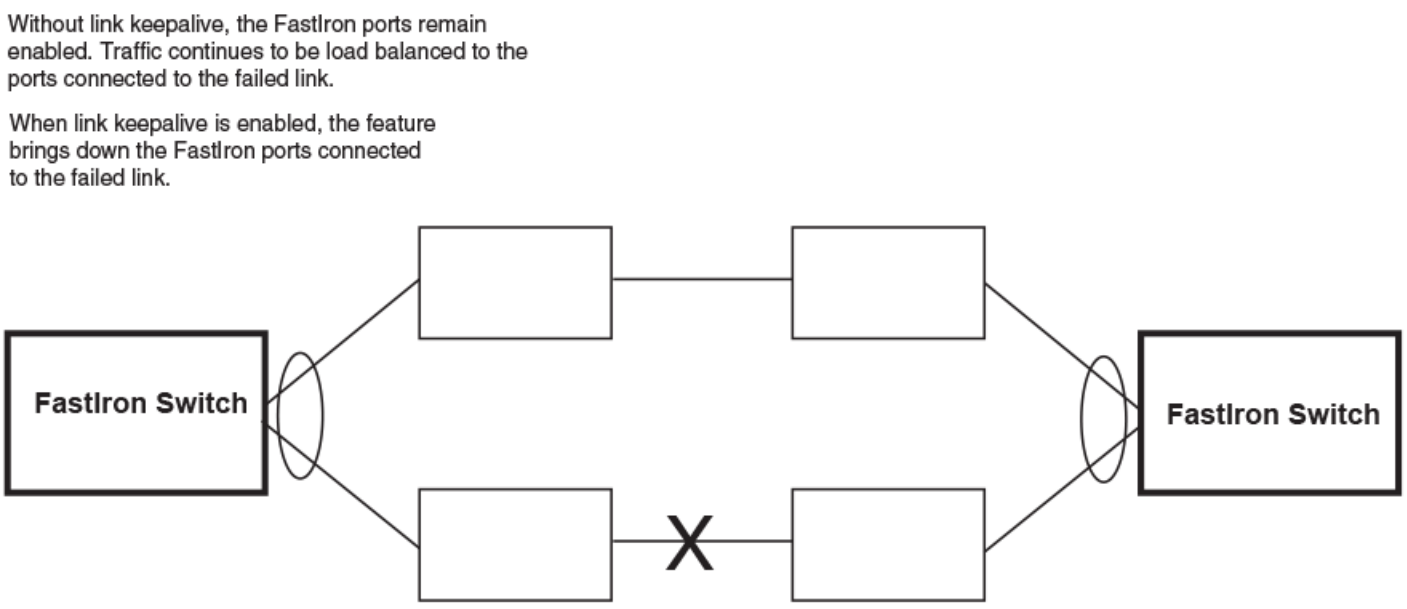
UDLD and Protected Link Groups

- UDLD overview..... 59
- UDLD for tagged ports..... 60
- Configuration notes and feature limitations for UDLD..... 60
- Enabling UDLD..... 60
- Enabling UDLD for tagged ports..... 61
- Changing the Keepalive interval..... 61
- Changing the keepalive retries..... 61
- Displaying information for all ports..... 61
- Displaying information for a single port..... 62
- Clearing UDLD statistics..... 63

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. This feature is useful for links that are individual ports and for trunk links. The following figure shows an example.

FIGURE 20 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

For configuration details, refer to Enabling UDLD for tagged ports on page 61.

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. Refer to Changing the hello and preforwarding times on page 34.

Enabling UDLD

You use this command to enable UDLD on a port for untagged control packets.

NOTE
This section shows how to configure UDLD for untagged control packets. To configure UDLD for tagged control packets, see Enabling UDLD for tagged ports.

To enable UDLD on a port, enter a command such as the following.

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

To enable the feature on a trunk group, enter commands such as the following.

```
device(config)# link-keepalive ethernet 1/1/1 ethernet 1/1/2
device(config)# link-keepalive ethernet 1/1/3 ethernet 1/1/4
```

Enabling UDLD for tagged ports

To enable ports to receive and send UDLD control packets tagged with a specific VLAN ID, enter commands such as the following.

```
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.

Syntax: [no] link-keepalive ethernet *port* [**vlan** *vlan-ID*]

For the *vlan-ID* variable, enter the ID of the VLAN that the UDLD control packets can contain to be received and sent on the port. If a VLAN ID is not specified, then UDLD control packets are sent out of the port as untagged packets.

> **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.

Changing the Keepalive interval

You use this command to change the link health-check packet send interval.

By default, ports enabled for UDLD send a link health-check packet once every 500 ms. You can change the interval to a value from 1 - 60, where 1 is 100 ms, 2 is 200 ms, and so on.

Changing the keepalive retries

You use this command to set how many retries a port to makes when sent health-checks receive no reply.

By default, a port waits one second to receive a health-check reply packet from the port at the other end of the link. If the port does not receive a reply, the port tries four more times by sending up to four more health-check packets. If the port still does not receive a reply after the maximum number of retries, the port goes down.

You can change the maximum number of keepalive attempts to a value from 3 - 64.

Displaying information for all ports

To display UDLD information for all ports, enter the following command.

```
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
```

Syntax: show link-keepalive

TABLE 5 CLI display of UDLD information

Field	Description
Total link-keepalive enabled ports	The total number of ports on which UDLD is enabled.
Keepalive Retries	The number of times a port will attempt the health check before concluding that the link is down.
Keepalive Interval	The number of seconds between health check packets.
Port	The port number.
Physical Link	The state of the physical link. This is the link between the Ruckus port and the directly connected device.
Logical Link	The state of the logical link. This is the state of the link between this Ruckus port and the Ruckus port on the other end of the link.
State	The traffic state of the port.
Link-vlan	The ID of the tagged VLAN in the UDLD packet.

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

```
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.

Syntax: show interfaces brief

Displaying information for a single port

To display detailed UDLD information for a specific port, enter a command such as the following.

```
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
```

Syntax: show link-keepalive [**ethernet** [*slotnum***/**] *portnum*]

TABLE 6 CLI display of detailed UDLD information

Field	Description
Current State	The state of the logical link. This is the link between this Ruckus port and the Ruckus port on the other end of the link.
Remote MAC Addr	The MAC address of the port or device at the remote end of the logical link.
Local Port	The port number on this Ruckus device.

TABLE 6 CLI display of detailed UDLD information (continued)

Field	Description
Remote Port	The port number on the Ruckus device at the remote end of the link.
Local System ID	A unique value that identifies this Ruckus device. The ID can be used by Ruckus technical support for troubleshooting.
Remote System ID	A unique value that identifies the Ruckus device at the remote end of the link.
Packets sent	The number of UDLD health-check packets sent on this port.
Packets received	The number of UDLD health-check packets received on this port.
Transitions	The number of times the logical link state has changed between up and down.
Link-vlan	The ID of the tagged VLAN in the UDLD packet.

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. An example is given below.

```
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 use this command to clear UDLD statistics.

To clear UDLD statistics, enter the following command.

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

Link Aggregation Group

- Overview of link aggregation 65
- LAG virtual interface 66
- LAG formation rules 66
- Configuration notes for FastIron devices in a traditional stack 68
- Maximum number of LAGs 69
- Upgrade and downgrade notes 70
- LAG Load Sharing 70
- LAG hashing on stacking products 71
- Symmetric load balancing 71
- Resilient hashing 74
- Configuring a LAG 75

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.

The benefits of link aggregation are summarized as follows:

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

This chapter describes how to configure Link Aggregation Groups (LAG). You can also 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 - This LAG type uses the Link Aggregation Control Protocol (LACP), to maintain aggregate links over multiple port. 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 whose connection is no longer active.

NOTE
The LAG functionality was referred to as Trunk Groups in previous releases.

Keep Alive LAGs - In a Keep Alive LAG a single connection between a single port on 2 Ruckus devices is established. In a keep alive LAG, LACP PDUs are exchanged between the 2 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
The Keep Alive LAG functionality was referred to as Single Link LACP in previous releases.

The new LAG configuration procedures supersede the previous configurations procedures for LAGs and Dynamic Link Aggregation. When a Ruckus device is upgraded to 08.0.61 any configurations for LAGs or Dynamic Link Aggregation defined in releases prior to 08.0.61 will be converted to a 08.0.61 (and later) compatible LAG configuration.

LAG virtual interface

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 will not exist from 8.0.61 release onwards. Instead, all features running over LAG will be anchored on the LAG virtual interface. All member ports of the LAG are treated as secondary ports and hence, 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.

A new mode called LAG virtual interface is introduced and the properties of 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 support are similar to an Ethernet interface. The configuration of the Link Aggregation Groups using the LAG virtual interface is described in Creating a Link Aggregation Group (LAG) on page 76. Like Ethernet interfaces, the LAG virtual interface also supports multi interface mode.

Any new configuration or changes made to the LAG virtual interface is 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. If the member ports are mentioned in the interface command, it automatically moves to the LAG virtual interface mode in which that port is a member.

NOTE
LAG virtual interface is applicable only for static LAG and dynamic LAG and is not supported for Keep-alive LAGs. LAG virtual interface configuration commands are not allowed for SPX LAG virtual interface.

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

To view the LAG virtual interface details, refer Displaying LAG information on page 82.

LAG virtual interface MAC

The L2 protocols (xSTP, MRP) running over the LAG use the LAG MAC as the source MAC. 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 can be assigned to the LAG virtual interface using the **lag-mac** command.

NOTE
If VE/L3 is configured on the LAG, the **show interface brief** command output displays the stack-mac as the LAG virtual interface MAC.

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.
- All ports configured in a LAG must be of same bandwidth and type, for example, all 10 GbE ports.
- 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. Layer 2 , Layer 3 and Layer 4 (ACL) requirements are listed below:
- Layer 2 requirements:

  The LAG is rejected if the LAG ports:

  - Do not have the same untagged VLAN component
  - Do not share the same VLAN membership or do not share the same uplink VLAN membership
  - Are configured as member ports of another LAG
- Layer 3 requirements:

  The LAG is rejected if any of the member port has any Layer 3 configurations, such as IPv4 or IPv6 address, OSPF, RIP, RIPng, IS-IS, route-only, and so on.
- Layer 4 (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)
  - Port dual-mode
  - Default port speed and duplex
  - Configured port speed and duplex
  - 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 combination of copper and fiber ports, even if they are of the same speed, cannot be members of the same LAG.
- A LAG is supported on 1 GbE, 10 GbE, 40 GbE, and 2.5G 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 under 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.
- Ruckus ICX 7450 devices do not support change in speed of a 2.5G paired port, if at least one of the port is part of a LAG. To resolve this issue, you need to delete the LAG and maintain the same speed on all ports in the LAG.

  NOTE
  If both the ports are 2.5G, and one of the ports is not part of a LAG, then the device does not allow speed change on this port as well. For example: 25 and 26 is a 2.5G port pair. If 25 is part of a LAG, then the device does not change the speed to 26 without deleting the LAG where 25 is part of.

Configuration notes for FastIron devices in a traditional stack

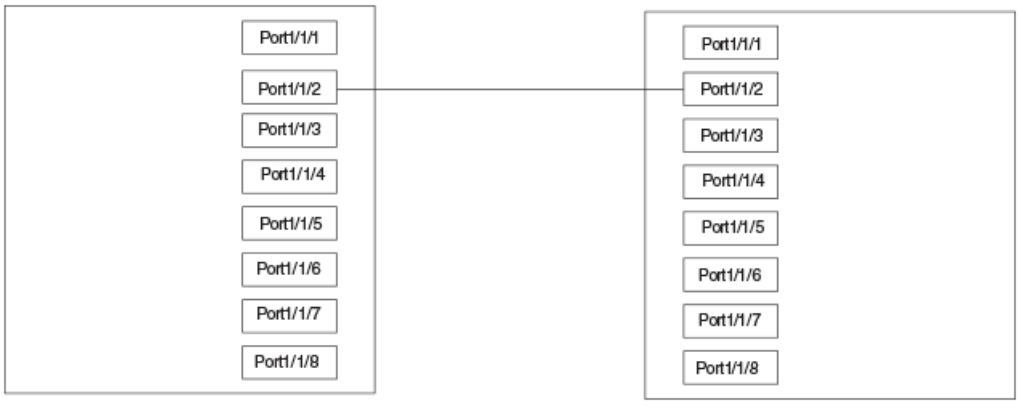
In a Ruckus traditional stack system, a LAG may have port members distributed across multiple stack units. Both static and dynamic LAGs are supported.

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. This can result in self-looped ports.
- You cannot configure a LAG between CB and PE ports in SPX stack.
- 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.

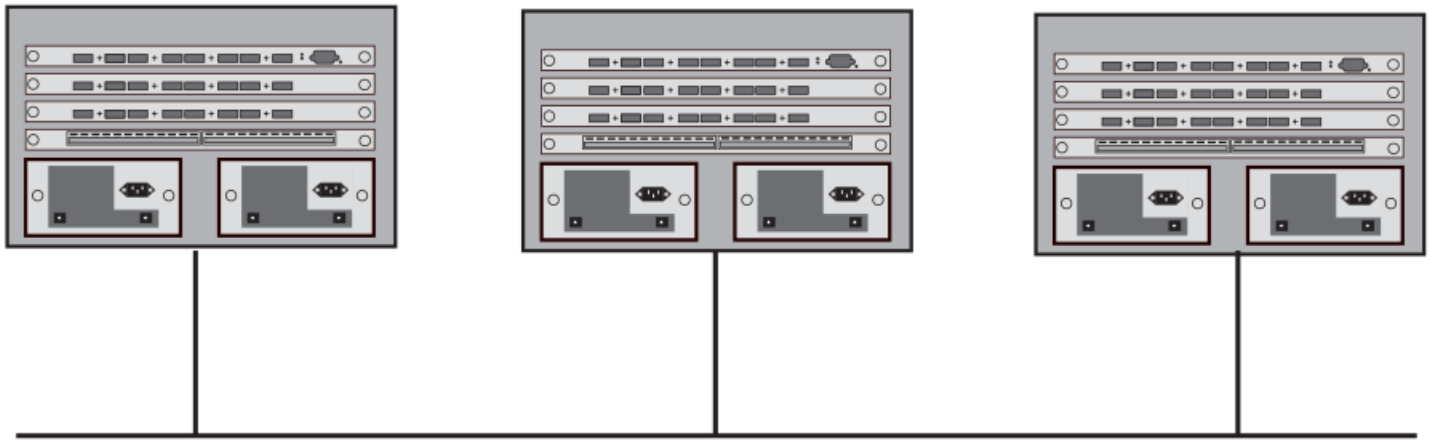
The following figure displays an example of a valid, keep-alive LAG link between two devices. This configuration does not aggregate ports but uses the LACP PDUs to maintain the connection status between the two ports.

FIGURE 21 Example of a 1-port keep-alive LAG



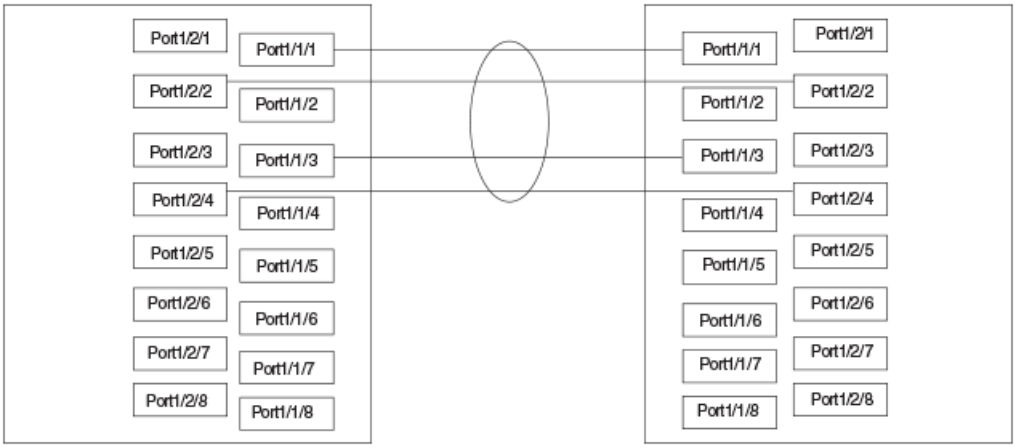
The following figure shows an example of a valid 2-port LAG link between devices where the ports on each end are on the same interface module. Ports in a valid 2-port LAG on one device are connected to two ports in a valid 2-port LAG on another device.

FIGURE 22 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 23 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 7 Maximum number of LAGs

Model	Maximum number of LAGs		Valid number of ports in a group
	Static	LACP	
ICX 7750 ICX 7450 ICX 7250	256	256	1 to 16 **NOTE** The Ruckus ICX device can scale up to a maximum of 2048 LAG ports only.
ICX 7150	128	128	1 to 8 Ruckus ICX 7150 can scale up to 1024 and is also limited to the number of ports on the device.

Upgrade and downgrade notes

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

Auto downgrade is not supported. Save the old config in the previous configuration format before upgrade followed by 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 unknown unicast, multicast, and broadcast 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.

Unknown unicast, multicast, and broadcast 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.

> **NOTE**
> ICX 7150 devices handle broadcast, unknown unicast, and multicast IP traffic distribution differently. ICX 7150 LAG hashing for these traffic types is based on source and destination MAC addresses.

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 8 LAG Load sharing on Ruckus ICX devices

Traffic type	Load balancing method[1]
Layer 2 bridged non-IP	Source and destination MAC addresses
Layer 2 bridged TCP/UDP	Source and destination MAC addresses, source and destination IP addresses, and source and destination TCP/UDP ports
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. Stacking trunks on ICX 7150 devices are an exception. Just as on ICX 7150 Layer 2 LAGs, ICX 7150 stacking trunks hash traffic based on traffic type, for example Layer 2, Layer 3, and Layer 4 header information.

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.

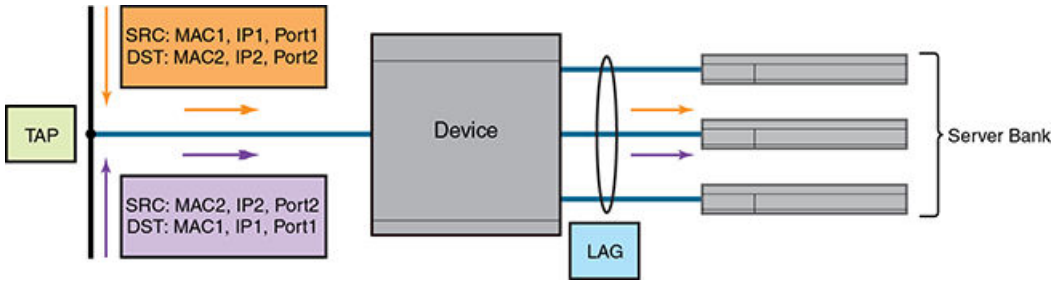
For many monitoring and security applications, bidirectional conversations flowing through the system must be carried on the same port of a LAG. For network telemetry applications, network traffic is tapped and sent to a Ruckus ICX device, which can

[1] In ICX 7150 devices, the LAG hashing scheme is different from other ICX platforms. As a result, the user may observe uneven distribution of packets when the traffic pattern is sequential or incremental in the same step, for example, when both the source and the destination IP address are incremented in the same step.

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 7750, Ruckus ICX 7450, and Ruckus ICX 7250.

FIGURE 24 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.

Run the **show running-config** command to check if symmetric load balancing is enabled.

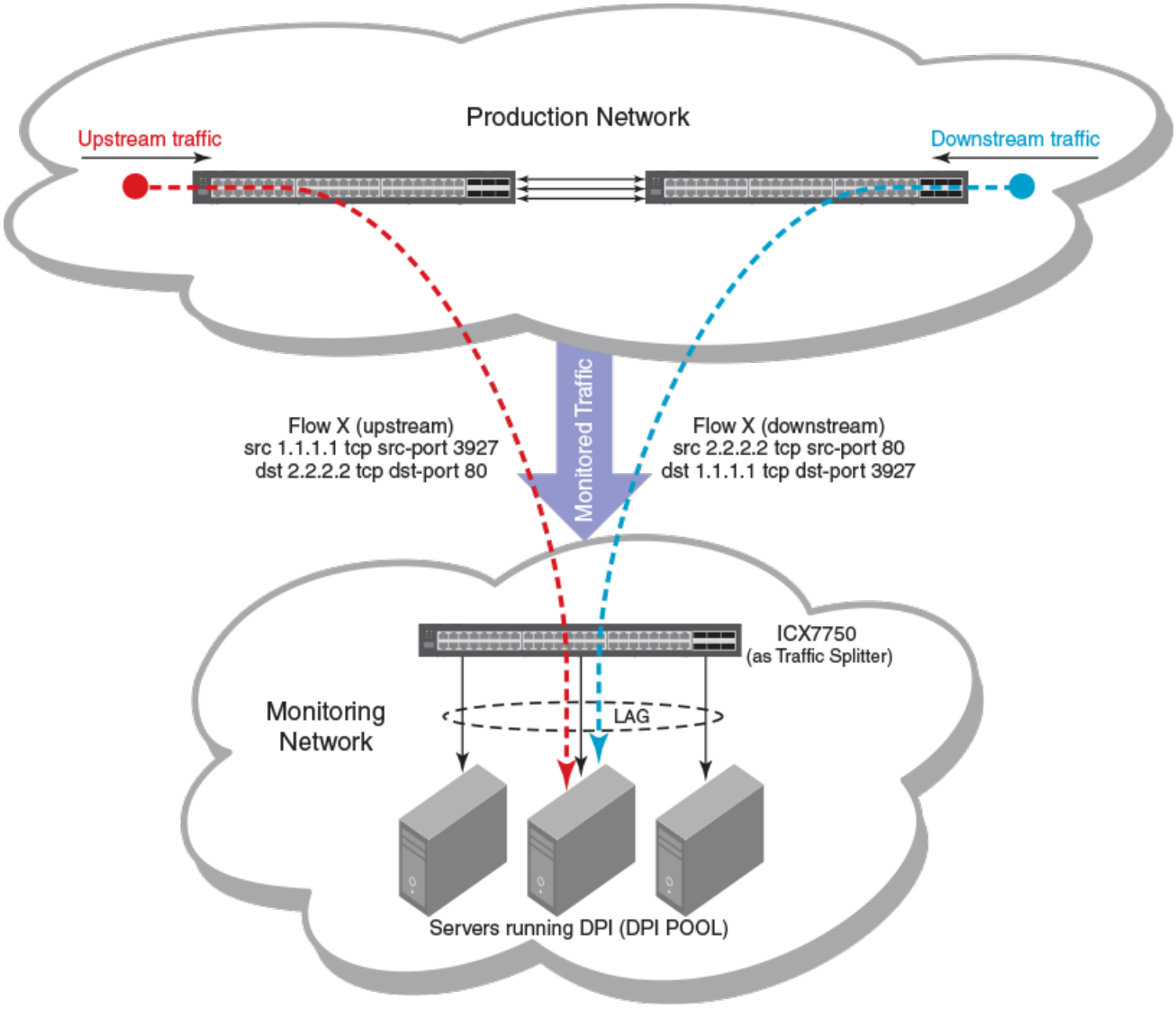
NOTE
Symmetric load balancing is a system level configuration and may affect load sharing among LAG members as compared to non-symmetric load balancing and the ECMP next hop load sharing by not fairly utilizing all the LAG links. It might also affect load sharing within a stack trunk in case of broadcast,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

TABLE 9 Fields used for hash calculation based on packet types

Packet type	Hashing field	Is symmetric load balancing supported on Ruckus ICX 7xxx 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 7750 as a traffic splitter in a DPI solution

FIGURE 25 Symmetric load balancing in Ruckus ICX 7750



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 ICX 7750 is deployed as a traffic splitter. There are multiple servers hosting the DPI application and connected to ICX 7750. All monitored traffic is transparently flooded onto the VLAN and is load-balanced among the outgoing ports connected to the DPI pool.

NOTE
For our analysis, we assume that the bidirectional traffic pertaining to the same SIP-DIP pair and/ or same layer 4 source/destination pair should go to the same DPI (connected to one of the LAG port).

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, 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, 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.

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 7750 devices 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 10 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. Create a static or dynamic LAG using the existing LAG command line interfaces.

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

2. In the LAG mode, enter the **trunk-type** command to 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>
```

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. Configure 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. 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-lg111)# acl-logging
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)# dual-mode
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 filter-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 an LACP timeout

In a dynamic or keep-alive LAG, a port's timeout can be configured as short (3 seconds) or long (120 seconds). After you configure a port timeout, the port remains in that timeout mode even it is up or down.

All the ports in a LAG should have the same timeout mode. This requirement is checked when the LAG is enabled on the ports. For example, to configure a port for a short LACP timeout, use the following command.

```
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)#
```

Syntax: [no] lacp-timeout [**long** | **short**]

To delete the configuration, use the **no** form of this command.

The **long** keyword configures the port for the long timeout mode-120 seconds. With the long timeout, an LACPDU is sent every 30 seconds. If no response comes from its partner after 3 LACPDUs are sent, a timeout event occurs, and the LACP state machine transition to the appropriate state based on its current state.

The **short** keyword configures the port for the short timeout mode--3 seconds. In the short timeout configuration, an LACPDU is sent every second. If no response comes from its partner after 3 LACPDUs are sent, a timeout event occurs, and the LACP state machine transitions to the appropriate state based on its current state.

If you specify neither **long** nor **short** , the state machine operates based on the standard IEEE specification as its default behavior. The original IEEE specification says that the state machine starts with short the timeout and moves to the long timeout after the LAG is established. However, sometimes a vendor’s implementation always uses either the short timeout or the long timeout without changing the timeout. Ruckus provides this command so that you can configure Ruckus devices to interoperate with other vendor’s devices.

> **NOTE**
> This configuration is applicable to the configuration of dynamic or keep-alive LAGs only.

Specifying the LAG threshold for a LAG group

You can configure the Ruckus device to disable all of the ports in a LAG group 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 group has 8 ports, and the threshold for the LAG group is 5, then the LAG group is disabled if the number of active ports in the LAG group drops below 5. If the LAG group is disabled, then traffic is forwarded over a different link or LAG group.

> **NOTE**
> This configuration is only applicable for static LAGs.

For example, the following commands establish a LAG group consisting of 4 ports, then establish a threshold for this LAG group of 3 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
```

In this example, if the number of active ports drops below 3, then all the ports in the LAG group are disabled.

Syntax: [no] trunk-threshold *number*

You can specify a threshold from 1 (the default) up to the number of ports in the LAG group.

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 the will not function properly.

Disabling ports within a LAG

You can disable an individual port within a LAG using the disable command within the LAG configuration as shown in the following.

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

To disable a port belonging to a keep-alive LAG, you need to configure from the interface level.

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

Enabling ports within a LAG

You can enable an individual port within a LAG using the enable command within the LAG configuration as shown in the following.

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

To enable a port belonging to a keep-alive LAG, you need to configure 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)#
```

Deleting 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:

- If removal of 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.

To delete port 1/3/1 which is in the "enabled" state from a LAG named "blue", use the following command:

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

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 member ports individually. If a new port is added to a LAG and if 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.

To monitor LAG virtual interface, run the following commands.

```
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   4
  Output monitoring     : (LAG)     1
```

To monitor traffic on an individual port in a LAG group, run 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
```

Syntax: [no] monitor { **ethe-port-monitored** *stack***/** *slot* **/** *port* | **named-port-monitored** *name* } [**ethernet** [*stack***/** *slot* **/** *port*]] { **input** | **output** | **both** }

Use the **ethe-port-monitored** option with the appropriate **[stack/***slot***/***port***]** variable to specify a Ethernet port within the LAG that you want to monitor.

Use the **named-port-monitored** option with the appropriate **name** variable to specify a named port within the LAG that you want monitor.

The **ethernet** *stack/slot/port* parameter specifies the port to which the traffic analyzer is attached.

The **input, output, and both** parameters specify the traffic direction to be monitored.

Assigning a name to a port within a LAG

You can assign a name to an individual port within a LAG using the **port-name** command within the LAG configuration as shown in the following.

```
device(config)# lag test dynamic id 1
device(config-lag-test)# ports ethernet 1/1/1 to 1/1/3
device(config-lag-test)# port-name "Ruckus lag" ethernet 1/1/1
```

The *name* variable specifies the port name. The name can be up to 255 characters long.

NOTE
Port name with space must be enclosed within double quotation marks.

Use the **ethernet** option with the appropriate *stack/slot/port* variable to apply the specified name to an Ethernet port within the LAG.

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

You can enable sFlow forwarding on an individual port within a LAG using the **sflow-forwarding** command within the LAG configuration as shown in the following example.

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

Use the **ethernet** option with the appropriate *stack/slot/port* variable to specify a Ethernet port within the LAG that you want to enable sFlow forwarding for.

Use the **port-name** option with the appropriate *name* variable to specify a named port within the LAG that you want to enable sFlow forwarding for.

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

You can set the sFlow sampling rate for a LAG using the **sflow sample** command within the LAG configuration as shown in the following.

```
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.

You can rename the LAG using the **update-lag-name** command within the LAG configuration mode. The new name provided must be unique and unused. The LAG configuration mode will exit after successful name update.

```
device(config)# lag blue static id 1
device(config-lag-blue)# update-lag-name blue1
INFORMATION: Lag blue is updated to new name blue1
```

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:             120, default: 120
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:             120, default: 120
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
Openflowis 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 Ruckus ICX 7750devices.

NOTE
LAG should be undeployed to configure the **failover all** command.

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

In this example, the LAG **failover all** command is enabled on all ports.

```
device(config)# lag one dynamic id 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

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

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

Run the **force-up ethernet** command in dynamic LAG configuration mode.

The following example shows PXE boot support enabled on member port 3/1/1 of a dynamic LAG R4-dyn.

```
device(config)# lag R4-dyn dynamic id 1
device(config-lag-R4-dyn)# force-up ethernet 3/1/1
```

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 11 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.

Multi-Chassis Trunking

- Multi-Chassis Trunking Overview 87
- Basic MCT configuration 93
- Cluster client automatic configuration 99
- MCT failover scenarios 101
- Layer 2 behavior with MCT 104
- Layer 3 behavior with MCT 113
- Displaying MCT information 129
- Single-level MCT configuration example 134
- Two-level MCT configuration example 137
- MCT configuration examples using STP 142

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.

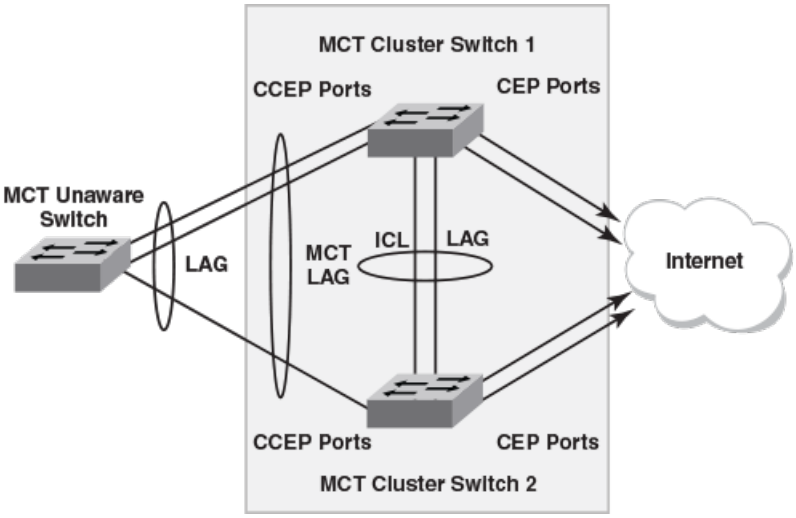
Standard static or dynamic LACP trunks provide link-level redundancy and increased capacity. However, trunks do not provide device-level redundancy. If the device to which the trunk is attached fails, the entire trunk loses network connectivity. Two devices are needed for network resiliency with trunked 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 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 detections, 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. MCT eliminates the single point of failure that exists at the device level when all links of a trunk terminate on the same device without the overhead associated with spanning tree. 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 table shows a basic MCT configuration. The MCT originates at a single MCT-unaware server or switch and terminates at two MCT-aware devices.

FIGURE 26 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 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 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 non-MCT devices.
- Known unicast traffic received on Cluster Edge Ports (CEP) or ICL ports is forwarded to the destination port.
- For unknown unicast, multicast, and broadcast traffic received on ICL ports, the forwarding behavior depends on the peer MCT device's ability to reach the same client.
- Unknown unicast, multicast, and broadcast traffic received from 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 should be different from the cluster ID on a stage 2 pair of switches.

MCT terminology

- 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/router but is neither a CCEP not 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. CCP peer management deals with establishing and maintaining a TCP transport session between peers, while CCP client management provides event-based, reliable packet transport to CCP peers.
- Inter-Chassis Link (ICL): A single-port or multi-port 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 (switches) that is 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 in an MCT cluster.
- MCT peer device: From the perspective of an MCT cluster device, the other device 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 protocol runs over this VLAN. The interface can be a single link or a LAG. In a LAG, it should be configured on the LAG virtual interface. The MCT session VLAN subnet is not distributed in routing protocols using redistribute commands.
- RBridgeID: RBridgeID 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

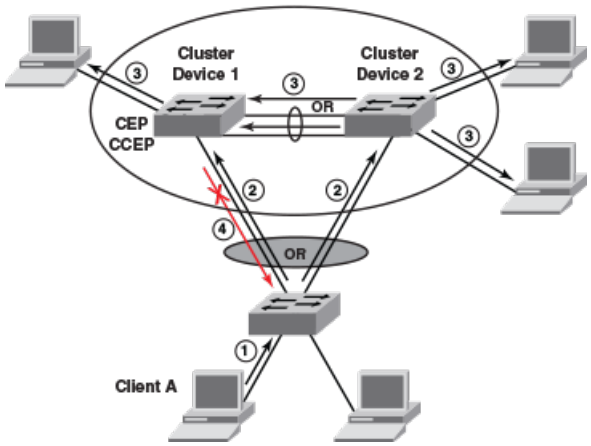
MCT can be deployed in a single-level configuration that includes two MCT cluster devices or in a cascading configuration, where a pair of MCT cluster devices operate as switches, and another pair of cluster devices operates as routers. Refer to Single-level MCT configuration example on page 134 for a single-level illustration and configuration example, and Two-level MCT configuration example on page 137 for a two-level or cascading configuration example.

Basic MCT data flow works as follows.

Broadcast, unknown unicast, and multicast (BUM) 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. It passes to the peer cluster device over the ICL link, where it is sent to the peer device's local CEPs.
4. Traffic does not pass back down to the client through the CCEP.

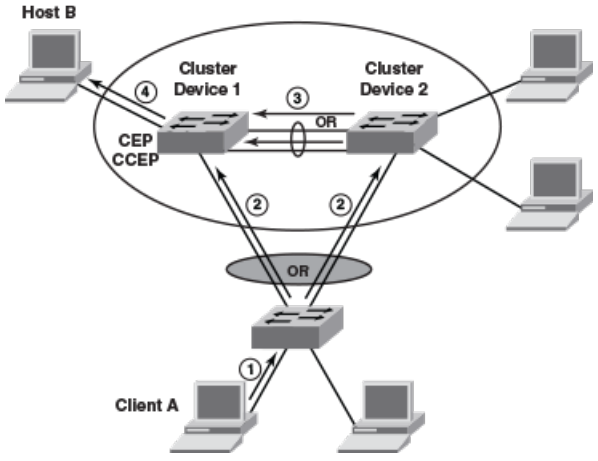
FIGURE 27 MCT data flow - BUM traffic from CCEP



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 28 MCT data flow - unicast traffic from CCEP

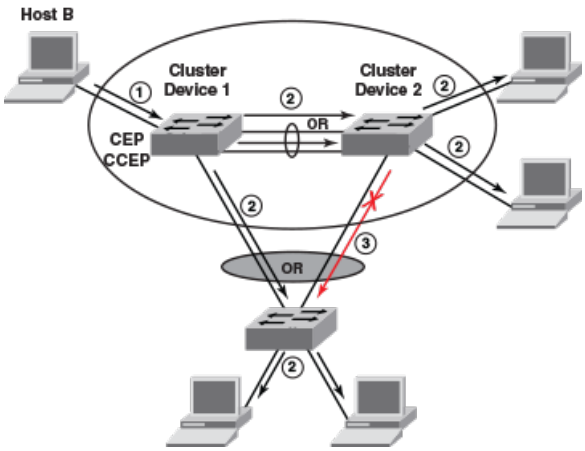


Broadcast, unknown unicast, and multicast (BUM) traffic from a client through a CEP

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 and is also sent to any local CCEPs and CEPs. Once traffic is received on the peer cluster device, it will be sent to its local CEPs.

3. Traffic does not pass back down to the client through the CCEP.

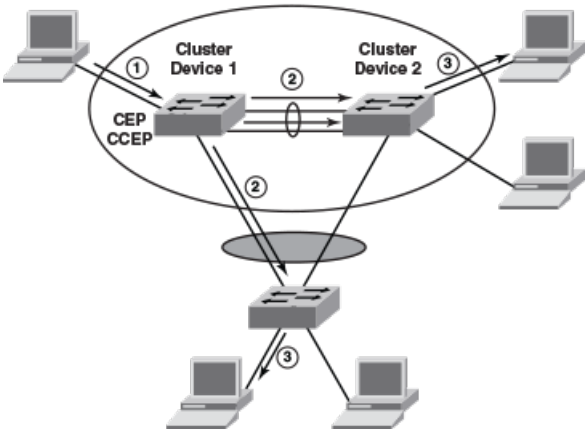
FIGURE 29 MCT data flow - BUM traffic from a CEP



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.

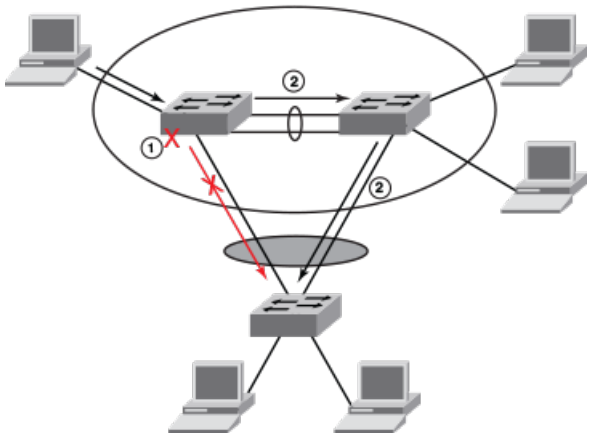
FIGURE 30 MCT data flow - unicast traffic from a CEP



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 31 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 Unsupported 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).
- STP and RSTP.
- Ingress ACLs on all MCT ports, except for ICL ports. Egress ACLs are supported only on MCT Cluster Edge Ports (CEPs). Egress ACLs are not supported on MCT CCEPs.
- QoS and MAC filters 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.
- DAI and DHCP snooping for clients connected through CEPs. They must be configured independently on both cluster devices.
  - If the trusted ports are off the CCEP, the **arp inspection trust** or **dhcp snoop trust** command must be used on the CCEPs and ICL ports.
  - DHCP and ARP entries are created on both MCT cluster devices if the flow traverses both the CCEP and ICL.
- 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.
- Hitless upgrade. If the upgrade 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. Hitless upgrades are only supported between minor (letter, patch) releases, for example, FastIron 08.0.70a to 08.0.70b. Hitless upgrades between major releases, for example, FastIron 08.0.70 to 08.0.80, are not supported.

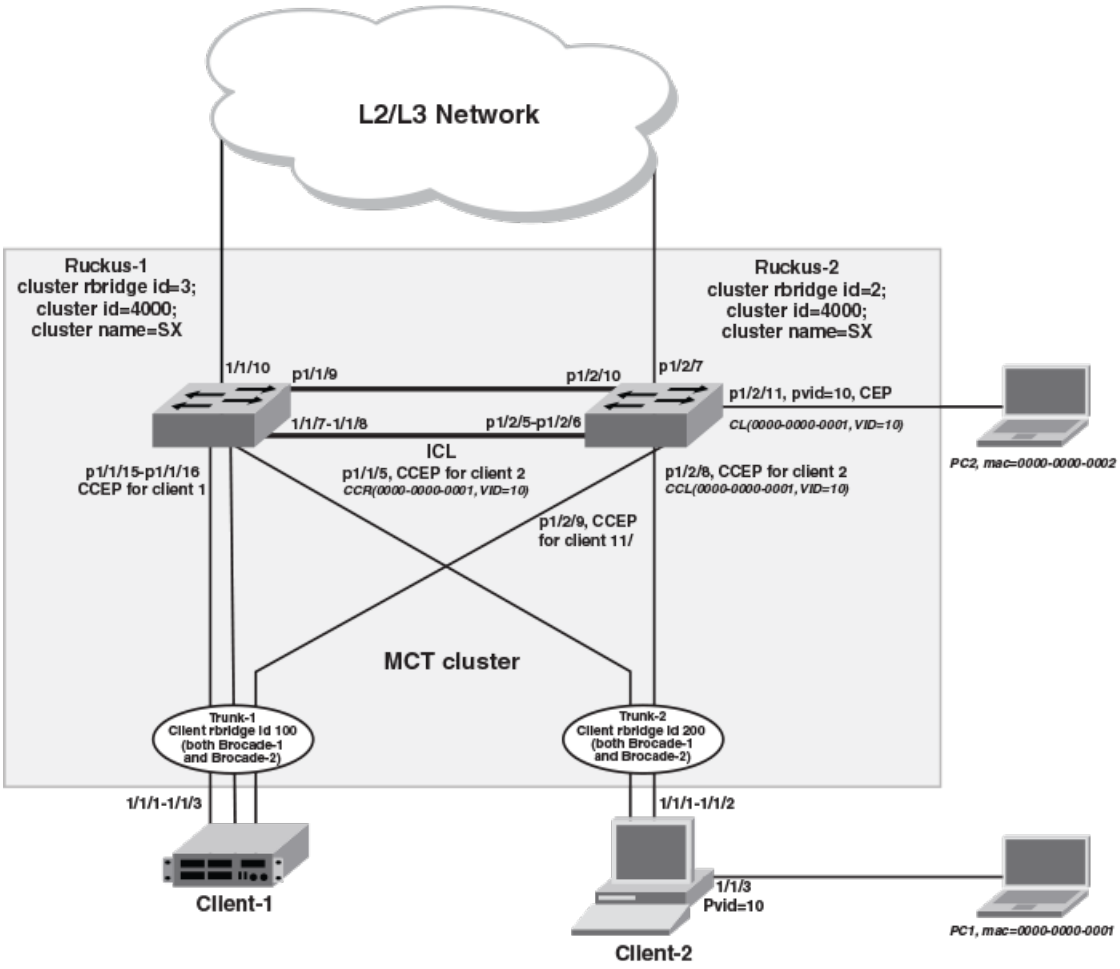
The following FastIron features are not supported with MCT:

- ACLs on VLAN session (ICL) ports.
- LACP on ICL.
- MSTP, VSRP, and RIP.
- MSDP, Anycast RP, and embedded RP.
- IPv6, VRRP-E (IPv6), and VRRPv3.
- GRE on the ICL VE interfaces.
- DAI on the CCEPs.
- Egress ACLs on MCT 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.

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 32 Basic MCT configuration



MCT configuration considerations

- Configuring flow-based MAC address learning and MCT on the same device is not supported.
- When running STP, the STP state should be the same on both cluster devices. For additional information on running STP with MCT, refer to "STP/RSTP" under MCT Layer 2 protocols and to related configuration examples.
- Management VLAN configuration is not supported on MCT cluster devices running a switch image. MCT is not supported on switches with the management VLAN enabled.
- One ICL can be configured per device, and a device can be in only one cluster.
- The software version in both cluster devices must be exactly the same for the cluster to function.
- An ICL port should 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 ports static LAG.
- MAC learning is disabled on ICL ports for all VLANs.

- 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 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 ID are unique within an MCT configuration and cannot be confused with each other.
- The cluster ID should be the same on both cluster devices.
- The cluster RBridgeID should not conflict with any client RBridgeID or with the peer RBridgeID.
- The client RBridgeID is unique and should be the same on cluster devices.
- Ruckus recommends keeping only ICL ports in the session VLAN during operation.
- MCT can support up to 16 members per trunk group, depending on the software version and Switch type.
- An ICL interface cannot be configured as the CCEP in any client.
- BPDU guard and root guard configuration should be identical on both cluster devices.
- Because Egress PCL is configured on CCEPs, egress ACL cannot be configured on them. All types of ingress ACLs, DoS attack prevention, and so on can still be configured on those ports.
- 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.
- 48GC ports should not be used as MCT trunks or CCEP ports.

Differences in configuring MCT for the switch and router image

There are some differences in the MCT configuration for the switch image versus the router image:

- On a switch image, STP is by default enabled for all the VLANs; however, for MCT, Layer 2 protocols such as STP and RSTP should not be enabled on the session VLAN. Therefore, STP must be disabled explicitly for the session VLAN. STP is automatically disabled in the router image.
- Virtual Ethernet (VE) cannot be configured on a session VLAN in a switch image, but an IP address is needed for the cluster devices to communicate via CCP. Therefore, in a switch image, the configured management IP address is used to establish communication between the cluster devices.
- The management IP addresses in each of the cluster devices should be configured in the same subnet. If the IP addresses are in different subnets, ARP does not resolve the addresses, and MCT may not work. ARP for the peer cluster devices is always learned on the ICL port or trunk, so any management traffic between the two devices always goes through the ICL ports.

  NOTE
  CLI may vary somewhat among different Ruckus ICX platforms. These variations are not documented in the configuration examples provided in this chapter.

Configuring MCT

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

Step 1: Configure ICL and LAGs for client devices on page 96

Step 2: Configure the MCT VLAN, MCT session VLAN, and recommended MCT keep-alive VLAN on page 96

Step 3: Configure the cluster on page 97

Step 4: Configure clients on page 97

After completing these steps, you can verify the configuration by running the **show cluster** command. Refer to Displaying MCT information on page 129.

Step 1: Configure ICL and LAGs for client devices

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

NOTE
ICL LAGs support only static trunks.

To configure an ICL static LAG, enter the following commands.

```
device-1(config)# lag MCT_lag1 static id 2
Ruckus-1(config-lag-MCT_lag1)# ports ethernet 1/1/7 to 1/1/8

device-1(config)# lag MCT_lag1 static id 2
Ruckus-2(config-lag-MCT_lag1)# ports ethernet 1/2/5 to 1/2/6
```

To configure a dynamic LAG for a client device, enter the following commands for each MCT cluster device.

```
device-1(config)# lag client_lag2 dynamic id 5
Ruckus-1(config-lag-client_lag2)# ports ethernet 1/2/11
```

Step 2: Configure the MCT VLAN, MCT session VLAN, and recommended MCT keep-alive VLAN

To create the MCT session VLAN and recommended MCT keep-alive VLAN for Ruckus-1 in the topology of Basic MCT configuration on page 93, enter the following commands.

```
device-1(config)# vlan 3001 name MCT-keep-alive
device-1(config-vlan-3001)# tagged ethernet 1/1/9
device-1(config-vlan-3001)# exit
device-1(config)# vlan 3000 name Session-VLAN
device-1(config-vlan-3000)# tagged lag 2
device-1(config-vlan-3000)# no spanning-tree
```

For routers, add the following commands.

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

For switches, add the following commands.

```
device-1(config)# ip address 10.1.1.3/24
```

To create a session VLAN and keep-alive VLAN for device-2, enter the following commands.

```
device-2(config)# vlan 3001 name MCT-keep-alive
device-2(config-vlan-3001)# tagged ethernet 1/2/10
device-2(config-vlan-3001)# exit
device-2(config)# vlan 3000 name Session-VLAN
device-2(config-vlan-3000)# tagged lag 2
device-2(config-vlan-3000)# no spanning-tree
```

For routers, add the following commands.

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

For switches, add the following commands.

```
device-2(config)# ip address 10.1.1.2/24
```

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

```
device-1(config)# vlan 1000 name MCT-VLAN-example
device-1(config-vlan-1000)# tagged ether 1/1/15 to 1/1/16 lag 2
```

Step 3: Configure the cluster

Cluster local configuration uses the cluster ID and RBridge ID for the local switch or router.

Syntax: [no] cluster [*cluster-name*] *cluster-id*

Syntax: [no] rbridge-id *id*

Configuration of the peer device involves the peer's IP address, RBridge ID, and ICL specification. The *cluster-name* variable is optional; the device auto-generates the cluster name as CLUSTER-X when only the cluster ID is specified. The *cluster-id* variable must be the same on both cluster devices.

Syntax: [no] peer *peer-ip* **rbridge-id** *peer-rbridge* **icl** *map-icl*

The RBridge ID must be different from the cluster RBridge and any other client in the cluster. The MCT member VLAN is defined as any VLAN of which the ICL is a member.

To configure Ruckus-1 for the cluster in the topology of Basic MCT configuration on page 93, enter the following commands.

```
device-1(config)#cluster ICX 4000
device-1(config-cluster-ICX)# rbridge-id 3
device-1(config-cluster-ICX)# session-vlan 3000
device-1(config-cluster-ICX)# keep-alive-vlan 3001
device-1(config-cluster-ICX)# icl SX-MCT lag 2
device-1(config-cluster-ICX)# peer 10.1.1.2 rbridge-id 2 icl SX-MCT
device-1(config-cluster-ICX)# deploy
```

To configure Ruckus-2 for the cluster in the topology of Basic MCT configuration on page 93, enter the following commands.

```
device-2(config)# cluster ICX 4000
device-2(config-cluster-ICX)# rbridge-id 2
device-2(config-cluster-ICX)# session-vlan 3000
device-2(config-cluster-ICX)# keep-alive-vlan 3001
device-2(config-cluster-ICX)# icl SX-MCT lag 2
device-2(config-cluster-ICX)# peer 10.1.1.3 rbridge-id 3 icl SX-MCT
device-2(config-cluster-ICX)# deploy
```

Step 4: Configure clients

This section describes how to configure clients manually. For instructions on automatic client configuration, refer to Setting up cluster client automatic configuration on page 100.

Client configuration requires the client name, RBridge ID, and CCEP. In the network shown in the Basic MCT configuration on page 93, 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.

The client name can be different on the different cluster devices. To configure the client name, enter the following command.

Syntax: [no] client *client-name*

The client RBridge ID must be identical on both of the cluster devices. To configure the client RBridge ID, use the following command.

Syntax: [no] rbridge-id *id*

To configure the physical port or static trunk as the client CCEP, use the following command.

Syntax: [no] client-interface { **ethernet** *unit/slot/port* | **lag** *lag-id* }

To configure Client-2 on Ruckus-1 in the topology of Basic MCT configuration on page 93, enter the following command.

```
device-1(config-cluster-ICX)# client client-2
device-1(config-cluster-ICX-client-1)# rbridge-id 200
device-1(config-cluster-ICX-client-1)# client-interface ether 1/1/5
device-1(config-cluster-ICX-client-1)# deploy
```

To configure Client-2 on Ruckus-2 in the topology of Basic MCT configuration on page 93, enter the following command.

```
device-2(config-cluster-ICX)# client client-2
device-2(config-cluster-ICX-client-2)# rbridge-id 200
device-2(config-cluster-ICX-client-2)# client-interface ether 1/2/8
device-2(config-cluster-ICX-client-2)# 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 an ICX 7750 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 an ICX 7750, the port on the ICX 7750 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 can be configured as the force-up port. If you configure multiple ports as force-up, this 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 begin to forward packets.
- If the port receives an LACPDU, it bundles with other ports and forms a link aggregation group. The server is operational.
- If the force-up port goes down while in a link aggregation group, 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 LACPDU, 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.

Syntax: [no] client-interface link-aggregation force-up ethernet *unit/slotnum/portnum*

The following example shows the link aggregation information for a port configured to 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:             120, default: 120
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 RBridgeID (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.
- LACP client interface auto-detection is supported only for devices running release 7.4 software and later on FastIron platforms.
- RBridgeID collision: When hash collisions occur, cluster client automatic configuration reports errors, and manual intervention is required.

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 switch should 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.

Refer to Setting up cluster client automatic configuration on page 100 for detailed instructions on the cluster client automatic configuration process.

Setting up cluster client automatic configuration

Complete the following steps to configure cluster client automatic configuration.

1. Enable the client auto-detect ports on both MCT devices.

   ```
   device-1(config-cluster-ICX)# client-auto-detect ethernet 1/1/15-1/1/16
   ```

   In the port list, specify all the CCEPs for all potential clients.
2. Start the client auto-detect process on both cluster devices.

   ```
   device-1(config-cluster-ICX)# 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.
3. Check and fix the automatically detected clients.

   ```
   device-1(config-cluster-SX)# show cluster cluster-SX client-auto-detect
   cluster cluster-SX 4000
    rbridge-id 3
    session-vlan 3000
    icl SX-MCT lag 2
   peer 10.1.1.2 rbridge-id 2 icl SX-MCT
   client-auto-config ethe 1/1/15 to 1/1/16 ethe 1/2/8 ethe 1/2/9 eth 1/2/10
    client-auto-config start
   deploy
    client AUTO-Router002438769e00
     rbridge-id 3593
     client-interface ethe 1/1/15
   !
   ```

   NOTE
   At this point, the client configuration does not appear in the running configuration and cannot be modified. Static trunk and LACP configuration are not effective yet.

4. Configure automatically detected clients into the running configuration.

```
device-1(config-cluster-ICX)# client-auto-detect config
```

   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 CCEPs is put into the disabled state. Ports that are successfully programmed as CCEP are removed from the autoconfig-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.

Other cluster client automatic configuration commands

You can use the following commands as an alternative to the step-by-step procedure in Cluster client automatic configuration on page 99.

Use the following command to enable or disable cluster client automatic configuration on a range of ports.

Syntax: [no] client-auto-detect Ethernet *x* [**to** y]

Use the following command as an alternative to **client-auto-detect config** . This command also configures automatically detected clients into the running configuration and deploys all of the automatically detected clients.

Syntax: client-auto-detect config deploy-all

Use the following command to start the cluster client automatic configuration. Within one minute of the time that each client is discovered, the client is automatically configured and deployed into the running configuration.

Make sure that the network connection and configuration are in place before using this command.

Syntax: client-auto-detect start [**config-deploy-all**]

Use the following command to stop the current running cluster client automatic configuration process. All auto-detected but unconfigured clients will be cleared.

Syntax: client-auto-detect stop

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.
- 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 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 master/slave negotiation per client. After negotiation, the slave shuts down its client ports, and the master client ports continue to forward the traffic.
  - The master/slave negotiation is performed per MCT client on the basis of RBridgeID and client Local or Remote accessibility. If the client is reachable from both MCT devices, the lower RBridgeID becomes the master. If the client can be accessed only from one of the MCT devices, the cluster device on which it is reachable becomes the master.
  - 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 should not be configured when **bpdu-flood-enable** is configured. Refer to MCT Layer 2 protocols on page 106.

- ICL interface or CCP goes down (keep-alive 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.

  NOTE
  The keep-alive link is supported on the ICX 7750 in an SSTP or MST environment, even though ICX 7750 has the BPDU flood-enable feature built in. This is because the BPDU flood-enable in the ICX 7750 is hardware enabled.

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.

To disable the fast-failover mode, enter a command such as the following.

```
device-1(config-cluster-SX)# peer 10.1.1.3 disable-fast-failover
```

Syntax: [no] peer *peer-ip* **disable-fast-failover**

Client isolation mode

NOTE
You must create the same isolation mode on both cluster devices. The CLI will allow modification of the client isolation mode on MCT cluster devices even when the cluster is deployed.

MCT cluster devices can operate in two modes. Both peer devices should be configured in the same mode.

Loose mode (default): When the CCP goes down, the peer device performs the master/slave negotiation. After negotiation, the slave shuts down its peer ports, but the master 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 masters, and both of the client ports stay up.

```
device-1(config-cluster-SX)# client-isolation loose
```

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.

```
device-1(config-cluster-SX)# client-isolation strict
```

Syntax: [no] client-isolation strict

Shutting down all client interfaces

Use the **client-interfaces shutdown** command when performing a hitless upgrade operation. This command can be used to shut down all the local client interfaces in the cluster, which results in failover of traffic to the peer device.

```
device-1(config-cluster-SX)# client-interfaces shutdown
```

Syntax: [no] 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**
> Keep-alive VLAN configuration is not allowed when the client isolation mode is strict. When a keep-alive VLAN is configured, client isolation mode cannot be configured as strict.

```
device-1(config-cluster-SX))# keep-alive-vlan 10
```

Syntax: [no] keep-alive-vlan *vlan-id*

The *vlan_id* variable specifies the VLAN range. Possible values are from 1 to 4089.

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, master/slave 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 slave.
  - Otherwise, the device with the higher RBridgeID becomes the slave.
- 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

To specify the keep-alive timers and hold time for the peer devices, enter a command such as the following.

```
device-1(config-cluster-SX))# peer 10.1.1.3 timers keep-alive 40 hold-time 120
```

Syntax: [no] peer *peer-ip* **timers keep-alive** *keep-alive-time* **hold-time** *hold-time*

The *peer-ip* parameter should be in the same subnet as the cluster management interface.

The *keep-alive-time* variable can be from 0 to 21845 seconds. The default is 10 seconds.

The *hold-time* variable can be from 3 to 65535 seconds and must be at least 3 times the keep-alive time. The default is 90 seconds.

NOTE
The keep-alive VLAN and keep-alive timers are not related. The keep-alive timer is used by CCP.

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 operations

This section describes configuration operations related to MAC addresses.

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 address.

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 RBridgeID wins and is installed in the FDB.

MAC addresses in MCT VLANs are updated across the cluster using 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 via 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 RBridgeID.

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 RBridgeIDs.

Cluster Client Remote MAC (CCR): MAC addresses that are learned via MDUP message from the peer device (CCL on the peer). The MAC addresses are always programmed on the corresponding CCEP port 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 RBridgeIDs.

Cluster Multi-Destination Local MAC (CML): A static MAC entry that is configured locally on the MCT VLAN. Any static MAC address configured on 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 2 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 co-exist, 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 CEP ports.

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 CCEP local port is down, the MAC addresses are flushed locally, and individual MAC deletion messages are sent to the peer device.

If the **clear mac** command is given, all the MDB and FDB are rebuilt.

If the **clear mac vlan** command is given, all the local MDB and FDB are rebuilt for the VLAN.

MAC movement happens normally on the local device.

CEP to CCEP MAC movement - MAC movement happens normally on the local device, and it deletes all the other MDBs from the peer 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 should 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 does not involve stacking, and 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 should be up at all times to prevent the cluster from going down. These links should not be shut down when a loop is detected in a network. Instead, other available ports (CCEPs) should 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.

- MRP— An ICL interface cannot be configured as an MRP secondary interface or vice versa because the ICL cannot be BLOCKING.

  MRP cannot be enabled on MCT CCEP port or vice versa
- STP/RSTP—Do not configure STP on MCT VLANs at MCT cluster devices. By default, the spanning tree is disabled in the MCT VLANs.

  If the network topology may create Layer 2 loops through external connections, STP may be enabled on switches outside the MCT cluster to prevent the Layer 2 loop. The MCT cluster devices then performs a pass-through forwarding of STP BPDUs received through its ports in the MCT VLAN.
  - In rare cases in which the network topology consists of Layer 2 loops outside the MCT cluster that require STP/RSTP to be enabled on MCT VLANs in the cluster, the CCEPs are always in the spanning tree disabled state. Refer to MCT configuration examples using STP on page 142 to view deployment scenarios where STP is used in an MCT configuration to prevent Layer 2 loops.
  - The STP/RSTP algorithms have been modified so that the ICL never enters blocking state. The ICL guard mechanism ensures that if the ICL is about to go into a blocking state, the port on which the superior BPDUs are being received is moved to blocking state, and the ICL guard timer starts running on it. This timer runs as long as superior BPDUs are received on this interface. As long as this timer runs on an interface, the superior BPDUs are dropped.

  - The new BLK_BY_ICL STP state indicates that superior BPDUs were received on this interface, which could have led to blocking of the ICL interface, with the result that the CL port guard mechanism has been triggered on this port.
  - In an 802.1s MSTP deployment, Ruckus recommends disabling spanning tree on MCT cluster devices at the global level. MSTP cannot be configured on individual cluster devices.
  - An MCT cluster can support up to 32 spanning tree instances.
- BPDU forwarding—If the network deploys single STP or IEEE 802.1s (MSTP), the MCT cluster devices must be configured using the **bpdu-flood-enable** command to flood the single STP/MSTP BPDUs in the SSTP/MSTP domain (that is, to forward to all of the ports in the cluster switch, irrespective of VLAN.)

  When **bpdu-flood-enable** is configured, only the ICL should connect the two MCT cluster devices. (The keep-alive VLAN link should not connect them.) If there is an additional link, the flooded BPDU will cause a loop and high CPU utilization.

  NOTE
  The **bpdu-flood-enable** command is not supported on the Ruckus ICX 7750.

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:

- MAC-forward entries (mcache entries on MCT VLAN)
- IGMP/MLD Join/Leave (control packets on MCT VLAN)
- PIM-SM/PIM6-SM Join/Prune (control packets on MCT VLAN)
- IGMP/MLD dynamic router ports on MCT VLAN

IGMP/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 commands show configuration commands for the VLAN level (IGMP and MLD), the global level (IGMP/MLD), and for PIM-SM and PIM6-SM.

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/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 a 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 a passive mode):

```
device(config)# vlan 100
device(config-vlan-100)# multicast6 passive
device(config-vlan-100)# multicast6 pimsm-snooping
```

IGMP/MLD snooping behavior on MCT cluster devices

- Local information is synchronized to the MCT peer device using CCP. The information includes Mcache/FDB entry (on arrival of data traffic), joins/leaves, dynamic router ports, and PIM-SM snooping joins/prunes.
- Native control packets (joins/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 outgoing interface (OIF) by default whenever the CCEP is a source or a receiver. This provides faster convergence during MCT failover.
- For IGMP/MLD joins/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 CCEP 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 101 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/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/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 MCT1 devices must run pimsm-snoop.

- PIM messages are forwarded via the hardware.
- PIM join/prune is synced to the peer cluster device using CCP.
- PIM prune is processed only if indicated by the peer cluster device.
- PIM join/prune received natively on ICL is ignored.
- PIM hello is not synced but is received natively on ICL.
- PIM port/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 12 and Table 13 list the forwarding entries for PIM-SM and PIM6-SM multicast snooping.

TABLE 12 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 13 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 via PIM join/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
```

Syntax: show ip pimsm-snooping cache

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, YES indicates that reports/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/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 synched 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)
```

Syntax: show ip multicast cluster { **group** | **mcache** | **vlan** *vlan-id* }

Use the **show ip multicast cluster pimsm-snooping** command to display detailed information about OIFs added via a pimsm-snoop module.

```
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)
```

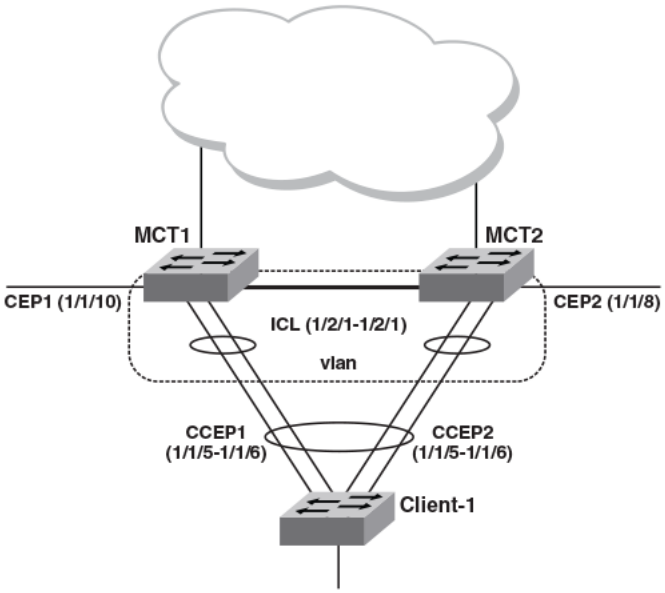
Syntax: show ip multicast cluster pimsm-snooping [**group** | **vlan**]

Refer to the *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 33 Multicast snooping over MCT



The following example shows the configuration for multicast snooping for the MCT1 cluster device in the previous figure.

```
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
router-interface ve 3000
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 ICX7750 3000
rbridge-id 2
session-vlan 3000
keep-alive-vlan 3001
icl SX-MCT ethernet 1/2/1
peer 10.1.1.3 rbridge-id 3 icl ICX7750-MCT
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 33.

```
!
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
router-interface ve 3000
vlan 3001 name keep-alive-vlan
tagged eth 1/2/2
interface ve 3000
ip address 10.1.1.3 255.255.255.0
!
cluster SX 3000
rbridge-id 3
session-vlan 3000
keep-alive-vlan 3001
icl SX-MCT ethernet 1/2/1
peer 10.1.1.2 rbridge-id 2 icl SX-MCT
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 33.

```
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
router-interface ve 3000
vlan 3001 name keep-alive-vlan
tagged eth 1/2/2
ip multicast active
interface ve 3000
ip address 10.1.1.2 255.255.255.0
!
cluster SX 3000
rbridge-id 2
session-vlan 3000
keep-alive-vlan 3001
icl SX-MCT ethernet 1/2/1
peer 10.1.1.3 rbridge-id 3 icl SX-MCT
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 33.

```
!
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
router-interface ve 3000
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 SX 3000
rbridge-id 3
session-vlan 3000
keep-alive-vlan 3001
icl SX-MCT ethernet 1/2/1
```

```
peer 10.1.1.2 rbridge-id 2 icl SX-MCT
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.

TABLE 14 Layer 3 Feature Support with MCT

Feature	Sub-feature	Session VLAN VE	Member VLAN VE	Design Philosophy
ip	access-group[a]	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	No	
	irdp	No	Yes	
	local-proxy-arp	No	Yes	
	metric	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	
	tunnel	No	Yes	
	use-acl-on-arp	Yes	Yes	
	vrrp	No	Yes	
	vrrp-extended	No	Yes	

TABLE 14 Layer 3 Feature Support with MCT (continued)

Feature	Sub-feature	Session VLAN VE	Member VLAN VE	Design Philosophy
ipv6		No	No	IPv6 is not supported for MCT management. IPv6 is not supported on member VLAN VE.

a.) *ICL: The ICL port is added as default whenever a CCEP is in OIF. The data traffic received from the ICL port is filtered out by a dynamically programmed egress filter on the CCEPs.

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. The following sections describe the details of Layer 3 behavior in an MCT environment.

ARP Resolution

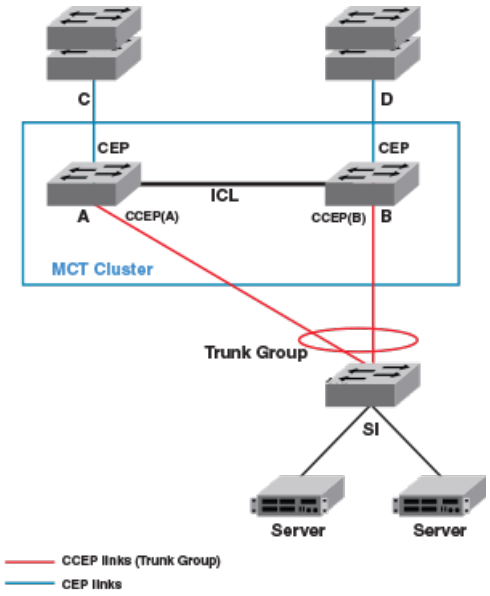
ARP resolution for the MCT client is required at the MCT cluster to forward traffic from a CEP to the CCEP. This ARP packet would normally be learned over the CCEP port. However, if the client's MAC address is not already known on the CCEP, its ARP could be temporarily learned over the ICL. When the MAC Database Update Protocol (MDUP) message from the cluster peer device moves the client MAC address from the ICL to the CCEP, ARP is also moved to the CCEP. During this transient time, no client ARP should 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. While the ICX 7750, as an MCT peer, can successfully forward this traffic to the client, SX 800 and SX 1600 devices will experience loss of traffic during this time.

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 learnt 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 34 Configuration for Layer 3 unicast



Layer 3 traffic forwarding towards MCT clients

Traffic destined to the MCT clients follows normal IP routing. By default, the best route should not involve the ICL link. Only when the local CCEP is down is traffic rerouted to pass over the ICL.

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 should 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 traffic in the worst case) can pass through the ICL. This fact should be considered when designing the ICL capacity in the network.

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 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 Layer 2 switch all packets destined to VRRP/ VRRP-E virtual MAC address to the VRRP/VRRP-E master router for routing. The VRRP/VRRP-E backup learns the virtual MAC address while processing the VRRP hello message from the VRRP master. Both data traffic and VRRP/VRRP-E control traffic travel through the ICL unless the short-path forwarding feature is enabled (VRRP-E only).

VRRP/VRRP-E and VRRP-E2 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 fifty percent of CCEP to CEP traffic (and as much as a hundred 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/VRRP-E master and the peer device is the VRRP/VRRP-E backup, the following behavior is observed:

- Frames sent to the VRRP/VRRP-E virtual MAC address are Layer 2-switched to the VRRP/VRRP-E master device for routing. The VRRP-E MAC address is learned by the other MCT device that acts as 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/VRRP-E virtual MAC address are Layer 2 forwarded to the VRRP/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/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.

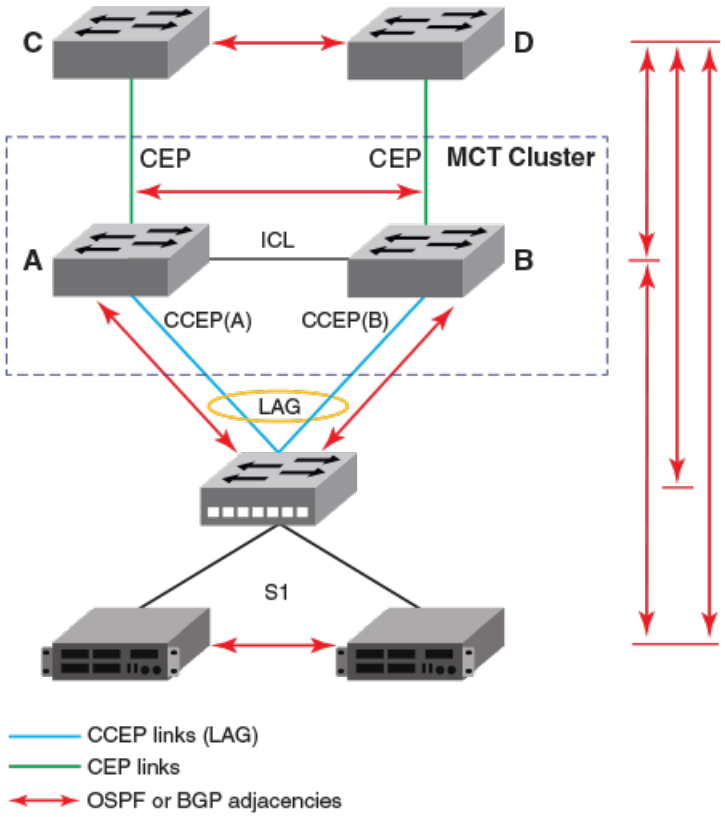
The combinations that can be established are:

- Devices connected to MCT cluster over CEP ports
- Devices connected to 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 L3 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 since each MCT cluster device forms an independent adjacency. When one of the MCT devices goes down, a layer 3 re-convergence is required and traffic loss is expected during this time.

FIGURE 35 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.

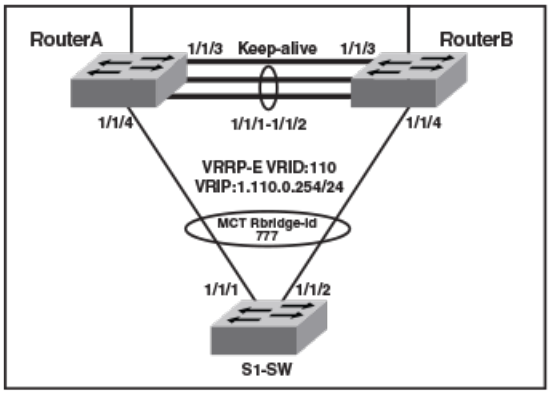
- Not all layer 3 features on MCT management interface are supported. If a VLAN is already configured with these Layer 3 features, it cannot be made the session VLAN. To see the list of unsupported features on MCT management interface, refer to Layer 3 behavior with MCT on page 113.
- IPv6 configurations are not supported on VEs of session or member VLANs.
- Route-only ports cannot be used as CCEP 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 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 should not be used for BGP peers on neighboring devices.
- IP addresses on the MCT management interface should 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 and on the VRRP backup, so as to disable short path forwarding when it loses its relevance
- VRRP or VRRP-E shouldn't 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. The associated configuration follows.

FIGURE 36 Sample Configuration for a single-level MCT



Router A - MCT configuration

This example presents the MCT configuration for Router A 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
 router-interface ve 110
!
vlan 1000 name ICL-Session-vlan by port
 tagged lag 55
 router-interface ve 1000
!
vlan 1001 name MCT-Keep-Alive by port
 tagged ethe 1/1/3
!
interface ve 1000
 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
!
```

Router B- MCT configuration

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
 router-interface ve 110
!
vlan 1000 name ICL-Session-vlan by port
 tagged lag 55
 router-interface ve 1000
!
vlan 1001 name MCT-Keep-Alive by port
 tagged ethernet 1/1/3
!
interface ve 1000
 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
!
```

S1-SW configuration

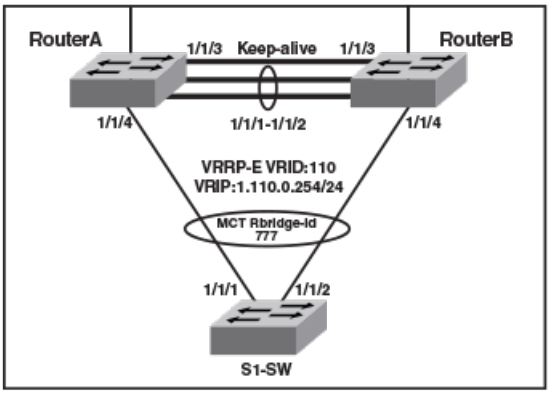
This example presents the configuration for the S1-SW device.

```
!
lag lag_s1_sw static id 60
 ports ethe 1/1/1 to 1/1/2
!
vlan 110 by port
tagged lag 60
 router-interface ve 110
!
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 37 Sample MCT configuration with VRRP-E



Router A - VRRP-E configuration

This example presents the VRRP-E configuration for the Router A 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
!
```

Router B - VRRP-E configuration

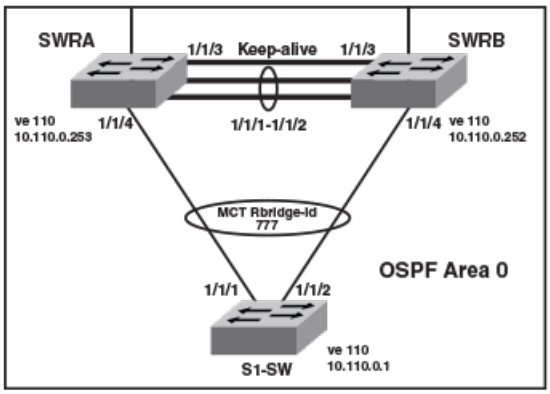
This example presents the VRRP-E configuration for the RouterB cluster device.

```
!
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 38 MCT Configuration with OSPF



SWRA - OSPF configuration

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
!
```

SWRB - OSPF configuration

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
!
```

S1-SW configuration

This example presents the configuration for the S1-SW device.

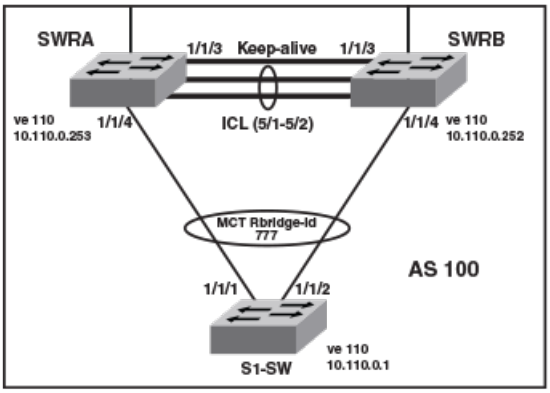
```
!
lag lag_s1_sw static id 60
ports ethernet 1/1/1 to 1/1/2

!
vlan 110 by port
tagged lag 60
router-interface ve 110
!
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 39 MCT Configuration with BGP



SWRA - BGP configuration

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
!
```

SWRB - BGP configuration

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
!
```

S1-SW configuration

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 ethernet lag 60
router-interface ve 110
!
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 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 ICL. The nature of the MCT LAG requires this. Packets from the MCT client on the CCEP ports are received by only one of the MCT peers. Hence the control packets that are received natively on the CCEP ports are sent over ICL to synchronize the states. The Join or Prune and Asserts are synchronized to maintain the Outgoing Interface (OIF) state for the CCEP ports 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 MCT VLAN. Exceptions are Assert packets and Join packets triggered only for ICL OIFs.
- Control packets received on any port of MCT VLAN are flooded on MCT VLAN.
- Control packets received on ICL are flooded in a controlled manner on 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 of the packet in the MAC table to determine the packet ingress port as follows:

- If the source MAC is learned on CCEP port, the packet ingress port will be a CCEP port.
- If it is not, the packet ingress port will be an ICL port.

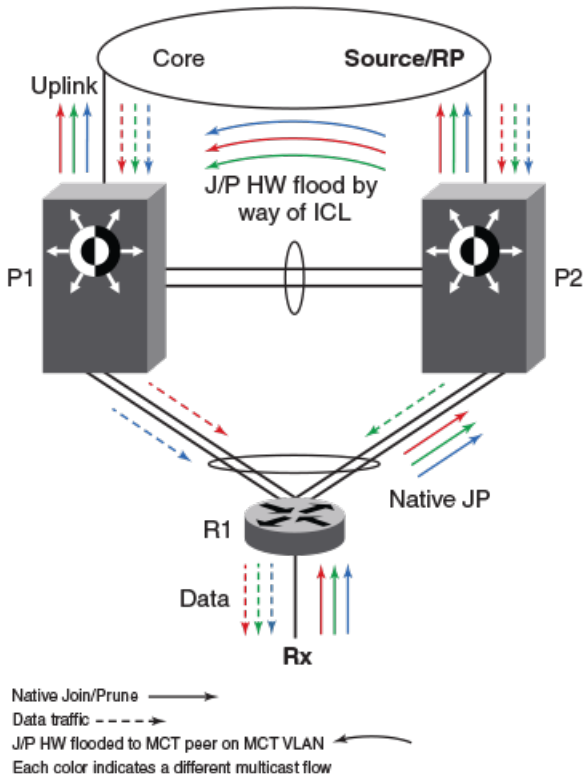
In the following figures, P1 and P2 are MCT peers and R1 is the MCT client. P1, P2 and R1 are configured with PIM on the MCT VE interface. MCT peers act as PIM intermediate routers with respect to R1.

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. RP and source is in the core and the connectivity to the core is via an uplink.

FIGURE 40 MCT peer as immediate 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. That means 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 could reach either P1 or P2 due to the above nature of MCT LAG.
- Hellos that reach P2 are sent to P1 natively over ICL. That means P1 learns about R1 (by searching the source-MAC 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 neighbor state for R1.
- Hellos originated from P1 and P2 are flooded on the MCT VLAN i.e. on ICL, CEP, local CCEP ports. 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 neighbor state, for each other.

Join or prune exchange and mcache state:

- As receivers are connected to R1, R1 creates *,G state and sends a join state towards RP and sends it on the MCT LAG. This join, like any other packet, is received by only one of the MCT peers.
- Suppose P2 receives the *,G join natively. This join is processed or consumed and is also flooded to P1 over ICL.
- P1 processes the join received over ICL as if it is received on CCEP.
- Both P1 and P2 create *,G state with CCEP as OIF.
- Both the peers send the *,G join towards RP and both the peers pull the traffic.

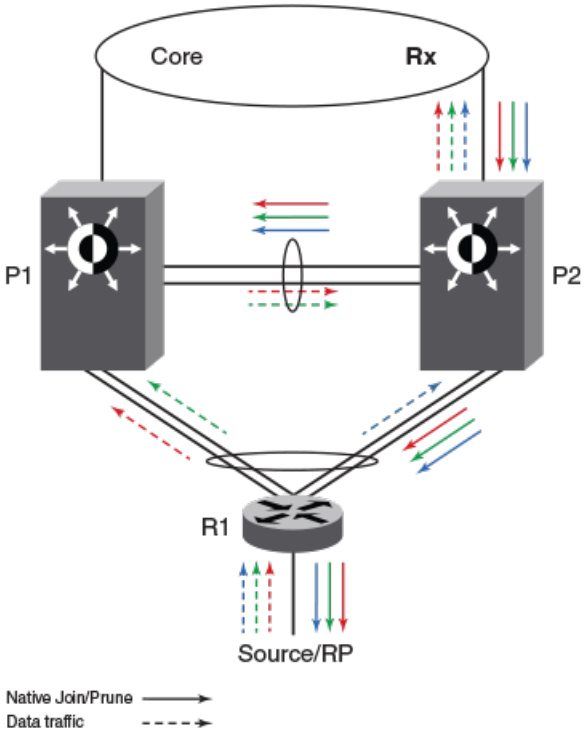
- When the traffic arrives, the S,G state is created on both the peers but only one of them forwards based on the software hashing algorithm.

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. RP and source are beyond R1.

FIGURE 41 MCT peer as immediate downstream router



Hello exchange and neighbor state:

It 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 *,G state with uplink as OIF by consuming the join state.
- P2 due to its *,G state originates a join towards RP. This join is flooded on the MCT VLAN and R1 creates *,G state.
- P1 on receiving this join natively via ICL creates *,G state and adds ICL as OIF. Note that as a special case P1 will not include the *,G in the join it generates towards RP as in this case the IIF is 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 RP and pulls the traffic. Because the OIF at R1 is LAG, traffic pulled by R1 will be load-shared among the member links.

- Thus traffic for S,G will reach only one of the MCT peers. Assuming the traffic reaches P2, (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 via ICL to P2 and P2 will forward it to its OIFs 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. That means 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 the feature works best when both CCEP links have the same capacity and the source and group addresses are evenly distributed. That situation avoids the timing synchronization between the MCT peer routers, which would be very hard to achieve.

The sharing is done at a stream, not packet, level ,using the following software hash algorithm:

((source address + group address) & 0x00000001) ^ ((local_bridge_id > remote_bridge_id))

If result is 1, local CCEP forwards the traffic; if result is 0, remote CCEP forwards the traffic

Fast convergence of multicast traffic

The multicast routing on MCT feature 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 state for, 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 tradeoff 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

Requirements for multicast MCT

OSPF must be supported on MCT member VLAN virtual Ethernet (VE) interfaces, that is, on CCEP, CEP, and ICL links.

Limitations

These are the limitations for MCT peers to support intermediate router functionality. These limitations are due to load-sharing and fast convergence trade-offs.

- PIM-DM is not supported.
- Few packets may be lost during convergence interval or forwarding duplication may happen.
- MCT client will do flow based load-sharing, not per packet load-sharing.
- Traffic loss or duplication will happen when Keep-Alive VLAN, Cluster Communication Protocol (CCP) ,or ICL between MCT peers are not up.

- Multicast routing configurations on 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 interval.
- Both the MCT peers maintain state and pull down traffic for all multicast flows from the core, whether the chassis is forwarding this stream to the local CCEP or not. This could potentially waste the bandwidth inside the core and on 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.
- PIM neighbor on CEP in an MCT VLAN is not supported if the MCT cluster is running PIM on the same MCT VLAN.
- First-hop routing (FHR) and "Last-hop routing (LHR) are not supported on MCT clusters on MCT VLAN interfaces.
- Rendezvous points (RP) are not supported on MCT clusters.
- MSDP and Anycast-RP are not supported on MCT clusters.
- This feature is not supported on non default VRFs.
- IPv6 multicast routing on MCT is not supported on MCT clusters.

Configuring multicast routing over MCT

Follow these steps to configure multicast routing over MCT.

1. Configure an MCT cluster.
2. Configure an MCT member VLAN.
3. Configure multicast routing (PIM) over MCT member VE.

This example shows the configuration of an MCT cluster, MCT member VLAN with router interface (VE), 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
 router-interface ve 10
 spanning-tree 802-1w
 spanning-tree 802-1w ethe 1/1/11 disable
!
!

interface ve 10
 ip address 10.10.10.100 255.255.255.0
 ip pim-sparse
 ip ospf area 0
```

This example shows the configuration of an MCT cluster, MCT member VLAN with router interface (VE), 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 2/1/1
 peer 5.5.5.10 rbridge-id 1000 icl MCT
 deploy
 client client-100
  rbridge-id 100
  client-interface ethernet 2/1/11
  deploy
!
!
!
!
!
!
end


vlan 10 name member-vlan by port
tagged ethe 2/1/1 ethe 2/1/11 ethe 2/1/27
router-interface ve 10
spanning-tree 802-1w
spanning-tree 802-1w ethe 2/1/11 disable
!


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

Displaying peer and client states

Use the **show cluster config** command to display 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
peer 172.17.0.2 rbridge-id 101 icl SXR122-MCT
deploy
client KL134
rbridge-id 14
client-interface ethernet 1/23
deploy
```

```
client AGG131
rbridge-id 10
client-interface ethernet 12/2
deploy
client FOX135
rbridge-id 15
client-interface ethernet 1/25
deploy
```

Displaying state machine information

Use the **show cluster client** command to display 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 15 Reasons for Local CCEP down

Reason for Local CCEP down	Meaning
client-interfaces shutdown	Command is configured.
client-isolation strict	Command is configured.
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.

Displaying cluster, peer, and client states

Use the **show cluster ccp peer** command to display 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
```

Displaying information about Ethernet interfaces

Use the **show interface ethernet** command to display 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
```

Syntax: show interface ethernet *x/y*

Displaying STP information

Use the **show span** command to display STP information for an entire device.

```
device# show span
...
STP instance owned by VLAN 90

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
  90 8000748ef8f9739d 0    Root   8000 20  2   1   15  259968  1   748ef8f9739d

Port STP Parameters:

Port   Prio Path  State       Fwd    Design  Designated       Designated
Num    rity Cost              Trans  Cost    Root             Bridge
       Hex
1/2/1  80   1     FORWARDING  1      0       8000748ef8f9739d 8000748ef8f9739d
1/3/5  80   0     DISABLED    0      0       0000000000000000 0000000000000000
1/3/6  80   0     DISABLED    0      0       0000000000000000 0000000000000000
```

MAC show commands

To display all local MAC address entries for a cluster, use the **show mac cluster** command.

```
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

To clear all MAC addresses in the system, enter the following command.

```
device# clear mac
```

Syntax: clear mac

NOTE
On SXL with authentication protocols and high traffic, the clear mac and mac flush operations log a lot of new address messages, which results in high CPU utilization for a few minutes.

Clearing cluster-specific MAC addresses

To clear cluster-specific MAC addresses in the system, enter a command such as the following.

```
device# clear mac cluster AGG-1 local
```

Syntax: clear mac cluster { *cluster-id* |*cluster-name* } { **local** | **remote** }

Clearing client-specific MAC addresses

To clear client-specific MAC addresses in the system, enter a command such as the following.

```
device# clear mac cluster AGG-1 client 1 local
```

Syntax: clear mac cluster { *cluster-id* | *cluster-name* } **client** *client-name* { **local** | **remote** }

Clearing VLAN-specific MAC addresses

To clear VLAN-specific MAC addresses in the system, enter a command such as the following.

```
device# clear mac vlan 2
```

Syntax: clear mac vlan *vlan_id*

Clearing MCT VLAN-specific MAC addresses

To clear MCT VLAN-specific MAC addresses in the system, enter a command such as the following.

```
device# clear mac cluster AGG-1 vlan 1 local
```

Syntax: clear mac cluster { *cluster_id* | *cluster-name* } **vlan** *vlan_id* { **local** | **remote** }

Clearing cluster client vlan-specific MACs

To clear cluster client-specific MAC addresses in the system, enter a command such as the following.

```
device# clear mac cluster AGG-1 vlan 2 client 1 local
```

Syntax: clear mac cluster {*cluster_id* | *cluster-name* } **vlan** *vlan_id* **client** *client_name* { **local** | **remote** }

Displaying MDUP packet statistics

To display the statistics of MDUP packets, enter a command such as the following.

```
device#show mac 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
```

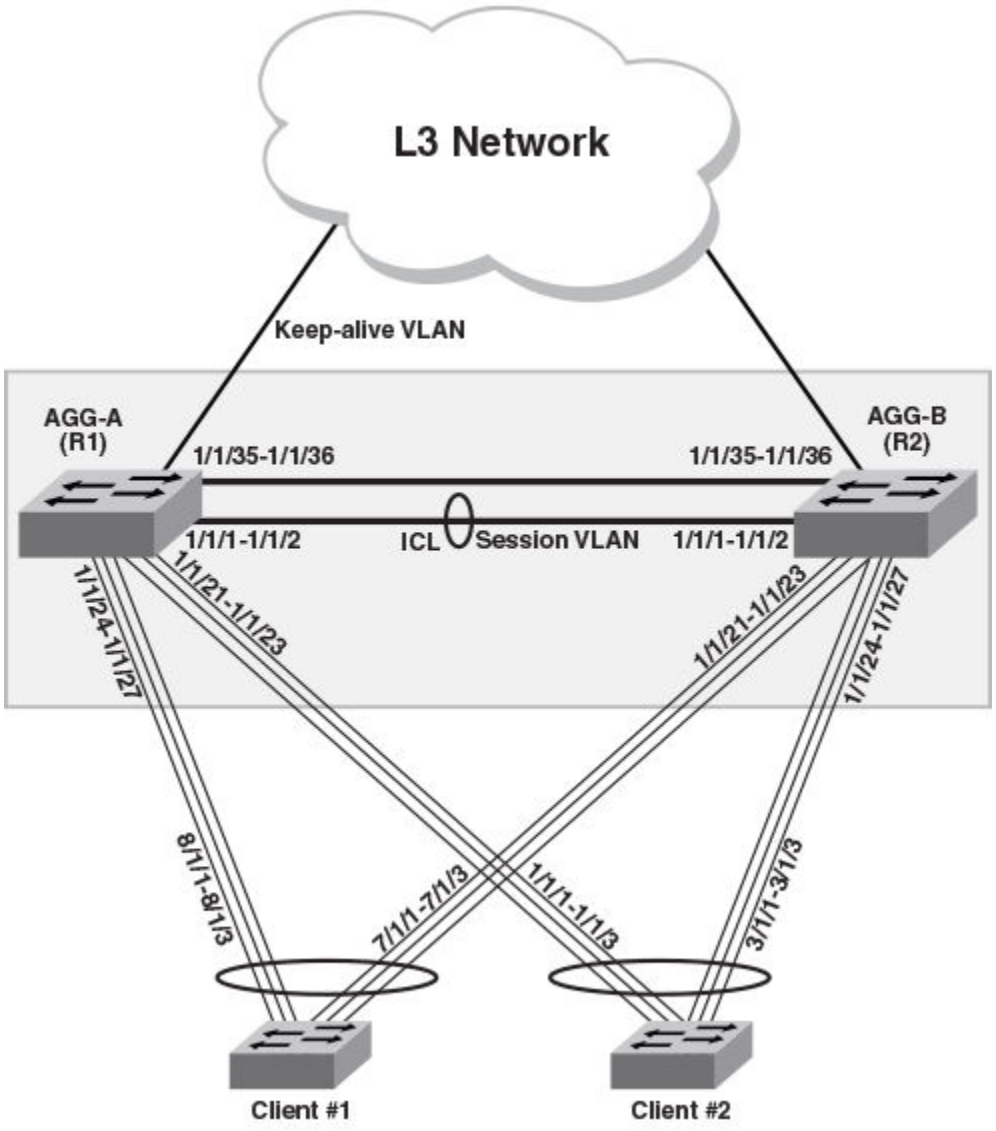
Syntax: show mac mdup-stats

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 be matching on the two ends of a LAG.

FIGURE 42 Single level MCT configuration



Client 1 - Configuration

If client 1 is a Ruckus ICX switch in Figure 42 on page 135, you can configure it as follows:

```
!
lag lag_client1_1 dynamic id 100
ports ethe 7/1/1 to 7/1/3 ethe 8/1/1 to 8/1/3

!
vlan 1905 by port
tagged lag 100
spanning-tree

!
```

Client 2- Configuration

If client 2 is a Ruckus ICX switch in Figure 42 on page 135, you can configure it as follows:

```
!

lag lag_client2_1 dynamic id 200
 ports ethe 1/1/1 to 1/1/3 ethe 3/1/1 to 3/1/3

!
vlan 1905 by port
 tagged lag 200
 spanning-tree


!
```

AGG-A (R1) - Configuration

This section presents the configuration for the AGG-A (R1) cluster device in Figure 42 on page 135.

```
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
 primary-port 1/1/21
 deploy
!
vlan 2 name session-vlan by port
 tagged lag 103
 router-interface ve 2
!
vlan 3 name keep-alive-vlan by port
 tagged ethe 1/1/35-1/1/36
 router-interface ve 3
!
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

This section presents the configuration for the AGG-B (R2) cluster device in Figure 42 on page 135.

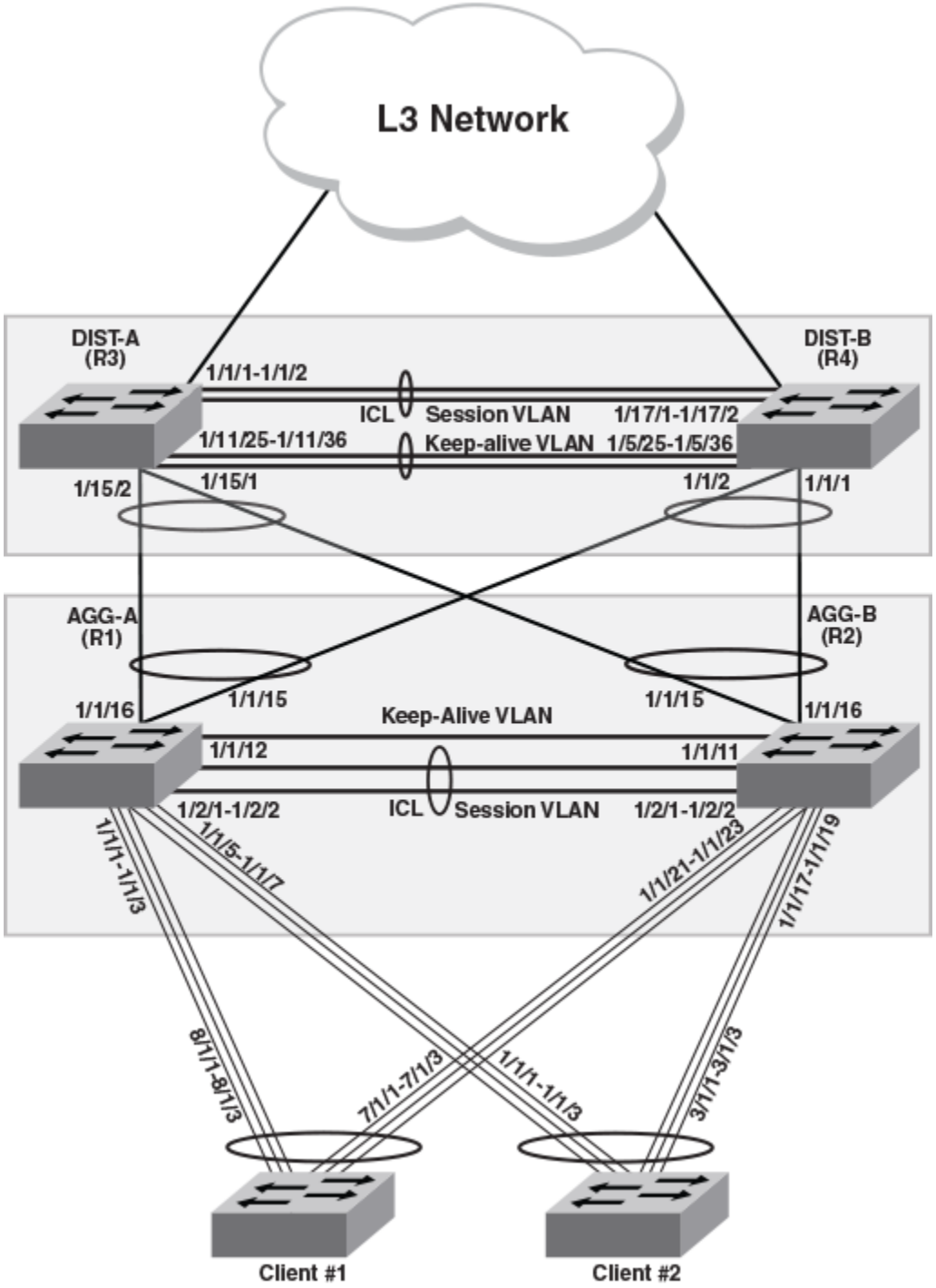
```
lag lag_agg_b_1 static id 103
 ports ethe 1/1/1 to 1/1/2

!
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
 router-interface ve 2
!
vlan 3 by port
 tagged ethe 1/1/35-1/1/36
 router-interface ve 3
!
!
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 be matching on the two ends of a LAG.

FIGURE 43 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's system ID.

The client configuration is the same as in the single-level example (refer to Single-level MCT configuration example on page 134).

AGG-A (R1) - Configuration

This example presents the configuration for the AGG-A (R1) cluster device in Figure 43 on page 138.

```
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
 router-interface ve 2
!
vlan 3 name keep-alive-vlan by port
 tagged ethe 1/1/12
 router-interface ve 3
!
!
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

This example presents the configuration for the AGG-B (R2) cluster device in Figure 43 on page 138.

```
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
 router-interface ve 2
!
vlan 3 name keep-alive-vlan by port
 tagged ethe 1/1/11
 router-interface ve 3
!
!
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

This example presents the configuration for the DIST-A (R3) cluster device in Figure 43 on page 138.

```
!
lag lag_dist_a_1 static id 15
 ports ethe 1/1/1 to 1/1/2
```

```
lag lag_dist_a_2 dynamic id 16
 ports ethe 1/15/1 to 1/15/2

!
lag keep-alive static id 200
ports ether 1/11/25 to 1/11/36

!
vlan 5 name session-vlan by port
 tagged lag 15
 router-interface ve 5
!
vlan 6 name keep-alive-vlan by port
 tagged ethe 1/11/25 to 1/11/36
 router-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

This example presents the configuration for the DIST-B (R4) cluster device in Figure 43 on page 138.

```
lag lag_dist_b_1 static id 40
 ports ethe 1/17/1 to 1/17/2

!
lag dist_b_2 dynamic id 41
ports ethe 1/1/1 to 1/1/2

!
lag keep-alive static id 201
ports ethe 1/5/25 to 1/5/36

!
vlan 5 name session-vlan by port
 tagged lag 40
 router-interface ve 5
!
vlan 6 name keep-alive-vlan by port
 tagged ethe 1/5/25 to 1/5/36
 router-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 examples using STP

Although MCT is considered an alternative to Spanning Tree, Spanning Tree protocols can be enabled in an MCT configuration as an added protection for any Layer 2 loops. The following use-case scenarios demonstrate the use of Spanning Tree protocols in an MCT configuration:

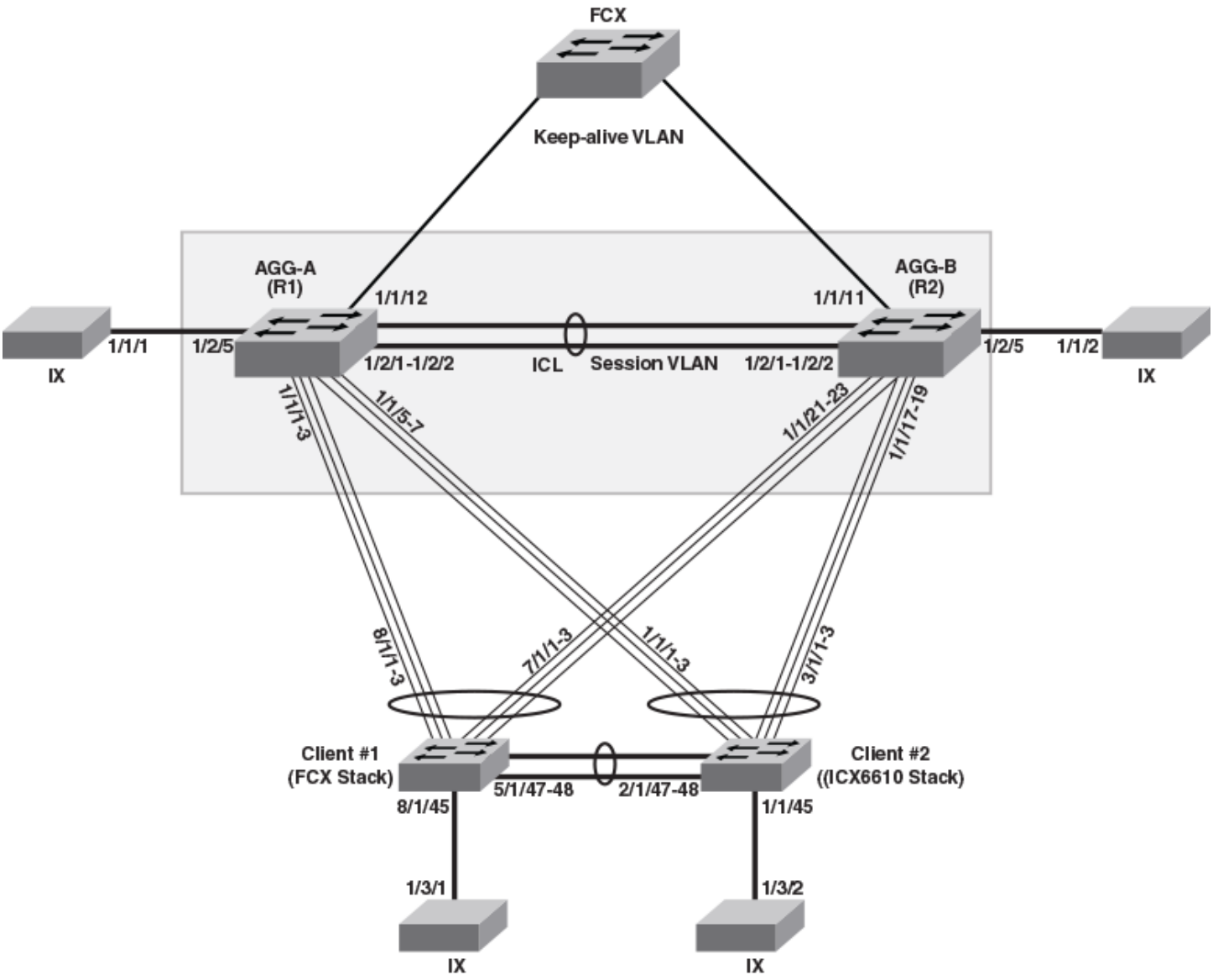
- Example 1: Configure the Per-VLAN Spanning Tree on the MCT Clients on page 145
- Example 2: Configure Single Spanning Tree (SSTP) on the MCT Clients on page 145
- Example 3: Configure Multiple Spanning Tree (MSTP) on the MCT Clients on page 146

The examples show STP elements enabled on MCT clients. It is recommended that you enable STP only on MCT clients, NOT on MCT cluster devices.

The following figure shows the base configuration of the MCT cluster, MCT clients, and the associated link-aggregation. The scenarios are depicted based on this network topology.

NOTE
The LAG IDs are locally significant only and need not be matching on the two ends of a LAG.

FIGURE 44 Sample network topology - Using STP in an MCT configuration



Router-1 configuration

```
!
lag "1" static id 1
 ports ethernet 1/1/1 ethernet 1/1/3


lag "1" static id 2
 ports ethernet 1/1/5 ethernet 1/1/7


lag "1" static id 3
 ports ethernet 1/2/1 ethernet 1/2/2

!
vlan 2 name session-vlan by port
```

```
tagged lag 3
router-interface ve 2
!
vlan 3 name keep-alive-vlan by port
tagged ethe 1/1/12
router-interface ve 3
!
vlan 1905 name MAC-scaling-vlan by port
tagged ethe 1/1/15 to 1/1/16 eth 1/3/1 1/3/3 lag 1 to 3 ethe 1/1/15 to
1/1/16 ethe 1/2/1 to 1/2/2 ethe 1/2/4 ethe 1/3/1 ethe 1/3/3
!
hostname R1
!
link-keepalive ethe 1/1/1 to 1/1/3 ethe 1/1/5 to 1/1/7
hitless-failover enable
!
interface ve 2
ip address 21.1.1.1 255.255.255.0
!
interface ve 3
```

AGG-B (R2) - Configuration

```
!
lag "1" static id 1
 ports ethernet 1/1/17 ethernet 1/1/19


lag "1" static id 2
 ports ethernet 1/1/21 ethernet 1/1/23


lag "1" static id 3
 ports ethernet 1/2/1 ethernet 1/2/3

!
vlan 2 name session-vlan by port
tagged lag 3
router-interface ve 2
!
vlan 3 by port
tagged ethe 1/1/11
router-interface ve 3
!
vlan 1905 name MAC-scaling-vlan by port
tagged ethe 1/1/15 to 1/1/19 ethe 1/1/21
```

Client-1 - Configuration

```
!
lag lag1 dynamic id 1
 ports ethernet 5/1/47 to 5/1/48
!
vlan 1905 by port
 tagged ethe 7/1/1 to 7/1/3 ethe 8/1/1 to 8/1/3 ethe 8/1/45 lag 1
!
link-keepalive ethe 7/1/1 to 7/1/3 ethe 8/1/1 to 8/1/3

!
lldp run
end
```

Client-2 - Configuration

```
!
lag lag1 dynamic id 1
ports ethernet 2/1/47 to 2/1/48

vlan 1905 name MAC-scaling-vlan by port
tagged ethe 1/1/1 to 1/1/3 ethe 1/1/45 ethe 3/1/1 to 3/1/3 lag 1
!
link-keepalive ethe 1/1/1 to 1/1/3 ethe 3/1/1 to 3/1/3

!
lldp run
end
```

Example 1: Configure the Per-VLAN Spanning Tree on the MCT Clients

External connections between clients other than the links in an MCT cluster can cause Layer 2 loops. Use Spanning Tree on the MCT clients so that the MCT cluster forwards Spanning Tree Bridge Protocol Data Units (BPDU) as if the cluster were in a pass-through mode.

Configure per-VLAN Spanning Tree on the two MCT VLANS 1901 and 1905 to have Rapid Spanning Tree (RSTP/802.1w). This example is based on the network topology shown in Figure 44 on page 143.

Client-1 Configuration

```
Client-1(config)# vlan 1901 1905
Client-1(config-mvlan-1901*1905)# spanning-tree 802-1w
Client-1(config-mvlan-1901*1905)# end
```

Client-2 Configuration

```
Client-2(config)# vlan 1901
Client-2(config-vlan-1901)# spanning-tree 802-1w
Client-2(config-vlan-1901)# spanning-tree 802-1w priority 4095
Client-2(config-vlan-1901)# vlan 1905
Client-2(config-vlan-1901)# spanning-tree 802-1w
Client-2(config-vlan-1901)# end
```

The MCT cluster switches do not have the spanning tree configured, but the BPDUs are passed through, and the Spanning Tree on the clients converges.

Use the **show 802-1w vlan** command to display the RSTP information for the specified port-based VLAN.

Example 2: Configure Single Spanning Tree (SSTP) on the MCT Clients

In a network where MCT clients have Single 802.1d or Single 802-1w elements of the Spanning Tree protocol enabled, configure the central processing unit (CPU) of the MCT peer switches so that it performs BPDU forwarding to avoid Layer 2 loops.

Using SSTP on MCT clients allows you to run a separate spanning tree on each port-based VLAN, which you can enable or disable on an individual basis. As an alternative, you can run a single spanning tree across all ports and VLANs on the device.

Enabling BDPU flooding can increase the CPU usage. When BPDU flooding is enabled, do not create redundant links between the MCT cluster devices or cascade multiple MCT clusters.

To enable the CPU to perform BPDU forwarding, use the bpdu-flood-enable command. This example is based on the network topology shown in Figure 44 on page 143.

Router-1 configuration

```
Router-1(config)# bpdu-flood-enable
Warning - Any recieved untagged BPDUs will now be flooded to all the ports.
```

Router-2 configuration

```
Router-2(config)# bpdu-flood-enable
Warning - Any recieved untagged BPDUs will now be flooded to all the ports.
```

Client-1 configuration

```
Client-1(config)# spanning-tree single 802-1w
Client-1(config)# show 802-1w vlan 1905
Single spanning tree is enabled. use "show 802-1w" command.
VLAN is a member of global SSTP - IEEE 802-1w
    PORT 5/1/47 - FORWARDING
    PORT 5/1/48 - FORWARDING
    PORT 7/1/1 - FORWARDING
    PORT 7/1/2 - FORWARDING
    PORT 7/1/3 - FORWARDING
    PORT 8/1/1 - FORWARDING
    PORT 8/1/2 - FORWARDING
    PORT 8/1/3 - FORWARDING
    PORT 8/1/45 - FORWARDING
Client-1(config)#
```

Client-2 configuration

```
Client-2(config)# spanning-tree single 802-1w
Client-2(config)# end
Client-2(config)# show 802-1w vlan 1905
Single spanning tree is enabled. use "show 802-1w" command.
VLAN is a member of global SSTP - IEEE 802-1w
    PORT 1/1/1 - FORWARDING
    PORT 1/1/2 - FORWARDING
    PORT 1/1/3 - FORWARDING
    PORT 2/1/47 - BLOCKING
    PORT 2/1/48 - BLOCKING
    PORT 3/1/1 - FORWARDING
    PORT 3/1/2 - FORWARDING
    PORT 3/1/3 - FORWARDING
Client-2(config)#
```

Example 3: Configure Multiple Spanning Tree (MSTP) on the MCT Clients

MSTP (802.1s) allows mutliple VLANs to be managed by a single STP instance, and several VLANs can be mapped to a reduced number of spanning-tree instances. Use MSTP on MCT clients to ensure loop-free topology for one or more VLANs that have a similar Layer 2 topology.

MSTP requires that BPDU flooding be enabled on the MCT Cluster devices. This example is based on the network topology shown in Figure 44 on page 143.

Router-1 configuration

```
Router-1(config)# bpdu-flood-enable
Warning - Any recieved untagged BPDUs will now be flooded to all the ports.
```

Router-2 configuration

```
Router-2(config)# bpdu-flood-enable
Warning - Any received untagged BPDUs will now be flooded to all the ports.
```

Client-1 configuration

```
Client-1(config)# mstp scope all
Enter MSTP scope would remove STP and topology group related configuration for system
Are you sure? (enter 'y' or 'n'): y
'MSTP Start' need to be entered in order to activate this MSTP feature
Client-1(config)# mstp start
Client-1(config)# mstp instance 1 vlan 1901
Client-1(config)# mstp instance 1 vlan 1905
Client-1(config)#
```

Client-2 configuration

```
Client-2(config)# mstp scope all
Enter MSTP scope would remove STP and topology group related configuration for system
Are you sure? (enter 'y' or 'n'): y
'MSTP Start" need to be entered in order to activate this MSTP feature
Client-2(config)# mstp start
Client-2(config)# mstp instance 1 vlan 1901
Client-2(config)# mstp instance 1 vlan 1905
Client-2(config)
```

GVRP

- GVRP overview....149
- GVRP application examples....149
- VLAN names created by GVRP....152
- Configuration notes for GVRP....152
- Configuring GVRP....153
- Clearing GVRP statistics....155
- Configuration example: Implementing the applications of GVRP....156

GVRP overview

GARP VLAN Registration Protocol (GVRP) is a Generic Attribute Registration Protocol (GARP) application that provides VLAN registration service by means of dynamic configuration (registration) and distribution of VLAN membership information.

A Ruckus device enabled for GVRP can do the following:

- Learn about VLANs from other Ruckus devices and configure those VLANs on the ports that learn about the VLANs. The device listens for GVRP Protocol Data Units (PDUs) from other devices, and implements the VLAN configuration information in the PDUs.
- Advertise VLANs configured on the device to other Ruckus devices. The device sends GVRP PDUs advertising its VLANs to other devices. GVRP advertises statically configured VLANs and VLANs learned from other devices through GVRP.

GVRP enables a Ruckus device to dynamically create 802.1Q-compliant VLANs on links with other devices that are running GVRP. GVRP reduces the chances for errors in VLAN configuration by automatically providing VLAN ID consistency across the network. You can use GVRP to propagate VLANs to other GVRP-aware devices automatically, without the need to manually configure the VLANs on each device. In addition, if the VLAN configuration on a device changes, GVRP automatically changes the VLAN configurations of the affected devices.

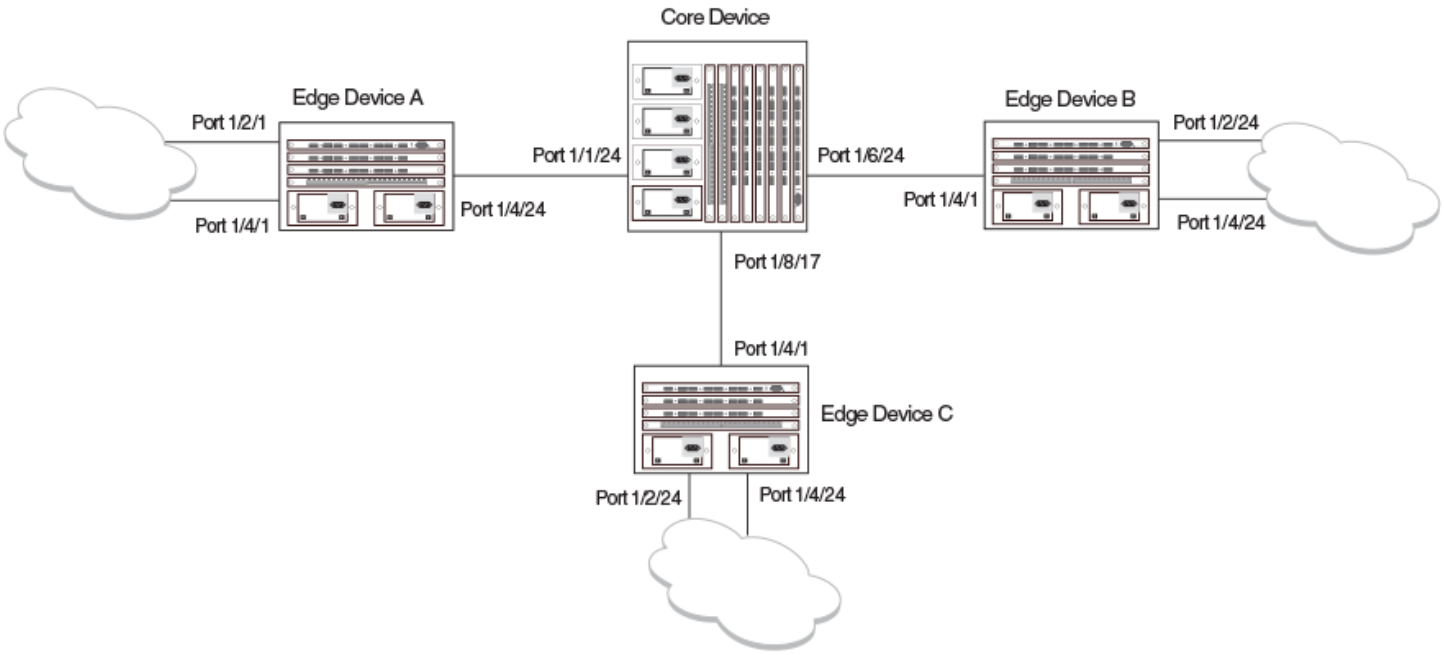
The Ruckus implementation of GARP and GVRP is based on the following standards:

- ANSI/IEEE standard 802.1D, 1998 edition
- IEEE standard 802.1Q, 1998 edition; approved December 8, 1998
- IEEE draft P802.1w/D10, March 26, 2001
- IEEE draft P802.1u/D9, November 23, 2000
- IEEE draft P802.1t/D10, November 20, 2000

GVRP application examples

The following figure shows an example of a network that uses GVRP. This section describes various ways you can use GVRP in a network such as this one. Configuration example: Implementing the applications of GVRP on page 156 lists the CLI commands to implement the applications of GVRP described in this section.

FIGURE 45 Example of GVRP



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 GVRP in this network depend on which devices the feature is enabled on, and whether both learning and advertising are enabled. In this type of network (a core device and edge devices), you can have the following four combinations:

- Dynamic core and fixed edge
- Dynamic core and dynamic edge
- Fixed core and dynamic edge
- Fixed core and fixed edge

Dynamic core and fixed edge

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. GVRP learning is disabled on the edge devices.

Core device	Edge device A	Edge device B	Edge device C
• GVRP is enabled on all ports. • Both learning and advertising are enabled. **NOTE** Since learning is disabled on all the edge devices, advertising on the core device has no effect in this configuration.	• GVRP is enabled on port 1/4/24. Learning is disabled. • VLAN 20 • Port 1/2/1 (untagged) • Port 1/4/24 (tagged) • VLAN 40 • Port 1/4/1 (untagged) • Port 1/4/24 (tagged)	• GVRP is enabled on port 1/4/1. Learning is disabled. • VLAN 20 • Port 1/2/24 (untagged) • Port 1/4/1 (tagged) • VLAN 30 • Port 1/4/24 (untagged) • Port 1/4/1 (tagged)	• GVRP is enabled on port 1/4/1. Learning is disabled. • VLAN 30 • Port 1/2/24 (untagged) • Port 1/4/1 (tagged) • VLAN 40 • Port 1/4/24 (untagged) • Port 1/4/1 (tagged)

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 GVRP on the core device results in the following VLAN configuration on the device:

- VLAN 20
  - 1/1/24 (tagged)
  - 1/6/24 (tagged)
- VLAN 30
  - 1/6/24 (tagged)
  - 1/8/17 (tagged)
- VLAN 40
  - 1/1/24 (tagged)
  - 1/8/17 (tagged)

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 GVRP VLAN ports configured by GVRP are tagged, to ensure that the port can be configured for additional VLANs.

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 GVRP. GVRP can dynamically add other ports to the statically configured VLANs but cannot delete statically configured ports from the VLANs.

Dynamic core and dynamic edge

GVRP is enabled on the core device and on the edge devices. This type of configuration is useful if the devices in the edge clouds are running GVRP and advertise their VLANs to the edge devices. The edge devices learn the VLANs and also advertise them to the core. In this configuration, you do not need to statically configure the VLANs on the edge or core devices, although you can have statically configured VLANs on the devices. The devices learn the VLANs from the devices in the edge clouds.

Fixed core and dynamic edge

GVRP learning is enabled on the edge devices. The VLANs on the core device are statically configured, and the core device is enabled to advertise its VLANs but not to learn VLANs. The edge devices learn the VLANs from the core.

Fixed core and fixed edge

The VLANs are statically configured on the core and edge devices. On each edge device, VLAN advertising is enabled but learning is disabled. GVRP is not enabled on the core device. This configuration enables the devices in the edge clouds to learn the VLANs configured on the edge devices.

VLAN names created by GVRP

The **show vlans** command lists VLANs created by GVRP as "GVRP_VLAN_*vlan-id* ". VLAN names for statically configured VLANs are not affected. To distinguish between statically-configured VLANs that you add to the device and VLANs that you convert from GVRP-configured VLANs into statically-configured VLANs, the **show vlans** command displays a converted VLAN name as "STATIC_VLAN_*vlan-id* ".

Configuration notes for GVRP

- If you disable GVRP, all GVRP configuration information is lost if you save the configuration change (**write memory** command) and then reload the software. However, if you reload the software without first saving the configuration change, the GVRP configuration is restored following a software reload.
- The maximum number of VLANS supported on a device enabled for GVRP is the same as the maximum number on a device that is not enabled for GVRP.
  - To display the maximum number of VLANs allowed on your device, enter the **show default values** command. See the "vlan" row in the System Parameters section. Make sure you allow for the default VLAN (1), the GVRP base VLAN (4093), and the Single STP VLAN (4094). These VLANs are maintained as "Registration Forbidden" in the GVRP database. Registration Forbidden VLANs cannot be advertised or learned by GVRP.
  - To increase the maximum number of VLANs supported on the device, enter the **system-max vlan***num* command at the global CONFIG level of the CLI, then save the configuration and reload the software. The maximum number you can specify is listed in the Maximum column of the **show default values** display.
- The default VLAN (VLAN 1) is not advertised by the Ruckus implementation of GVRP. The default VLAN contains all ports that are not members of statically configured VLANs or VLANs enabled for GVRP.

  NOTE
  The default VLAN has ID 1 by default. You can change the VLAN ID of the default VLAN, but only before GVRP is enabled. You cannot change the ID of the default VLAN after GVRP is enabled.

- Single STP must be enabled on the device. Ruckus implementation of GVRP requires Single STP. If you do not have any statically configured VLANs on the device, you can enable Single STP as follows.

```
device(config)#vlan 1
device(config-vlan-1)#exit
device(config)#span
device(config)#span single
```

These commands enable configuration of the default VLAN (VLAN 1), which contains all the device ports, and enable STP and Single STP.

- All VLANs that are learned dynamically through GVRP are added to the single spanning tree.
- All ports that are enabled for GVRP become tagged members of the GVRP base VLAN (4093). If you need to use this VLAN ID for another VLAN, you can change the GVRP VLAN ID. The software adds the GVRP base VLAN to the single spanning tree.
- All VLAN ports added by GVRP are tagged.
- GVRP is supported only for tagged ports or for untagged ports that are members of the default VLAN. GVRP is not supported for ports that are untagged and are members of a VLAN other than the default VLAN.
- To configure GVRP on a LAG, enable the protocol on the LAG virtual interface. The GVRP configuration of the LAG virtual interface is automatically applied to the other ports in the LAG.
- You can use GVRP on a device even if the device has statically configured VLANs. GVRP does not remove any ports from the statically configured VLANs, although GVRP can add ports to the VLANS. GVRP advertises the statically configured VLANs. Ports added by GVRP do not appear in the running-config and will not appear in the startup-config file when save the configuration. You can manually add a port to make the port a permanent member of the VLAN. After you manually add the port, the port will appear in the running-config and be saved to the startup-config file when you save the configuration.
- VLANs created by GVRP do not support virtual routing interfaces or protocol-based VLANs. virtual routing interfaces and protocol-based VLANs are still supported on statically configured VLANs even if GVRP adds ports to those VLANs.
- You cannot manually configure any parameters on a VLAN that is created by GVRP. For example, you cannot change STP parameters for the VLAN.
- The GVRP timers (Join, Leave, and Leaveall) must be set to the same values on all the devices that are exchanging information using GVRP.
- If the network has a large number of VLANs, the GVRP traffic can use a lot of CPU resources. If you notice high CPU utilization after enabling GVRP, set the GVRP timers to longer values. In particular, set the Leaveall timer to a longer value.
- The feature is supported only on Ethernet ports.

NOTE
If you plan to change the GVRP base VLAN ID (4093) or the maximum configurable value for the Leaveall timer (300000 ms by default), you must do so before you enable GVRP.

Configuring GVRP

To configure a device for GVRP, globally enable support for the feature, then enable the feature on specific ports. Optionally, you can disable VLAN learning or advertising on specific interface.

1. On any device on which you want to configure GVRP, from privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. (Optional) Change the GVRP base VLAN ID.

   By default, GVRP uses VLAN 4093 as base VLAN for the protocol. All ports that are enabled for GVRP become tagged members of this VLAN. If you need to use the VLAN ID 4093 for a statically configured VLAN, you can change the GVRP base VLAN ID.

   ```
   device(config)# gvrp-base-vlan-id 1001
   ```

   If you want to change the GVRP base VLAN ID, you must do so before enabling GVRP.

3. (Optional) Increase the maximum value you can configure for the Leaveall timer.

   You must configure this before enabling GVRP.

   ```
   device(config)# gvrp-max-leaveall-timer 1000000
   ```

4. Enable GVRP globally.

   ```
   device(config)# gvrp-enable
   ```

5. Enable GVRP on individual ports.

   ```
   device(config-gvrp)# enable ethernet 1/1/1 to 1/1/7 ethernet 1/2/1 ethernet 1/2/7 ethernet 1/2/11
   ```

   This example enables GVRP on all ports. If you want to enable GVRP on specific ports, specify the ports on which you want to enable GVRP.

6. Disable VLAN advertisement on required GVRP enabled ports.

   ```
   device(config-gvrp)# block-applicant ethernet 1/2/7 ethernet 1/2/11
   ```

7. (Optional) Disable VLAN learning on required GVRP enabled ports.

   ```
   device(config-gvrp)# block-learning ethernet 1/1/3
   ```

8. (Optional) Configure the Join, Leave, and Leaveall timers.

   ```
   device(config-gvrp)# join-timer 200 leave-timer 600 leaveall-timer 10000
   ```

   You can use the **default-timers** command to reset the timers to their default values.

9. Verify GVRP configuration.

   ```
   device(config-gvrp)# show gvrp
   GVRP is enabled on the system
   GVRP BASE VLAN ID          : 1001
   GVRP MAX Leaveall Timer    : 1000000 ms
   GVRP Join Timer            : 200 ms
   GVRP Leave Timer           : 600 ms
   GVRP Leave-all Timer       : 10000 ms
   ============================================================================
   Configuration that is being used:
    block-learning ethe 1/1/3
    block-applicant ethe 1/2/7 ethe 1/2/11
    enable ethe 1/1/1 to 1/1/7 ethe 1/2/1 ethe 1/2/7 ethe 1/2/11
   ============================================================================
   Spanning Tree: SINGLE SPANNING TREE
   Dropped Packets Count: 0
   ============================================================================
   Number of VLANs in the GVRP Database: 15
   Maximum Number of VLANs that can be present: 4095
   ============================================================================
   ```

10. Verify the GVRP statistics.

```
device(config-gvrp)# show gvrp statistics ethernet 1/2/1
PORT 1/2/1 Statistics:
 Leave All Received                    : 147
 Join Empty Received                   : 4193
 Join In Received                      : 599
 Leave Empty Received                  : 0
 Leave In Received                     : 0
 Empty Received                        : 588
 Leave All Transmitted                 : 157
 Join Empty Transmitted                : 1794
 Join In Transmitted                   : 598
 Leave Empty Transmitted               : 0
 Leave In Transmitted                  : 0
 Empty Transmitted                     : 1248
 Invalid Messages/Attributes Skipped   : 0
 Failed Registrations                  : 0
```

Clearing GVRP statistics

GVRP session counters can be cleared using a CLI command.

Ensure that GVRP is enabled in your network.

To determine the effect of clearing the VRRP statistics, an appropriate **show** command is entered before and after the **clear** command

1. From the privileged EXEC mode, enter the **show gvrp statistics** command for Ethernet 1/2/1.

```
device# show gvrp statistics ethernet 1/2/1
PORT 1/2/1 Statistics:
 Leave All Received                    : 147
 Join Empty Received                   : 4193
 Join In Received                      : 599
 Leave Empty Received                  : 0
 Leave In Received                     : 0
 Empty Received                        : 588
 Leave All Transmitted                 : 157
 Join Empty Transmitted                : 1794
 Join In Transmitted                   : 598
 Leave Empty Transmitted               : 0
 Leave In Transmitted                  : 0
 Empty Transmitted                     : 1248
 Invalid Messages/Attributes Skipped   : 0
 Failed Registrations                  : 0
```

2. Enter the **clear gvrp statistics** command for the interface Ethernet 1/2/1.

```
device# clear gvrp statistics ethernet 1/2/1
```

3. Enter the **show gvrp statistics** command for Ethernet 1/2/1.

```
device# show gvrp statistics ethernet 1/2/1
PORT 1/2/1 Statistics:
 Leave All Received                    : 0
 Join Empty Received                   : 0
 Join In Received                      : 0
 Leave Empty Received                  : 0
 Leave In Received                     : 0
 Empty Received                        : 0
 Leave All Transmitted                 : 0
 Join Empty Transmitted                : 0
 Join In Transmitted                   : 0
 Leave Empty Transmitted               : 0
 Leave In Transmitted                  : 0
 Empty Transmitted                     : 0
 Invalid Messages/Attributes Skipped   : 0
 Failed Registrations                  : 0
```

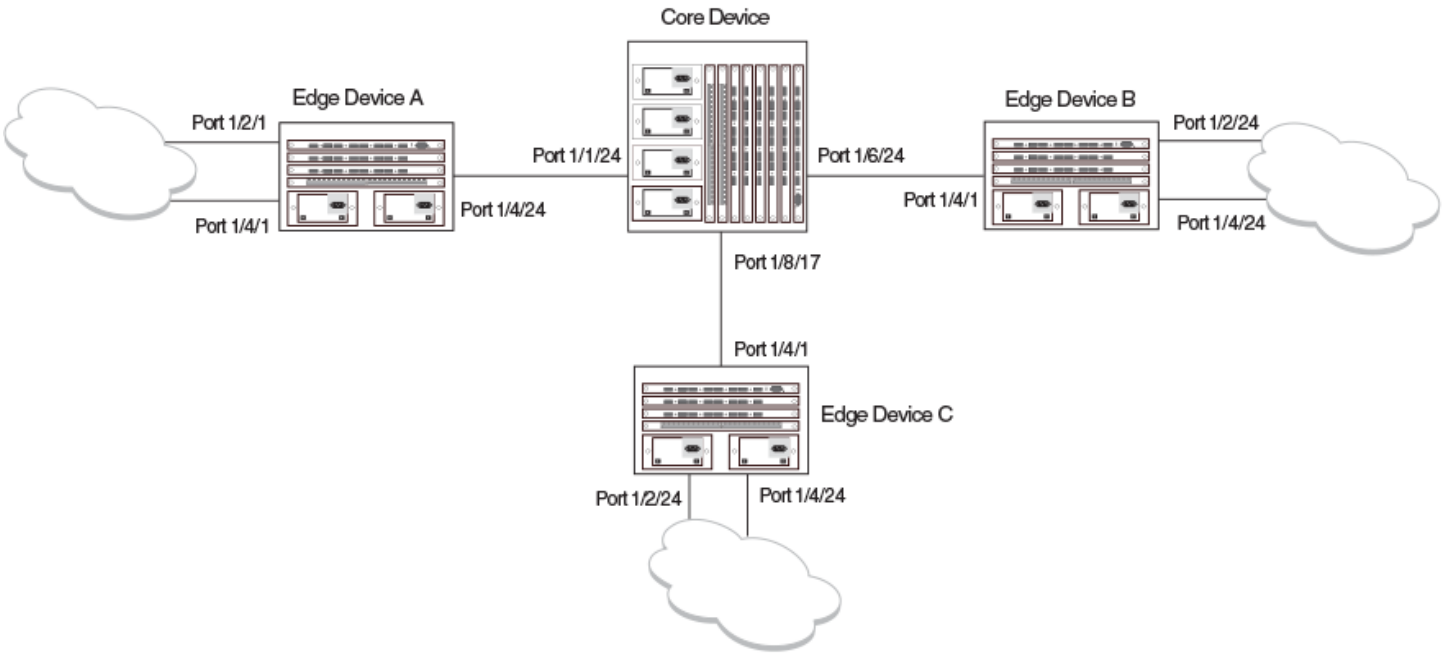
In this show output for a specified interface after the **clear gvrp statistics** command has been entered, you can see that the statistical counters have been reset.

Configuration example: Implementing the applications of GVRP

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 GVRP in this network depend on which devices the feature is enabled on, and whether both learning and advertising are enabled.

FIGURE 46 GVRP 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 GVRP.

Dynamic core and fixed edge

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.

```
device> enable
device# configure terminal
device(config)# gvrp-enable
device(config-gvrp)# enable all
```

These commands globally enable GVRP support and enable the protocol on all ports.

Enter the following commands on edge device A.

```
device> enable
device# configure terminal
device(config)# vlan 20
device(config-vlan-20)# untagged ethernet 1/2/1
device(config-vlan-20)# tagged ethernet 1/4/24
device(config-vlan-20)# vlan 40
device(config-vlan-40)# untagged ethernet 1/2/1
device(config-vlan-40)# tagged ethernet 1/4/24
device(config-vlan-40)# exit
device(config)# gvrp-enable
device(config-gvrp)# enable ethernet 1/4/24
device(config-gvrp)# block-learning ethernet 1/4/24
```

These commands statically configure two port-based VLANs, enable GVRP on port 1/4/24, and block GVRP learning on the port. The device will advertise the VLANs but will not learn VLANs from other devices.

Enter the following commands on edge device B.

```
device> enable
device# configure terminal
device(config)# vlan 20
device(config-vlan-20)# untagged ethernet 1/2/24
device(config-vlan-20)# tagged ethernet 1/4/1
device(config-vlan-20)# vlan 30
device(config-vlan-30)# untagged ethernet 1/4/24
device(config-vlan-30)# tagged ethernet 1/4/1
device(config-vlan-30)# exit
device(config)# gvrp-enable
device(config-gvrp)# enable ethernet 1/4/1
device(config-gvrp)# block-learning ethernet 1/4/1
```

Enter the following commands on edge device C.

```
device> enable
device# configure terminal
device(config)# vlan 30
device(config-vlan-30)# untagged ethernet 1/2/24
device(config-vlan-30)# tagged ethernet 1/4/1
device(config-vlan-20)# vlan 40
device(config-vlan-40)# untagged ethernet 1/4/24
device(config-vlan-40)# tagged ethernet 1/4/1
device(config-vlan-40)# exit
device(config)# gvrp-enable
device(config-gvrp)# enable ethernet 1/4/1
device(config-gvrp)# block-learning ethernet 1/4/1
```

Dynamic core and dynamic edge

In this configuration, the core and edge devices have no statically configured VLANs and are enabled to learn and advertise VLANs. The edge and core devices learn the VLANs configured on the devices in the edge clouds. To enable GVRP on all the ports, enter the following command on each edge device and on the core device.

```
device> enable
device# configure terminal
device(config)# gvrp-enable
device(config-gvrp)# enable all
```

Fixed core and dynamic edge

In this configuration, GVRP learning is enabled on the edge devices. The VLANs on the core device are statically configured, and the core device is enabled to advertise its VLANs but not to learn VLANs. The edge devices learn the VLANs from the core.

Enter the following commands on the core device.

```
device> enable
device# configure terminal
device(config)# vlan 20
device(config-vlan-20)# tagged ethernet 1/1/24
device(config-vlan-20)# tagged ethernet 1/6/24
device(config-vlan-20)# vlan 30
device(config-vlan-30)# tagged ethernet 1/6/24
device(config-vlan-30)# tagged ethernet 1/8/17
device(config-vlan-30)# vlan 40
device(config-vlan-40)# tagged ethernet 1/1/5
device(config-vlan-40)# tagged ethernet 1/8/17
device(config-vlan-40)# vlan 50
device(config-vlan-50)# untag ethernet 1/6/1
device(config-vlan-50)# tagged ethernet 1/1/11
device(config-vlan-50)# exit
device(config)# gvrp-enable
device(config-gvrp)# enable ethernet 1/1/24 ethernet 1/6/24 ethernet 1/8/17
device(config-gvrp)# block-learning ethernet 1/1/24 ethernet 1/6/24 ethernet 1/8/17
```

These VLAN commands configure VLANs 20, 30, 40, and 50. The GVRP commands enable the protocol on the ports that are connected to the edge devices, and disable VLAN learning on those ports. All the VLANs are advertised by GVRP.

Enter the following commands on edge devices A, B, and C.

```
device> enable
device# configure terminal
device(config)# gvrp-enable
device(config-gvrp)# enable all
device(config-gvrp)# block-applicant all
```

Fixed core and fixed edge

The VLANs are statically configured on the core and edge devices. On each edge device, VLAN advertising is enabled but learning is disabled. GVRP is not configured on the core device. This configuration enables the devices in the edge clouds to learn the VLANs configured on the edge devices.

This configuration does not use any GVRP configuration on the core device.

The configuration on the edge device is the same as in Dynamic core and fixed edge on page 157.

Spanning Tree Protocol

- STP overview....159
- Standard STP parameter configuration....159
- STP feature configuration....168
- PVST/PVST+ compatibility....221
- PVRST compatibility....230
- BPDU guard....230
- Root guard....232
- Designated Protection....233
- Packet InError Detection....234
- Error disable recovery....235
- 802.1s Multiple Spanning Tree Protocol....237

STP overview

STP (IEEE 802.1D bridge protocol) is supported on all Ruckus ICX devices. STP detects and eliminates logical loops in the network. STP also ensures that the least cost path is taken when multiple paths exist between ports or VLANs. If the selected path fails, STP searches for and then establishes an alternate path to prevent or limit retransmission of data.

The Spanning Tree Protocol (STP) eliminates Layer 2 loops in networks, by selectively blocking some ports and allowing other ports to forward traffic, based on global (bridge) and local (port) parameters you can configure.

STP related features, such as RSTP and PVST, extend the operation of standard STP, enabling you to fine-tune standard STP and avoid some of its limitations.

You can enable or disable STP on a global basis (for the entire device), a port-based VLAN basis (for the individual Layer 2 broadcast domain), or an individual port basis.

Configuration procedures are provided for the standard STP bridge and port parameters.

Standard STP parameter configuration

Ruckus Layer 2 Switches and Layer 3 Switches support standard STP as described in the IEEE 802.1D specification. STP is enabled by default on Layer 2 Switches but disabled by default on Layer 3 Switches.

By default, each port-based VLAN on a Ruckus device runs a separate spanning tree (a separate instance of STP). 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, then each of those VLANs on which STP is enabled and VLAN 1 all run separate spanning trees.

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. Thus, 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.

STP parameters and defaults

The following table lists the default STP states for Ruckus devices.

TABLE 16 Default STP states

Device type	Default STP type	Default STP state	Default STP state of new VLANs[1]
Switch image default	STP 802.1d	Enabled	Enabled
Router image default	No span	Enabled	Disabled
Base L3 image default	No span	Disabled	Disabled

1. When you create a port-based VLAN, the new VLAN STP state is the same as the default STP state on the device. The new VLAN does not inherit the STP state of the default VLAN. The Single Spanning Tree Protocol (SSTP) is another type of STP. SSTP includes all VLANs on which STP is enabled in a single spanning tree. Refer to Single Spanning Tree (SSTP) on page 214.

The following table lists the default STP bridge parameters. The bridge parameters affect the entire spanning tree. If you are using MSTP, the parameters affect the VLAN. If you are using SSTP, the parameters affect all VLANs that are members of the single spanning tree.

TABLE 17 Default STP 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.	15 seconds Possible values: 4 - 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 - 40 seconds
Hello Time	The interval of time between each configuration BPDU sent by the root bridge.	2 seconds Possible values: 1 - 10 seconds
Priority	A parameter used to identify the root bridge in a spanning tree (instance of STP). 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 - 65535

NOTE
If you plan to change STP bridge timers, Ruckus recommends that you stay within the following ranges, from section 8.10.2 of the IEEE STP specification. 2 * (forward_delay -1) >= max_agemax_age >= 2 * (hello_time +1)

The following table lists the default STP port parameters. The port parameters affect individual ports and are separately configurable on each port.

TABLE 18 Default STP port parameters

Parameter	Description	Default and valid values
Priority	The preference that STP 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 - 240 (configurable in increments of 16)

TABLE 18 Default STP port parameters (continued)

Parameter	Description	Default and valid values
Path Cost	The cost of using the port to reach the root bridge. When selecting among multiple links to the root bridge, STP chooses the link with the lowest path cost and blocks the other paths. Each port type has its own default STP path cost.	10 Mbps - 100 100 Mbps - 19 Gbps - 4 10 Gbps - 2 Possible values are 0 - 65535

Enabling or disabling the Spanning Tree Protocol (STP)

STP is *enabled* by default on devices running Layer 2 code. STP is *disabled* by default on devices running Layer 3 code.

You can enable or disable STP 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 STP within a port-based VLAN, the setting overrides the global setting. Thus, you can enable STP for the ports within a port-based VLAN even when STP is globally disabled, or disable the ports within a port-based VLAN when STP is globally enabled.
- Individual port - Affects only the individual port. However, if you change the STP state of the LAG virtual interface, the change affects all ports in the LAG.

NOTE
The CLI converts the STP groups into topology groups when you save the configuration. For backward compatibility, you can still use the STP group commands. However, the CLI converts the commands into the topology group syntax. Likewise, the **show stp-group** command displays STP topology groups.

Configuration modes for STP

The following configuration modes apply while configuring STP.

- Spanning-tree single - This configuration can be enabled on systems running IEEE 802.1D. The single spanning tree controls all the 4000 VLANs. You can opt in and out of this single spanning tree using the **spanning-tree** command under the VLAN prompt.
- Spanning-tree single 802.1w - This configuration can be enabled on systems running IEEE 802.1w. The single rapid spanning tree controls all the 4000 VLANs. The VLAN can opt in and out of this single rapid spanning tree using the **spanning-tree** command under the VLAN prompt. If there is a "spanning-tree" configuration under the VLAN, that VLAN will be with that single 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 - In this configuration mode you can turn on 802.1D or 802.1w (Rapid Spanning Tree) at the VLAN level individually.

Enabling or disabling STP globally

Use the following method to enable or disable STP on a device on which you have not configured port-based VLANs.

NOTE
When you configure a VLAN, the VLAN inherits the global STP settings. However, once you begin to define a VLAN, you can no longer configure standard STP parameters globally using the CLI. From that point on, you can configure STP only within individual VLANs.

To enable STP for all ports in all VLANs on a Ruckus ICX device, enter the **spanning-tree** command.

```
device(config)# spanning-tree
```

The **spanning-tree** command enables a separate spanning tree in each VLAN, including the default VLAN.

To set system maximum value for number of spanning tree instances, enter the command such as the following:

```
device(config)# system-max spanning-tree 254
```

NOTE
The number of spanning tree instances ranges from 1 through 254 on ICX 7750, ICX 7450, and ICX 7250 devices. The range of STP instances on Ruckus ICX 7150 device is from 1 through 253. The default value is 128 on ICX 7750 device and 32 on ICX 7450, ICX 7250, and Ruckus ICX 7150 devices.

Enabling or disabling STP in a port-based VLAN

Use the following procedure to disable or enable STP on a device on which you have configured a port-based VLAN. Changing the STP state in a VLAN affects only that VLAN.

To enable STP for all ports in a port-based VLAN, enter commands such as the following.

```
device(config)# vlan 10
device(config-vlan-10)# spanning-tree
```

Enabling or disabling STP on an individual port

Use the following procedure to disable or enable STP on an individual port.

NOTE
If you change the STP state of the LAG virtual interface, it affects all ports in the LAG.

To enable STP on an individual port, enter commands such as the following.

```
device(config)# interface 1/1/1
device(config-if-e1000-1/1/1)# spanning-tree
```

Changing STP bridge and port parameters

STP bridge and port parameters are preset with default values but various parameters such as priority can be modified using CLI commands.

Table 17 on page 160 and Table 18 on page 160 list the default STP parameters. If you need to change the default value for an STP parameter, use the following procedures.

You can modify the following STP Parameters:

- Bridge parameters—forward delay, maximum age, hello time, and priority
- Port parameters—priority and path cost

Changing STP bridge parameters

NOTE
If you plan to change STP bridge timers, Ruckus recommends that you stay within the following ranges, from section 8.10.2 of the IEEE STP specification. 2 * (forward_delay -1) >= max_age max_age >= 2 * (hello_time +1)

To change a STP bridge priority on a Ruckus device to the highest value to make the device the root bridge, enter the following command.

```
device(config)#spanning-tree priority 0
```

The command in this example changes the priority on a device on which you have not configured port-based VLANs. The change applies to the default VLAN. If you have configured a port-based VLAN on the device, you can configure the parameters only at the configuration level for individual VLANs. Enter commands such as the following.

```
device(config)#vlan 20
device(config-vlan-20)#spanning-tree priority 0
```

To make this change in the default VLAN, enter the following commands.

```
device(config)#vlan 1
device(config-vlan-1)#spanning-tree priority 0
```

The **forward-delay***value* parameter specifies the forward delay and can be a value from 4 - 30 seconds. The default is 15 seconds.

> **NOTE**
> You can configure a Ruckus device for faster convergence (including a shorter forward delay) using Fast Span or Fast Uplink Span. Refer to STP feature configuration on page 168.

The **hello-time** *value* parameter specifies the hello time and can be a value from 1 - 10 seconds. The default is 2 seconds.

> **NOTE**
> This parameter applies only when this device or VLAN is the root bridge for its spanning tree.

The **maximum-age** *value* parameter specifies the amount of time the device waits for receipt of a configuration BPDU from the root bridge before initiating a topology change. You can specify from 6 - 40 seconds. The default is 20 seconds.

The **priority** *value* parameter specifies the priority and can be a value from 0 - 65535. A higher numerical value means a lower priority. Thus, the highest priority is 0. The default is 32768.

You can specify some or all of these parameters on the same command line. If you specify more than one parameter, you must specify them in the order shown above, from left to right.

Changing STP port parameters

To change the path and priority costs for a port, enter commands such as the following.

```
device(config)#vlan 10
device(config-vlan-10)#spanning-tree ethernet 5 path-cost 15 priority 64
```

The **path-cost** *value* parameter specifies the port cost as a path to the spanning tree root bridge. STP prefers the path with the lowest cost. You can specify a value from 0 - 65535.

The default depends on the port type:

- 10 Mbps - 100
- 100 Mbps - 19
- Gbps - 4
- 10 Gbps - 2
- The **priority***value* parameter specifies the preference that STP gives this port relative to other ports for forwarding traffic out of the spanning tree. If you are upgrading a device that has a configuration saved under an earlier software release,

and the configuration contains a value from 0 - 7 for a port STP priority, the software changes the priority to the default when you save the configuration while running the new release.

The **disable** and **enable** parameter disables or re-enables STP on the port. The STP state change affects only this VLAN. The port STP state in other VLANs is not changed.

STP protection enhancement

STP protection provides the ability to prohibit an end station from initiating or participating in an STP topology change.

The 802.1W Spanning Tree Protocol (STP) 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 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. 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 STP BPDU, triggering an STP topology change.

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 Protection feature on the Ruckus port to which the end station is connected. STP Protection disables the connected device ability to initiate or participate in an STP topology change, by dropping all BPDUs received from the connected device.

Enabling STP protection

You can enable STP Protection on a per-port basis.

To prevent an end station from initiating or participating in STP topology changes, enter the following command at the Interface level of the CLI.

```
device#(config) interface ethernet 2
device#(config-if-e1000-2)#stp-protect
```

This command causes the port to drop STP BPDUs sent from the device on the other end of the link.

Enter the **no** form of the command to disable STP protection on the port.

Clearing BPDU drop counters

For each port that has STP Protection enabled, the Ruckus device counts and records the number of dropped BPDUs. You can use CLI commands to clear the BPDU drop counters for all ports on the device, or for a specific port on the device.

To clear the BPDU drop counters for all ports on the device that have STP Protection enabled, enter the following command in the Global configuration mode of the CLI.

```
device(config)# clear stp-protect-statistics
```

To clear the BPDU drop counter for a specific port that has STP Protection enabled, enter the following command.

```
device# clear stp-protect-statistics ethernet 1/1/2
```

Viewing the STP Protection Configuration

You can view the STP Protection configuration for all ports on a device, or for a specific port only. The **show stp-protect** command output shows the port number on which STP Protection is enabled, and the number of BPDUs dropped by each port.

To view the STP Protection configuration for all ports on the device, enter the following command at any level of the CLI.

```
device# show stp-protect
Port ID        BPDU Drop Count
1/1/3                  478
1/1/5                  213
1/1/6                    0
1/1/12                  31
```

To view STP Protection configuration for a specific port, enter the following command at any level of the CLI.

```
device# show stp-protect ethernet 1/1/3
STP-protect is enabled on port 1/1/3.  BPDU drop count is 478
```

If you enter the **show stp-protect** command for a port that does not have STP protection enabled, the following message displays on the console.

```
device# show stp-protect ethernet 1/1/4
STP-protect is not enabled on port 1/1/4.
```

Displaying STP information

You can display the following Spanning Tree Protocol (STP) information:

- All the global and interface STP settings
- CPU utilization statistics
- Detailed STP information for each interface
- STP state information for a port-based VLAN
- STP state information for an individual interface

Displaying STP information for an entire device

To display STP information, enter the following command at any level of the CLI.

```
device# show span
VLAN 1 BPDU cam_index is 3 and the Master DMA Are(HEX)
STP instance owned by VLAN 1
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
1         800000e0804d4a00  0        Root    8000   20     2      1      15       689      1
00e0804d4a00
Port STP Parameters:
Port         Prio     Path     State        Fwd     Design    Designated        Designated
Num          rity     Cost                  Trans   Cost      Root              Bridge
             Hex
1/1/1        80       19       FORWARDING   1       0         800000e0804d4a00  800000e0804d4a00
1/1/2        80       0        DISABLED     0       0         0000000000000000  0000000000000000
1/1/3        80       0        DISABLED     0       0         0000000000000000  0000000000000000
1/1/4        80       0        DISABLED     0       0         0000000000000000  0000000000000000
1/1/5        80       19       FORWARDING   1       0         800000e0804d4a00  800000e0804d4a00
1/1/6        80       19       BLOCKING     0       0         800000e0804d4a00  800000e0804d4a00
1/1/7        80       0        DISABLED     0       0         0000000000000000  0000000000000000
<lines for remaining ports excluded for brevity>
```

The **vlan***vlan-id* parameter displays STP information for the specified port-based VLAN.

The **pvst-mode** parameter displays STP information for the device Per VLAN Spanning Tree (PVST+) compatibility configuration. Refer to PVST/PVST+ compatibility on page 221

The *num* parameter displays only the entries after the number you specify. For example, on a device with three port-based VLANs, if you enter 1, then information for the second and third VLANs is displayed, but information for the first VLAN is not displayed. Information is displayed according to VLAN number, in ascending order. The entry number is not the same as the VLAN number. For example, if you have port-based VLANs 1, 10, and 2024, then the command output has three STP entries. To display information for VLANs 10 and 2024 only, enter **show span 1** .

The **detail** parameter and its additional optional parameters display detailed information for individual ports. Refer to Displaying detailed STP information for each interface on page 166.

Displaying the STP state of a port-based VLAN

When you display information for a port-based VLAN, that information includes the STP state of the VLAN.

To display information for a port-based VLAN, enter a command such as the following at any mode of the CLI.

```
device# show vlans
Total PORT-VLAN entries: 2
Maximum PORT-VLAN entries: 16
legend: [S=Slot]
PORT-VLAN 1, Name DEFAULT-VLAN, Priority level0, Spanning tree On
 Untagged Ports: (S3)  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16
 Untagged Ports: (S3) 17 18 19 20 21 22 23 24
 Untagged Ports: (S4)  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17
 Untagged Ports: (S4) 18 19 20 21 22 23 24
   Tagged Ports: None
   Uplink Ports: None
PORT-VLAN 2, Name greenwell, Priority level0, Spanning tree Off
 Untagged Ports: (S1)  1  2  3  4  5  6  7  8
 Untagged Ports: (S4)  1
   Tagged Ports: None
   Uplink Ports: None
```

Displaying detailed STP information for each interface

To display the detailed STP information, enter the following command at any mode of the CLI.

```
device# show span detail
========================================================================
VLAN 1 - MULTIPLE SPANNING TREE (MSTP) ACTIVE
========================================================================
Bridge identifier - 0x800000e0804d4a00
Active global timers - Hello: 0
Port 1/1/1 is FORWARDING
Port - Path cost: 19, Priority: 128, Root: 0x800000e052a9bb00
Designated - Bridge: 0x800000e052a9bb00, Interface: 1, Path cost: 0
Active Timers - None
BPDUs - Sent: 11, Received: 0
Port 1/1/2 is DISABLED
Port 1/1/3 is DISABLED
Port 1/1/4 is DISABLED <lines for remaining ports excluded for brevity>
```

If a port is disabled, the only information shown by this command is "DISABLED". If a port is enabled, this display shows the following information.

NOTE

If the configuration includes VLAN groups, the **show span detail** command displays the master VLANs of each group but not the member VLANs within the groups. However, the command does indicate that the VLAN is a master VLAN. The **show span detail vlan** *vlan-id* command displays the information for the VLAN even if it is a member VLAN. To list all the member VLANs within a VLAN group, enter the **show vlan-group** [*group-id*] command.

Displaying detailed STP information for a single port in a specific VLAN

Enter a command such as the following to display STP information for an individual port in a specific VLAN.

```
device# show span detail vlan 1 ethernet 1/1/7
Port 1/1/7 is FORWARDING
Port - Path cost: 19, Priority: 128, Root: 0x800000e052a9bb00
Designated - Bridge: 0x800000e052a9bb00, Interface: 7, Path cost: 0
Active Timers - None
BPDUs - Sent: 29, Received: 0
```

Displaying STP state information for an individual interface

To display STP state information for an individual port, you can use the methods in Displaying STP information for an entire device on page 165 or Displaying detailed STP information for each interface on page 166. You also can display STP state information for a specific port using the following method.

To display information for a specific port, enter a command such as the following at any level of the CLI.

```
device#show interface ethernet 1/1/3
FastEthernet 1/1/3 is up, line protocol is up
Port up for 1 hour 50 minutes 30 seconds
  Hardware is FastEthernet, address is 0000.00a9.bb49 (bia 0000.00a9.bb49)
  Configured speed auto, actual 100Mbit, configured duplex fdx, actual fdx
  Member of L2 VLAN ID 1, port is untagged, port state is FORWARDING
  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
  MTU 1518 bytes, encapsulation ethernet
  5 minute input rate: 352 bits/sec, 0 packets/sec, 0.00% utilization
  5 minute output rate: 0 bits/sec, 0 packets/sec, 0.00% utilization
  1238 packets input, 79232 bytes, 0 no buffer
  Received 686 broadcasts, 0 runts, 0 giants
  0 input errors, 0 CRC, 0 frame, 0 ignored
  529 multicast
  918 packets output, 63766 bytes, 0 underruns
  0 output errors, 0 collisions
```

The STP information is shown in bold type in this example.

You also can display the STP states of all ports by entering the show interface brief command such as the following, which uses the **brief** parameter.

```
device#show interface brief
Port  Link State     Dupl Speed Trunk Tag Priori MAC            Name
1/1/1   Down None      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
1/1/5   Down None      None None  None  No  level0 0000.00a9.bb04
1/1/6   Down None      None None  None  No  level0 0000.00a9.bb05
1/1/7   Down None      None None  None  No  level0 0000.00a9.bb06
1/1/8   Down None      None None  None  No  level0 0000.00a9.bb07
.
. some rows omitted for brevity
.
1/3/10  Down None      None None  None  No  level0 0000.00a9.bb4a
1/3/11  Up   Forward
   Full 100M  None  No  level0 0000.00a9.bb49
```

In the example above, only one port, 1/3/11, is forwarding traffic toward the root bridge.

STP feature configuration

Spanning Tree Protocol (STP) features extend the operation of standard STP, enabling you to fine- tune standard STP and avoid some of its limitations.

This section describes how to configure these parameters on Ruckus Layer 3 Switches using the CLI.

Fast port span

When STP is running on a device, message forwarding is delayed during the spanning tree recalculation period following a topology change. The STP forward delay parameter specifies the period of time a bridge waits before forwarding data packets. The forward delay controls the listening and learning periods of STP reconvergence. You can configure the forward delay to a value from 4 - 30 seconds. The default is 15 seconds. Thus, using the standard forward delay, convergence requires 30 seconds (15 seconds for listening and an additional 15 seconds for learning) when the default value is used.

This slow convergence is undesirable and unnecessary in some circumstances. The Fast Port Span feature allows certain ports to enter the forwarding state in four seconds. Specifically, Fast Port Span allows faster convergence on ports that are attached to end stations and thus do not present the potential to cause Layer 2 forwarding loops. Because the end stations cannot cause forwarding loops, they can safely go through the STP state changes (blocking to listening to learning to forwarding) more quickly than is allowed by the standard STP convergence time. Fast Port Span performs the convergence on these ports in four seconds (two seconds for listening and two seconds for learning).

In addition, Fast Port Span enhances overall network performance in the following ways:

- Fast Port Span reduces the number of STP topology change notifications on the network. When an end station attached to a Fast Span port comes up or down, the Ruckus device does not generate a topology change notification for the port. In this situation, the notification is unnecessary since a change in the state of the host does not affect the network topology.
- Fast Port Span eliminates unnecessary MAC cache aging that can be caused by topology change notifications. Bridging devices age out the learned MAC addresses in their MAC caches if the addresses are unrefreshed for a given period of time, sometimes called the MAC aging interval. When STP sends a topology change notification, devices that receive the notification use the value of the STP forward delay to quickly age out their MAC caches. For example, if a device normal MAC aging interval is 5 minutes, the aging interval changes temporarily to the value of the forward delay (for example, 15 seconds) in response to an STP topology change.

In normal STP, the accelerated cache aging occurs even when a single host goes up or down. Because Fast Port Span does not send a topology change notification when a host on a Fast Port Span port goes up or down, the unnecessary cache aging that can occur in these circumstances under normal STP is eliminated.

Fast Port Span is a system-wide parameter and is enabled by default. Thus, when you boot a device, all the ports that are attached only to end stations run Fast Port Span. For ports that are not eligible for Fast Port Span, such as ports connected to other networking devices, the device automatically uses the normal STP settings. If a port matches any of the following criteria, the port is ineligible for Fast Port Span and uses normal STP instead:

- The port is 802.1Q tagged
- The port is a member of a trunk group
- The port has learned more than one active MAC address
- An STP Configuration BPDU has been received on the port, thus indicating the presence of another bridge on the port.

You also can explicitly exclude individual ports from Fast Port Span if needed. For example, if the only uplink ports for a wiring closet switch are Gbps ports, you can exclude the ports from Fast Port Span.

Disabling and re-enabling fast port span

Fast Port Span is a system-wide parameter and is enabled by default. Therefore, all ports that are eligible for Fast Port Span use it.

To disable or re-enable Fast Port Span, enter the following commands.

```
device(config)# no fast port-span
device(config)# write memory
```

NOTE
The **fast port-span** command has additional parameters that let you exclude specific ports. These parameters are shown in the following section.

To re-enable Fast Port Span, enter the following commands.

```
device(config)# fast port-span
device(config)# write memory
```

Excluding specific ports from fast port span

To exclude a port from Fast Port Span while leaving Fast Port Span enabled globally, enter commands such as the following.

```
device(config)# fast port-span exclude ethernet 1/1/7
device(config)# write memory
```

To exclude a set of ports from Fast Port Span, enter commands such as the following.

```
device(config)# fast port-span exclude ethernet 1/1/1 ethernet 1/2/2 ethernet 1/2/3
device(config)# write memory
```

To exclude a contiguous (unbroken) range of ports from Fast Span, enter commands such as the following.

```
device(config)# fast port-span exclude ethernet 1/1/1 to 1/1/24
device(config)# write memory
```

To re-enable Fast Port Span on a port, enter a command such as the following.

```
device(config)# no fast port-span exclude ethernet 1/1/1
device(config)# write memory
```

This command re-enables Fast Port Span on port 1 only and does not re-enable Fast Port Span on other excluded ports. You also can re-enable Fast Port Span on a list or range of ports using the syntax shown above this example.

To re-enable Fast Port Span on all excluded ports, disable and then re-enable Fast Port Span by entering the following commands.

```
device(config)# no fast port-span
device(config)# fast port-span
device(config)# write memory
```

Disabling and then re-enabling Fast Port Span clears the exclude settings and thus enables Fast Port Span on all eligible ports. To make sure Fast Port Span remains enabled on the ports following a system reset, save the configuration changes to the startup-config file after you re-enable Fast Port Span. Otherwise, when the system resets, those ports will again be excluded from Fast Port Span.

Fast Uplink Span

The Fast Port Span feature described in the previous section enhances STP performance for end stations. The Fast Uplink Span feature enhances STP performance for wiring closet switches with redundant uplinks. Using the default value for the standard STP forward delay, convergence following a transition from an active link to a redundant link can take 30 seconds (15 seconds for listening and an additional 15 seconds for learning).

You can use the Fast Uplink Span feature on a device deployed as a wiring closet switch to decrease the convergence time for the uplink ports to another device to just one second. The new Uplink port directly goes to forward mode (bypassing listening and learning modes). The wiring closet switch must be a Ruckus ICX device but the device at the other end of the link can be a Ruckus ICX device or another vendor's switch.

Configuration of the Fast Uplink Span feature takes place entirely on the Ruckus ICX device. To configure the Fast Uplink Span feature, specify a group of ports that have redundant uplinks on the wiring closet switch (Ruckus ICX device). If the active link becomes unavailable, the Fast Uplink Span feature transitions the forwarding to one of the other redundant uplink ports in just one second. All Fast Uplink Span-enabled ports are members of a single Fast Uplink Span group.

NOTE
To avoid the potential for temporary bridging loops, Ruckus recommends that you use the Fast Uplink feature only for wiring closet switches (switches at the edge of the network cloud). In addition, enable the feature only on a group of ports intended for redundancy, so that at any given time only one of the ports is expected to be in the forwarding state.

NOTE
When the wiring closet switch (Ruckus ICX device) first comes up or when STP is first enabled, the uplink ports still must go through the standard STP state transition without any acceleration. This behavior guards against temporary routing loops as the switch tries to determine the states for all the ports. Fast Uplink Span acceleration applies only when a working uplink becomes unavailable.

Active uplink port failure

The active uplink port is the port elected as the root port using the standard STP rules. All other ports in the group are redundant uplink ports. If an active uplink port becomes unavailable, Fast Uplink Span transitions the forwarding of traffic to one of the redundant ports in the Fast Uplink Span group in one second bypassing listening and learning port states.

Switchover to the active uplink port

When a failed active uplink port becomes available again, switchover from the redundant port to the active uplink port is delayed by 30 seconds. The delay allows the remote port to transition to forwarding mode using the standard STP rules. After 30 seconds, the blocked active uplink port begins forwarding in just one second and the redundant port is blocked.

NOTE
Use caution when changing the spanning tree priority. If the switch becomes the root bridge, Fast Uplink Span will be disabled automatically.

Fast Uplink Span Rules for LAGs

If you add a port to a Fast Uplink Span group that is a member of a LAG, the following rules apply:

- If you add the LAG virtual interface of a LAG to the Fast Uplink Span group, all other ports in the LAG are automatically included in the group. Similarly, if you remove the LAG virtual interface from the Fast Uplink Span group, the other ports in the LAG are automatically removed from the Fast Uplink Span group.

- You cannot add a subset of the ports in a LAG to the Fast Uplink Span group. All ports in a LAG have the same Fast Uplink Span property, as they do for other port properties.
- If the working LAG is partially down but not completely down, no switch-over to the backup occurs. This behavior is the same as in the standard STP feature.
- If the working LAG is completely down, a backup LAG can go through an accelerated transition only if the following are true:
  - The LAG is included in the fast uplink group.
  - All other ports except those in this LAG are either disabled or blocked. The accelerated transition applies to all ports in this LAG.

When the original working LAG comes back (partially or fully), the transition back to the original topology is accelerated if the conditions listed above are met.

Configuring a Fast Uplink Span Port Group

To configure a group of ports for Fast Uplink Span, enter the following commands:

```
device(config)# fast uplink-span ethernet 1/1/1 to 1/1/4
device(config)# write memory
```

This example configures four ports, 1/1/1 - 1/1/4, as a Fast Uplink Span group. In this example, all four ports are connected to a wiring closet switch. Only one of the links is expected to be active at any time. The other links are redundant. For example, if the link on port 1/1/4 is the active link on the wiring closet switch but becomes unavailable, one of the other links takes over. Because the ports are configured in a Fast Uplink Span group, the STP convergence takes one second instead of taking at least 30 seconds using the standard STP forward delay.

You can add ports to a Fast Uplink Span group by entering the fast uplink-span command additional times with additional ports. The device can have only one Fast Uplink Span group, so all the ports you identify as Fast Uplink Span ports are members of the same group.

To remove a Fast Uplink Span group or to remove individual ports from a group, use "no" in front of the appropriate fast uplink-span command. For example, to remove ports 4/1/3 and 4/1/4 from the Fast Uplink Span group configured above, enter the following commands:

```
device(config)# no fast uplink-span ethernet 1/1/1 to 1/1/4
device(config)# write memory
```

To check the status of ports with Fast Uplink Span enabled.

```
device(config)# show span fast-uplink-span
STP instance owned by VLAN 1
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
   1 000000c100000001 2    1/3/1  8000 20  2   1   15  65      15  000011111111
Port STP Parameters:
Port   Prio Path  State       Fwd    Design  Designated       Designated
Num    rity Cost              Trans  Cost    Root             Bridge
       Hex
1/1/2  80   0     DISABLED    0      0       0000000000000000 0000000000000000
1/1/3  80   0     DISABLED    0      0       0000000000000000 0000000000000000
1/1/4  80   4     FORWARDING  1      2       000000c100000001 8000000011111111
1/1/5  80   0     DISABLED    0      0       0000000000000000 0000000000000000
1/1/6  80   0     DISABLED    0      0       0000000000000000 0000000000000000
1/1/7  80   0     DISABLED    0      0       0000000000000000 0000000000000000
1/1/8  80   0     DISABLED    0      0       0000000000000000 0000000000000000
1/1/9  80   0     DISABLED    0      0       0000000000000000 0000000000000000
```

Configuring Fast Uplink Span within a VLAN

You can also configure Fast Uplink Span on the interfaces within a VLAN.

To configure Fast Uplink Span for a VLAN, enter command such as the following.

```
device(config)#vlan 10
device(config-vlan-10)#untag ethernet 1/1/1 to 1/1/3
device(config-vlan-10)#fast uplink-span ethernet 1/1/1 to 1/1/3
```

To check the status of Fast Uplink Span for a specified VLAN.

```
device(config-vlan-2)#show span vlan 2 fast-uplink-span
STP instance owned by VLAN 2
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
   2 8000000011111111 0    Root   8000 20  2   1   15  29596   0   000011111111
Port STP Parameters:
Port   Prio Path  State       Fwd    Design  Designated       Designated
Num    rity Cost              Trans  Cost    Root             Bridge
       Hex
1/1/1  80   4     LISTENING   0      0       8000000011111111 8000000011111111
```

The VLAN *vlan-id* parameter displays Fast Uplink Span information for the specified VLAN.

802.1W Rapid Spanning Tree (RSTP)

Earlier implementation by Ruckus of Rapid Spanning Tree Protocol (RSTP), which was 802.1W Draft 3, provided only a subset of the IEEE 802.1W standard; whereas the 802.1W RSTP feature provides the full standard. The implementation of the 802.1W Draft 3 is referred to as RSTP Draft 3.

RSTP Draft3 will continue to be supported on Ruckus devices for backward compatibility. However, customers who are currently using RSTP Draft 3 should migrate to 802.1W.

The 802.1W feature provides rapid traffic reconvergence for point-to-point links within a few milliseconds (0 - 500 milliseconds), following the failure of a bridge or bridge port. This reconvergence occurs more rapidly than the reconvergence provided by the 802.1D Spanning Tree Protocol (STP)) or by RSTP Draft 3.

> **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 802.1W, make sure to explicitly configure all point-to-point links in a topology.

The convergence provided by the standard 802.1W protocol occurs more rapidly than the convergence provided by previous spanning tree protocols because of the following:

- Classic or legacy 802.1D STP protocol requires a newly selected Root port to go through listening and learning stages before traffic convergence can be achieved. The 802.1D traffic convergence time is calculated using the following formula.

2 x FORWARD_DELAY + BRIDGE_MAX_AGE.

If default values are used in the parameter configuration, convergence can take up to 50 seconds. (In this document STP will be referred to as 802.1D.)

- RSTP Draft 3 works only on bridges that have Alternate ports, which are the precalculated "next best root port". (Alternate ports provide back up paths to the root bridge.) Although convergence occurs from 0 - 500 milliseconds in RSTP Draft 3, the spanning tree topology reverts to the 802.1D convergence if an Alternate port is not found.

- Convergence in 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.

Bridges and bridge port roles

A bridge in an 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 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 ID 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 802.1W port is referred to as an RST BPDU, while it is operating in 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 802.1W-enabled bridge ports assume a Designated role. Once start-up is complete, the 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 type 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 .

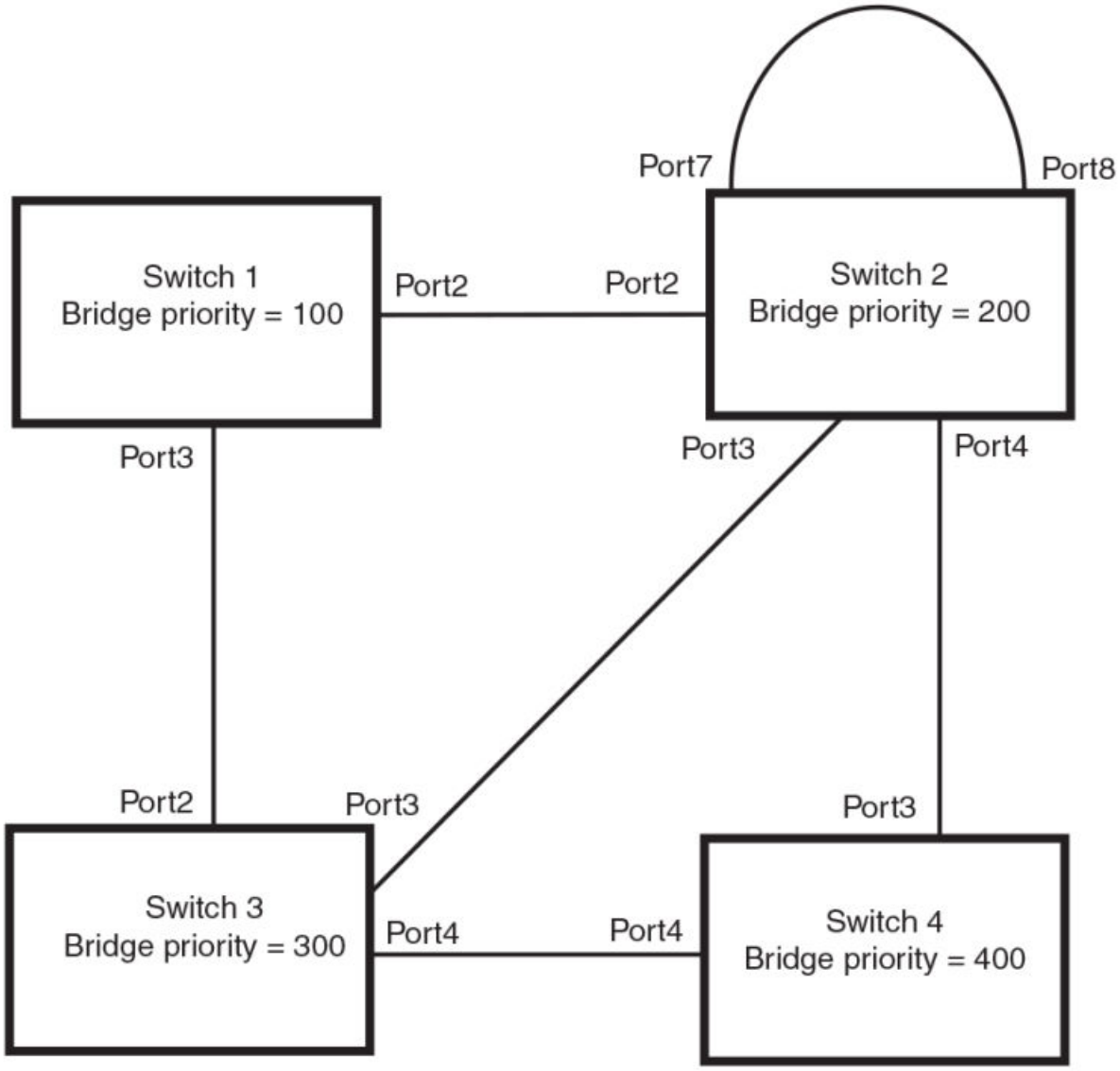
- If the port is down or if 802.1W is disabled on the port, that port is given the role of Disabled port . Disabled ports have no role in the topology. However, if 802.1W is enabled on a port with a link down and the link of that port comes up, then that port assumes one of the following port roles: Root, Designated, Alternate, or Backup.

The following example (Figure 47) 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 47 Simple 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

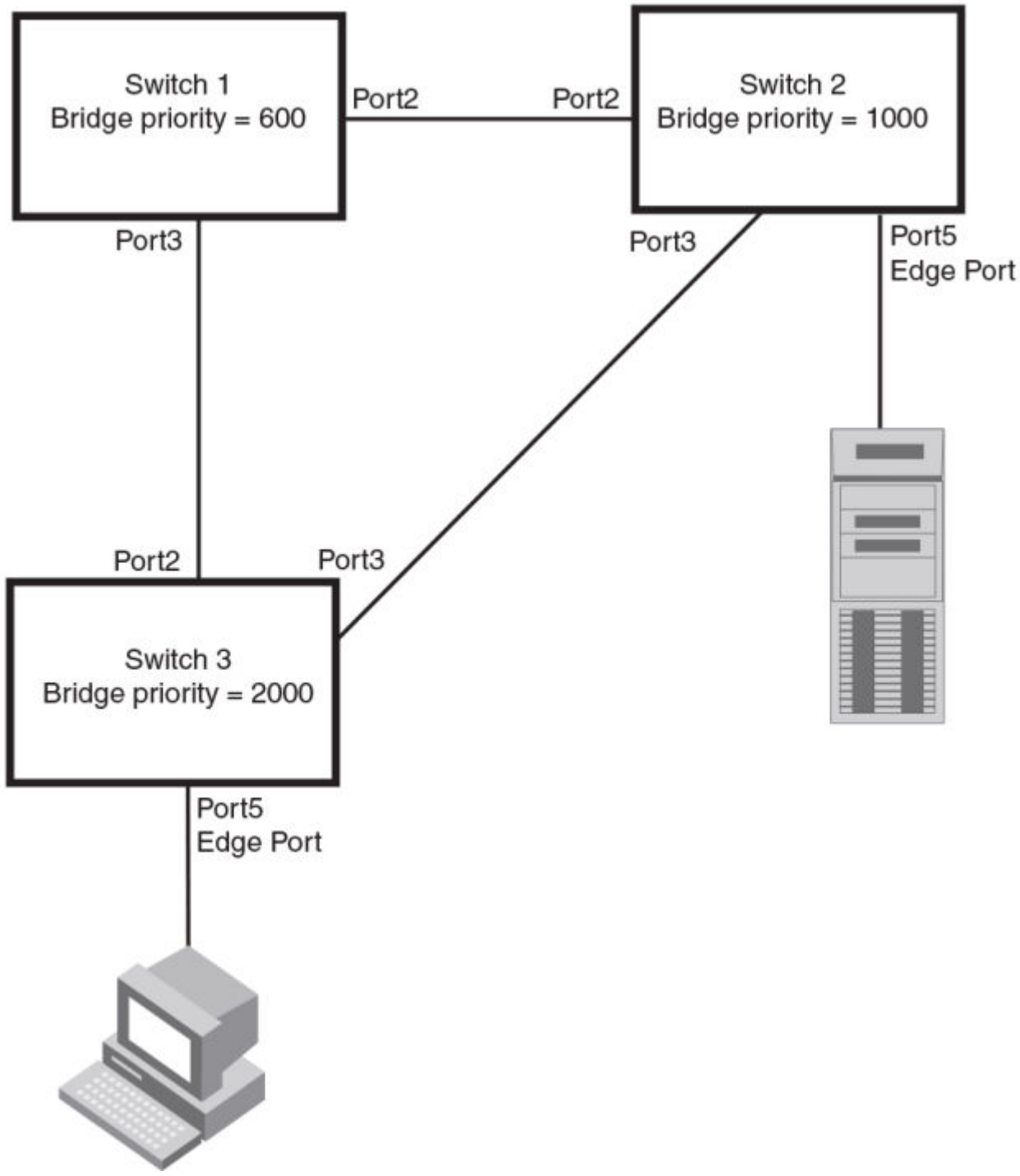
Switch 4 is not directly connected to the root bridge. It has two ports with superior incoming RST BPDUs from two separate LANs: Port3 and Port4. The RST BPDUs received on Port3 are superior to the RST BPDUs received on port 4; therefore, Port3 becomes the Root port and Port4 becomes the Alternate port.

Edge ports and edge port roles

The Ruckus implementation of 802.1W allows ports that are configured as Edge ports to be present in an 802.1W topology. (Figure 48). 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 802.1W does not consider Edge ports in the spanning tree calculations.

FIGURE 48 Topology with edge ports



NOTE
Port numbers are simplified.

However, if any incoming RST BPDU is received from a previously configured Edge port, 802.1W automatically makes the port as 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 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.

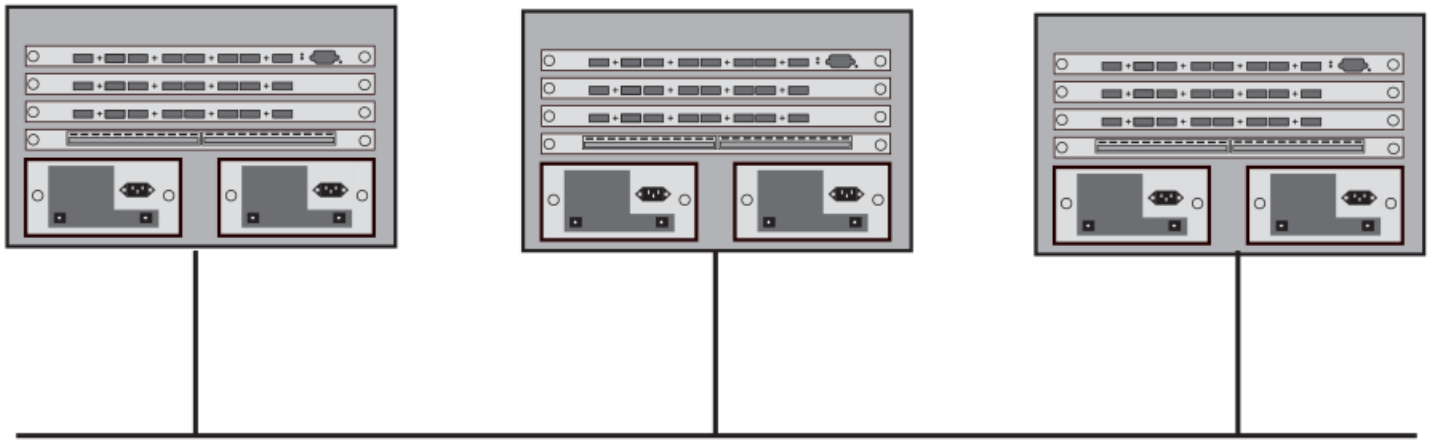
Point-to-point ports

To take advantage of the 802.1W features, ports on an 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 49 Example of shared media



Bridge port states

Ports roles can have one of the following states:

- Forwarding - 802.1W is allowing the port to send and receive all packets.
- Discarding - 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, whose 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 - 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 802.1W. This can occur when the port is disconnected or 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, 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 802.1W detects that the port is an Edge port, that port instantly goes into a forwarding state.

If 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 two hello timer expires 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 802.1D standard, the 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.

802.1W state machines attempt to quickly place the ports into either a forwarding or discarding state. Root ports are quickly placed in 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 a 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 50, 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 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 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, 802.1W uses the Proposing -> Proposed -> Sync -> Synced -> Agreed handshake:

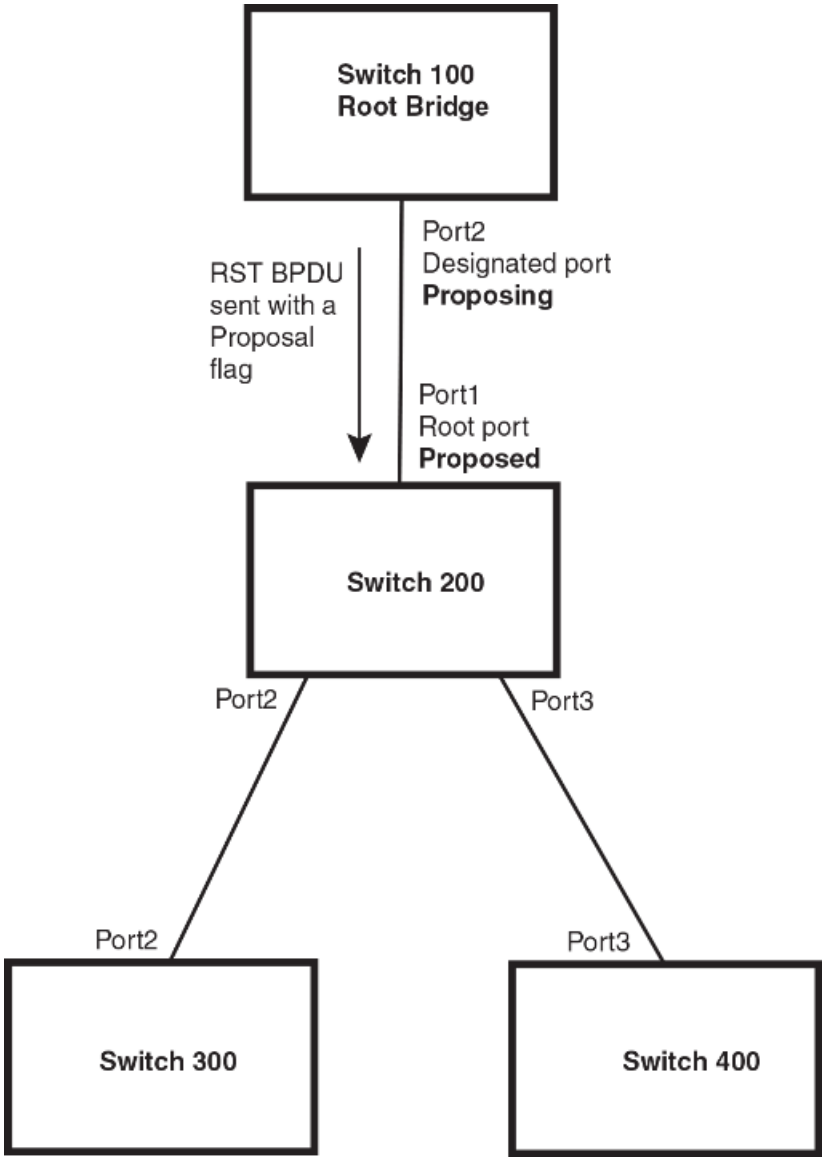
- Proposing - The Designated port on the root bridge sends an RST BPDU packet to its peer port that contains a proposal flag. The proposal flag is a signal that indicates that the Designated port is ready to put itself in a forwarding state (Figure 50). The Designated port continues to send this flag in its RST BPDU until it is placed in a forwarding state (Figure 53) or is forced to operate in 802.1D mode. (Refer to Compatibility of 802.1W with 802.1D on page 205).

- 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 50):
  - 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 the section on Bridges and bridge port roles on page 173.)
  - 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, Port3/Switch 200 is elected as the Root port.

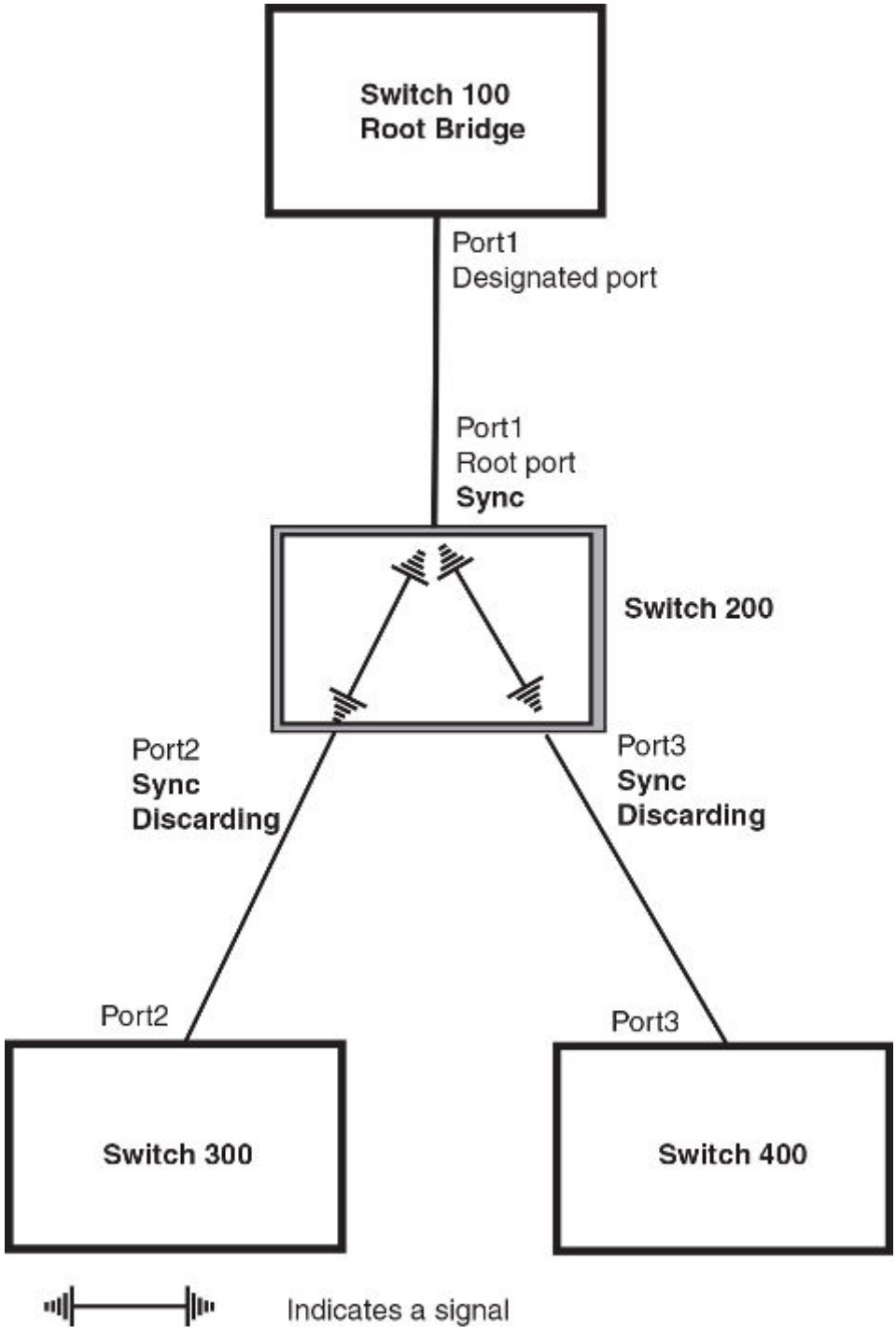
FIGURE 50 Proposing and proposed stage



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 51). 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.

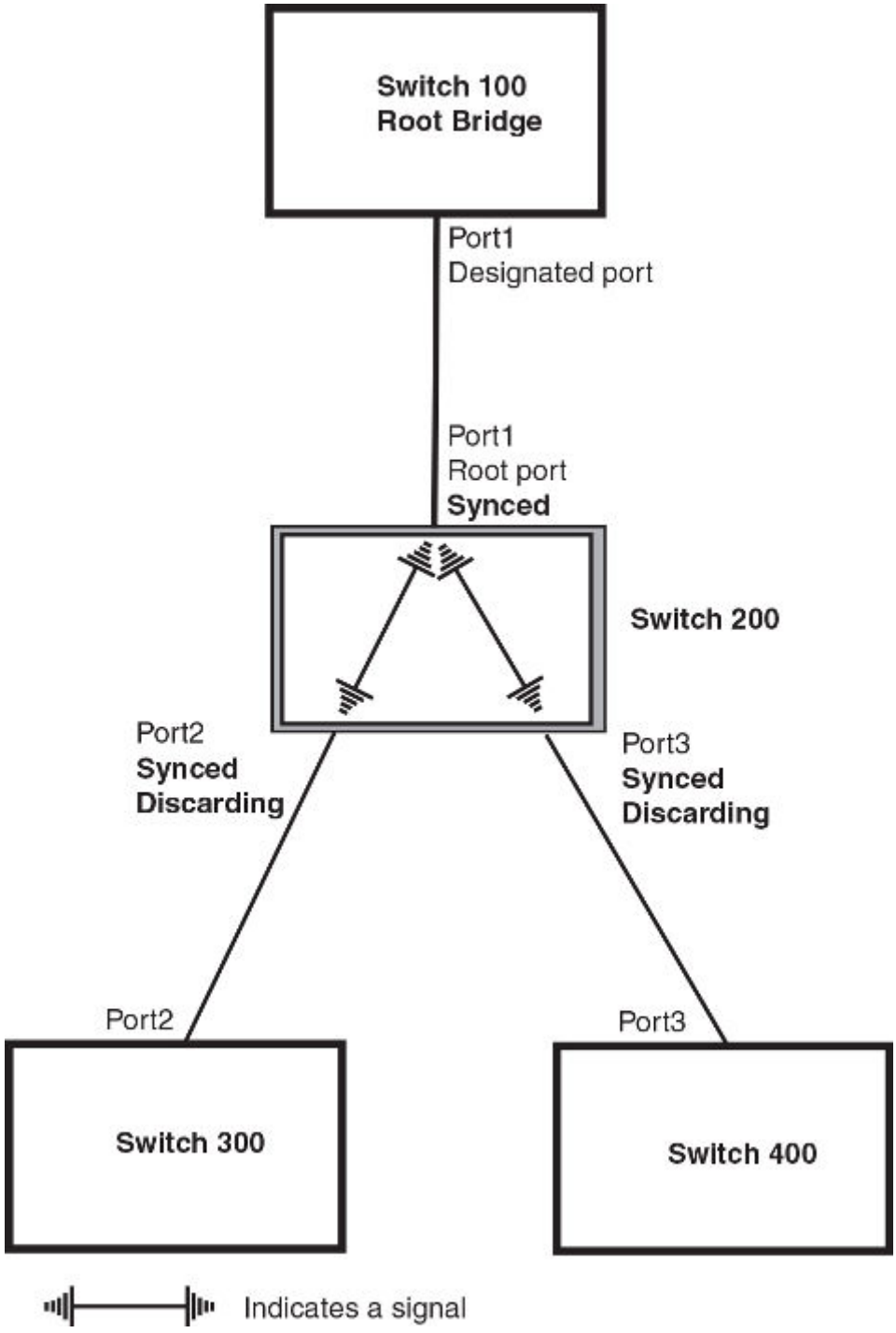
FIGURE 51 Sync stage



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.

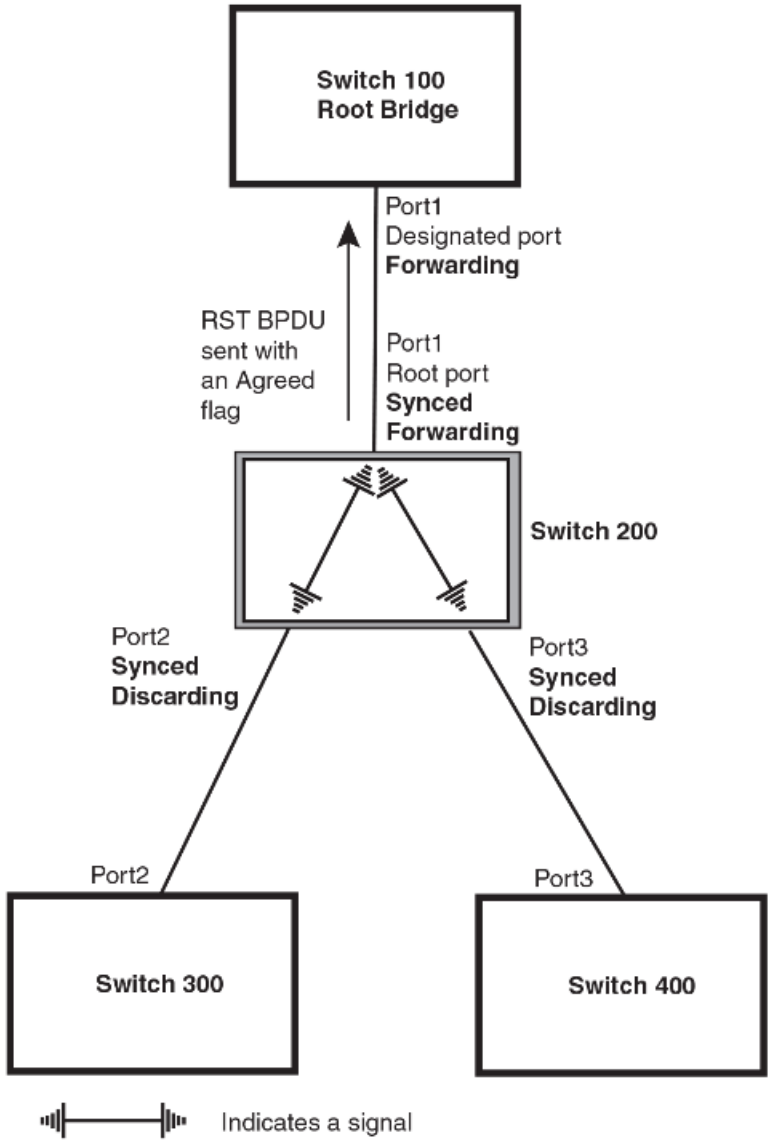
FIGURE 52 Synced stage



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 53 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.

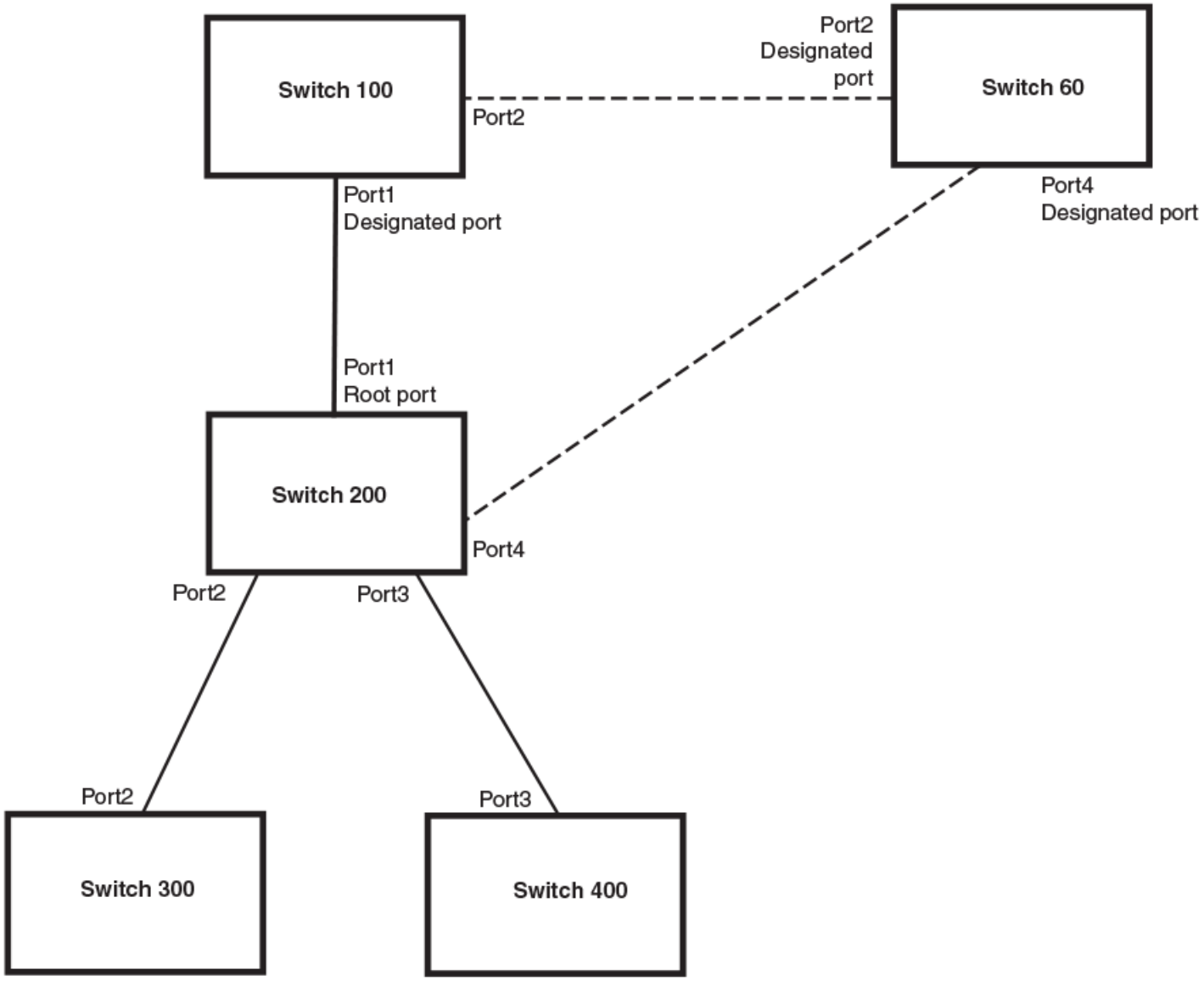
For example, Port2/Switch 200 sends an RST BPDU to Port2/Switch 300 that contains a proposal flag. Port2/Switch 300 asserts a proposed signal. Ports in Switch 300 then set sync signals on the ports to synchronize and negotiate their roles and states. Then the ports assert a synced signal and when the 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, 802.1W uses a different type of handshake. For example, in the following figure, a new root bridge is added to the topology.

FIGURE 54 Addition of a new root bridge



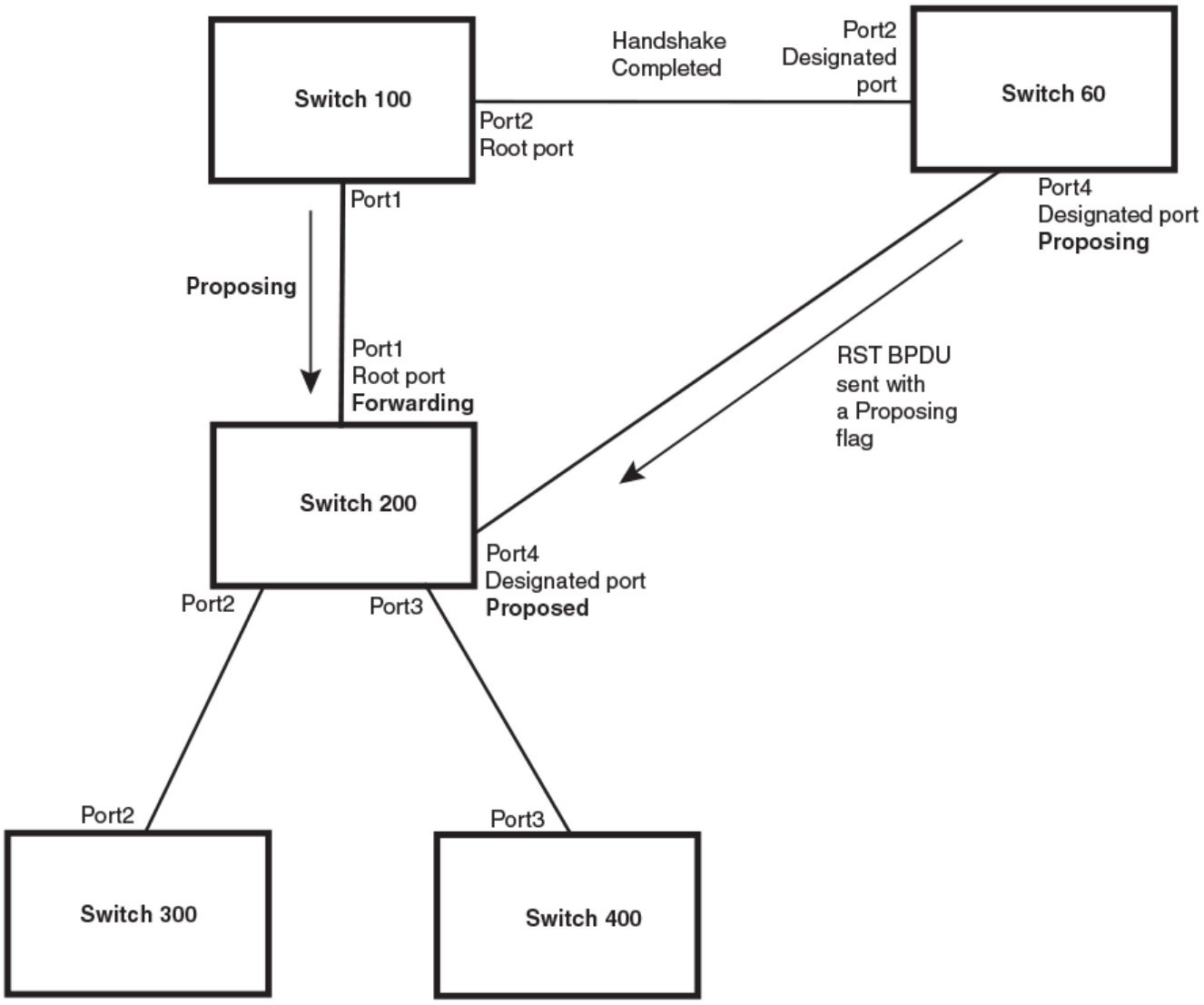
NOTE
Port numbers are simplified.

The handshake that occurs between Switch 60 and Switch 100 follows the one described in the previous section (Handshake when no root port is elected). The former root bridge becomes a non-root bridge and establishes a Root port (Figure 55).

However, since Switch 200 already had a Root port in a forwarding state, 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 55). 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.

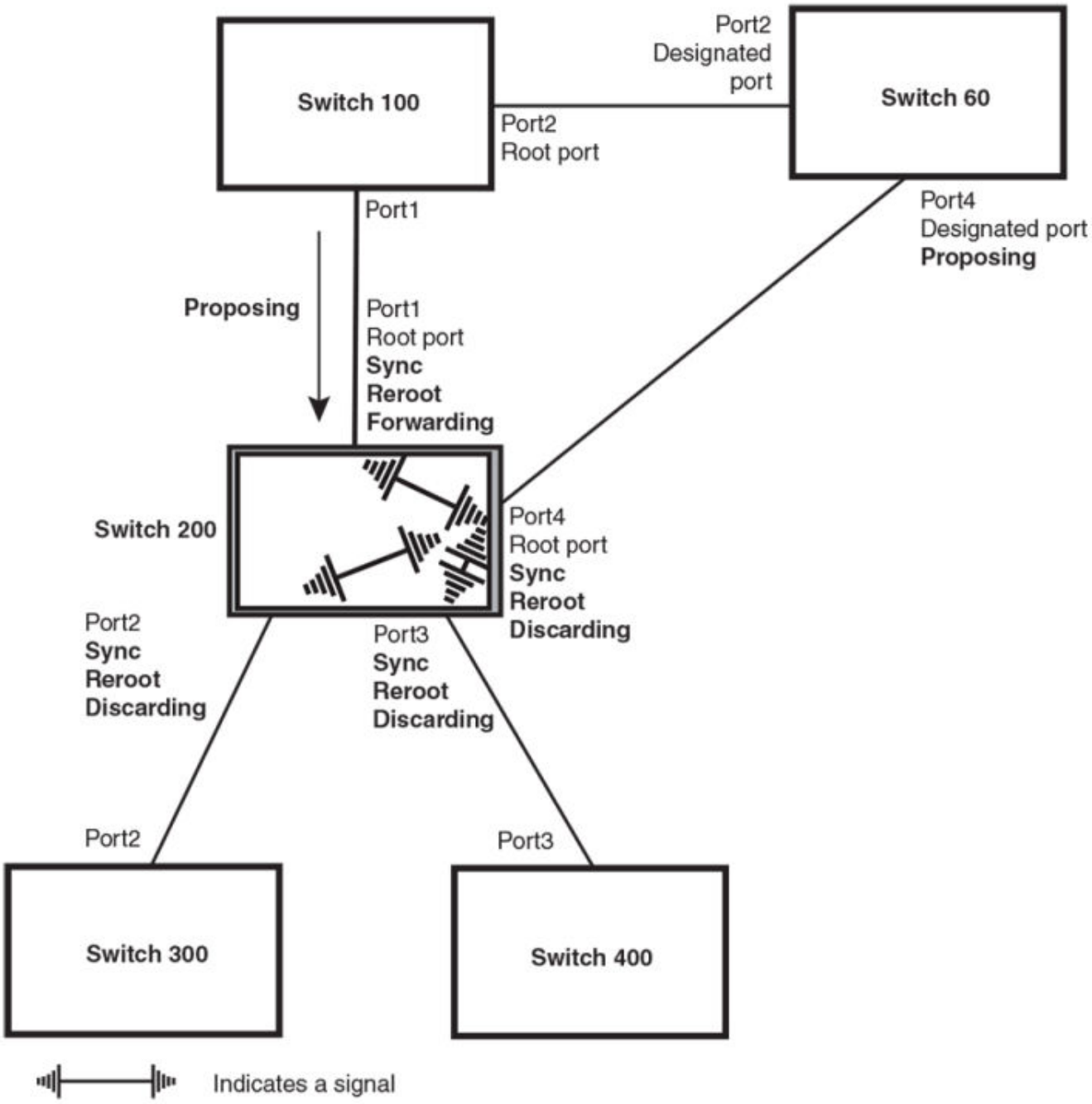
FIGURE 55 New root bridge sending a proposal flag



NOTE
Port numbers are simplified.

- 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.

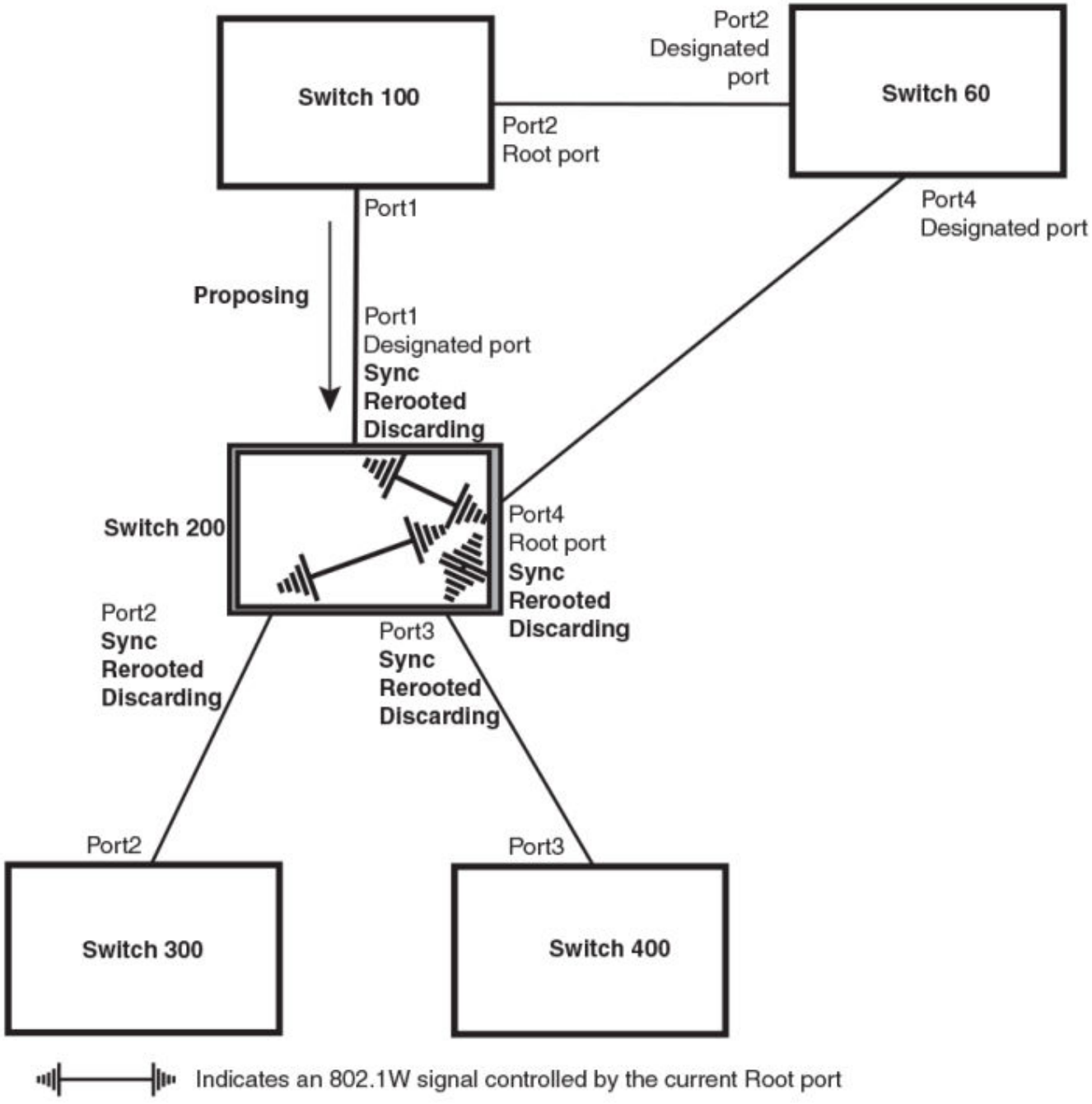
FIGURE 56 Sync and reroot



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.

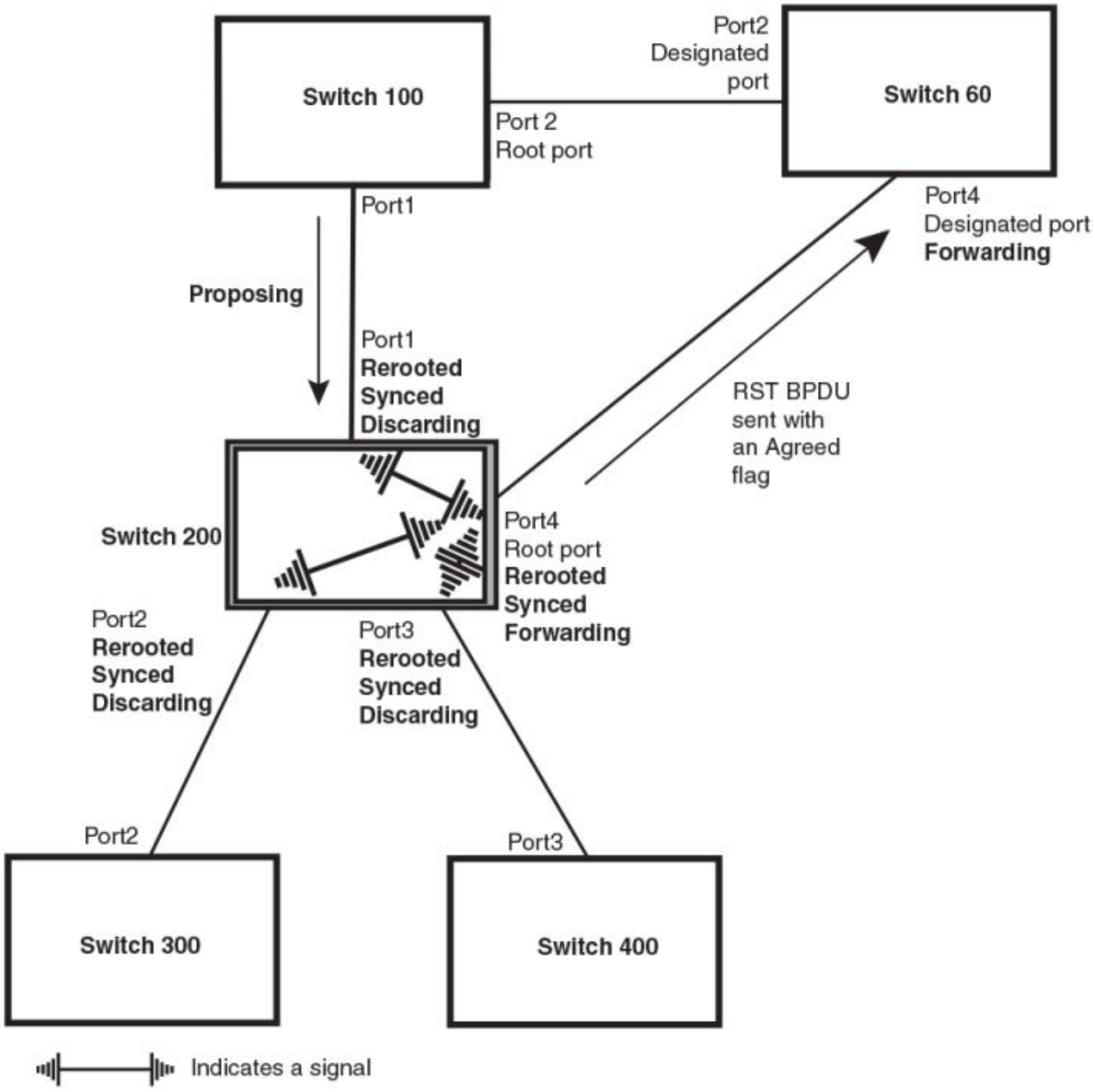
FIGURE 57 Sync and rerooted



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 58 Rerooted, synced, and agreed

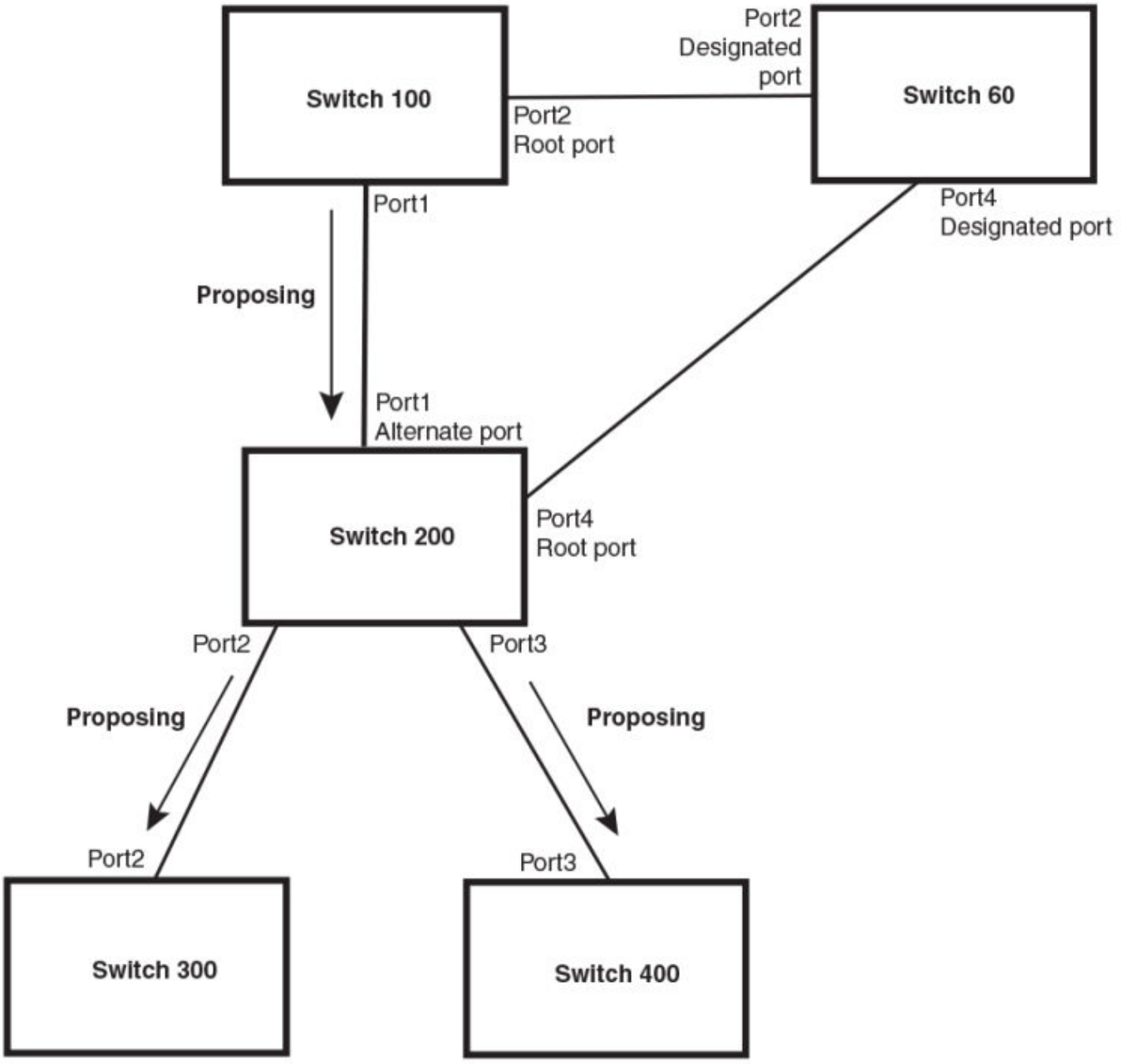


NOTE
Port numbers are simplified.

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 59 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 forwarding state instantly. It waits until two instances of the forward delay timer expires on the port before it goes into 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.

802.1W convergence in a simple topology

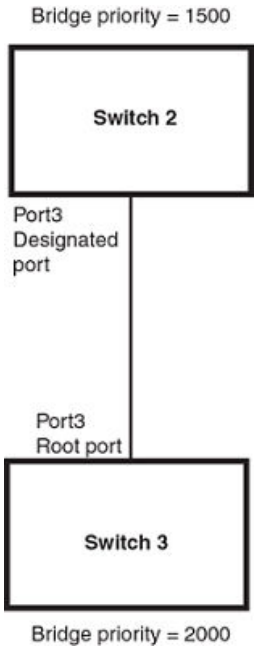
The examples in this section illustrate how 802.1W convergence occurs in a simple Layer 2 topology at start-up.

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 60 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 at 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 ports with a Designated role sends the proposal flag in its RST BPDU when they are ready to move to a forwarding state.

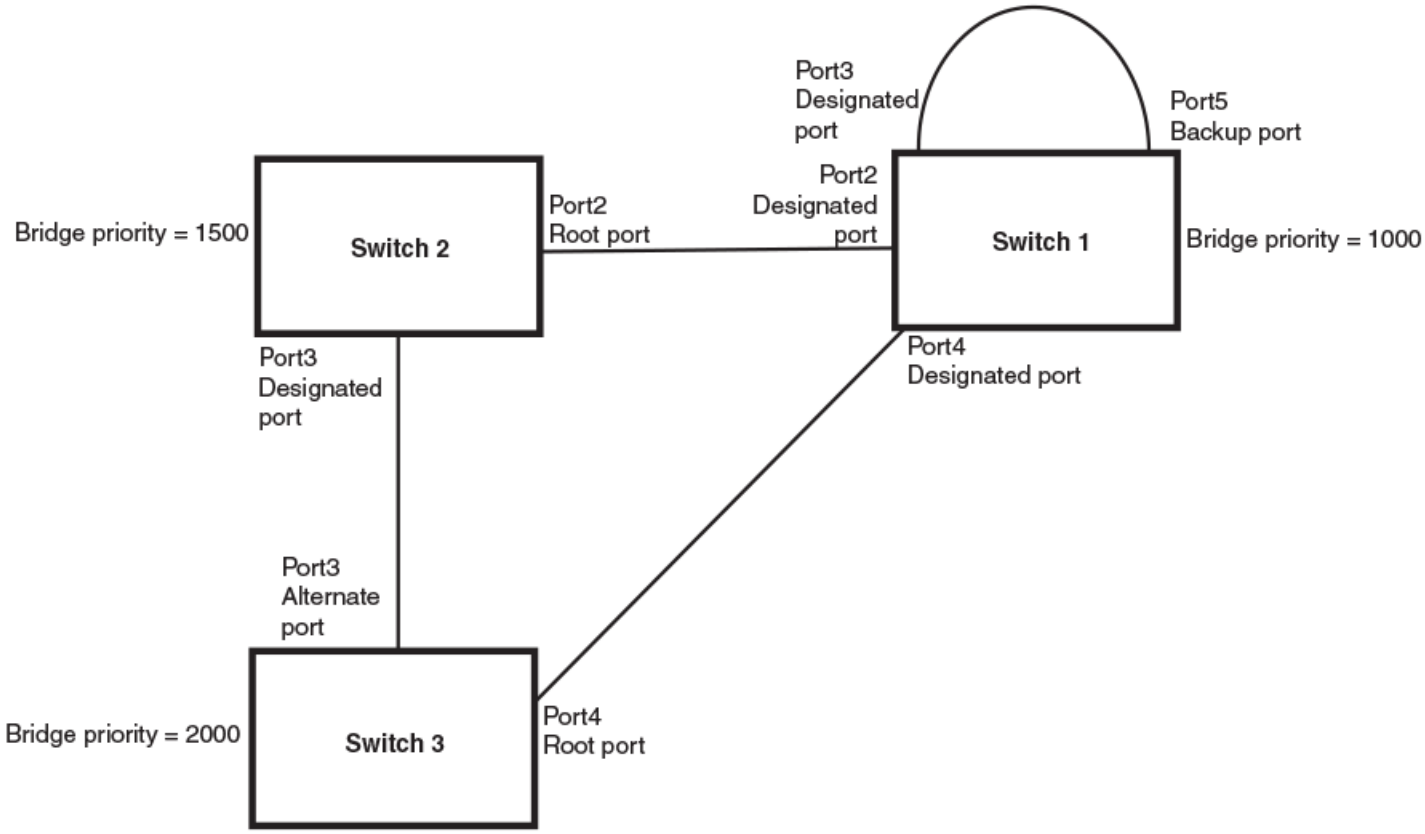
Port3/Switch 3, which starts with a role of Designated port, receives the RST BPDU and finds that it is superior to what it can transmit; therefore, Port3/Switch 3 assumes a new port role, that of a Root port. Port3/Switch 3 transmits an RST BPDU with an agreed flag back to Switch 2 and immediately goes into a forwarding state.

Port3/Switch 2 receives the RST BPDU from Port3/Switch 3 and immediately goes into a forwarding state.

Now 802.1W has fully converged between the two bridges, with Port3/Switch 3 as an operational root port in forwarding state and Port3/Switch 2 as an operational Designated port in forwarding state.

Next, Switch 1 is powered up. See the following figure.

FIGURE 61 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 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 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.

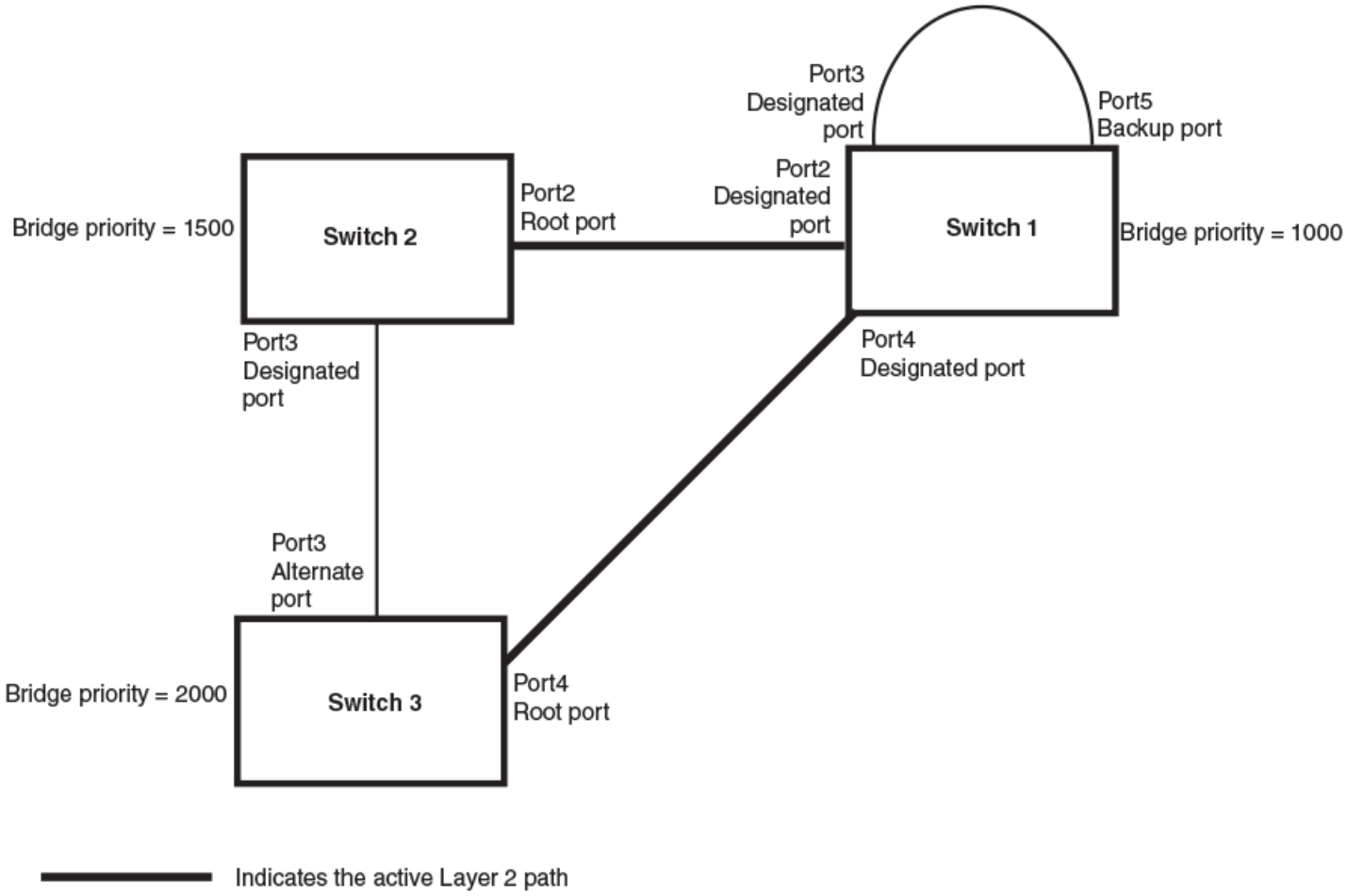
The Port2/Switch 2 bridge also sends an RST BPDU with an agreed flag Port2/Switch 1 that Port2 is the new Root port. Both ports go into forwarding states.

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 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.

Ports 3/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 62 Active Layer 2 path

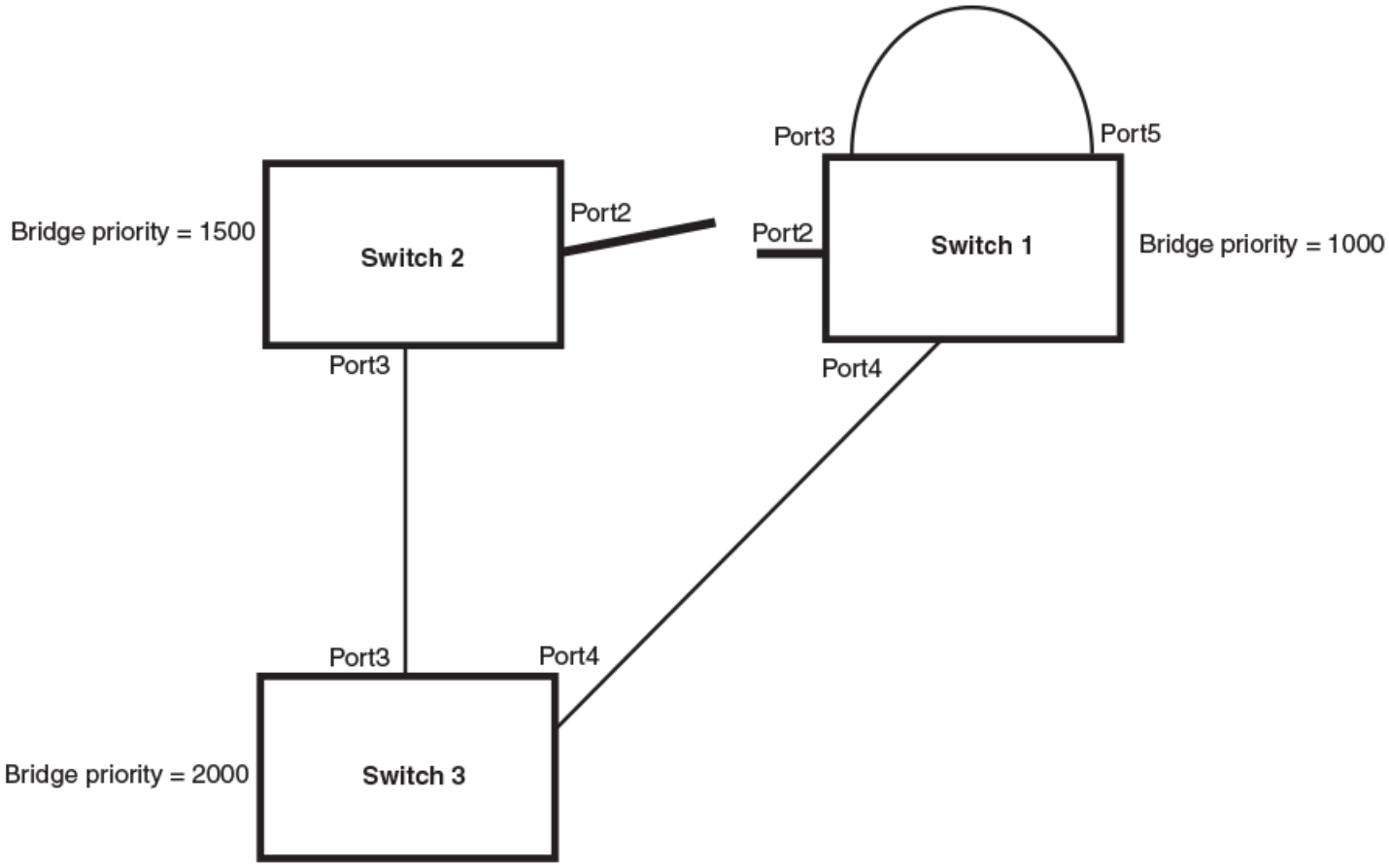


NOTE
Port numbers are simplified

Convergence after a link failure

The following figure illustrates a link failure in the 802.1W topology. In this example, Port2/Switch, 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 63 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, 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, 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, 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, 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.

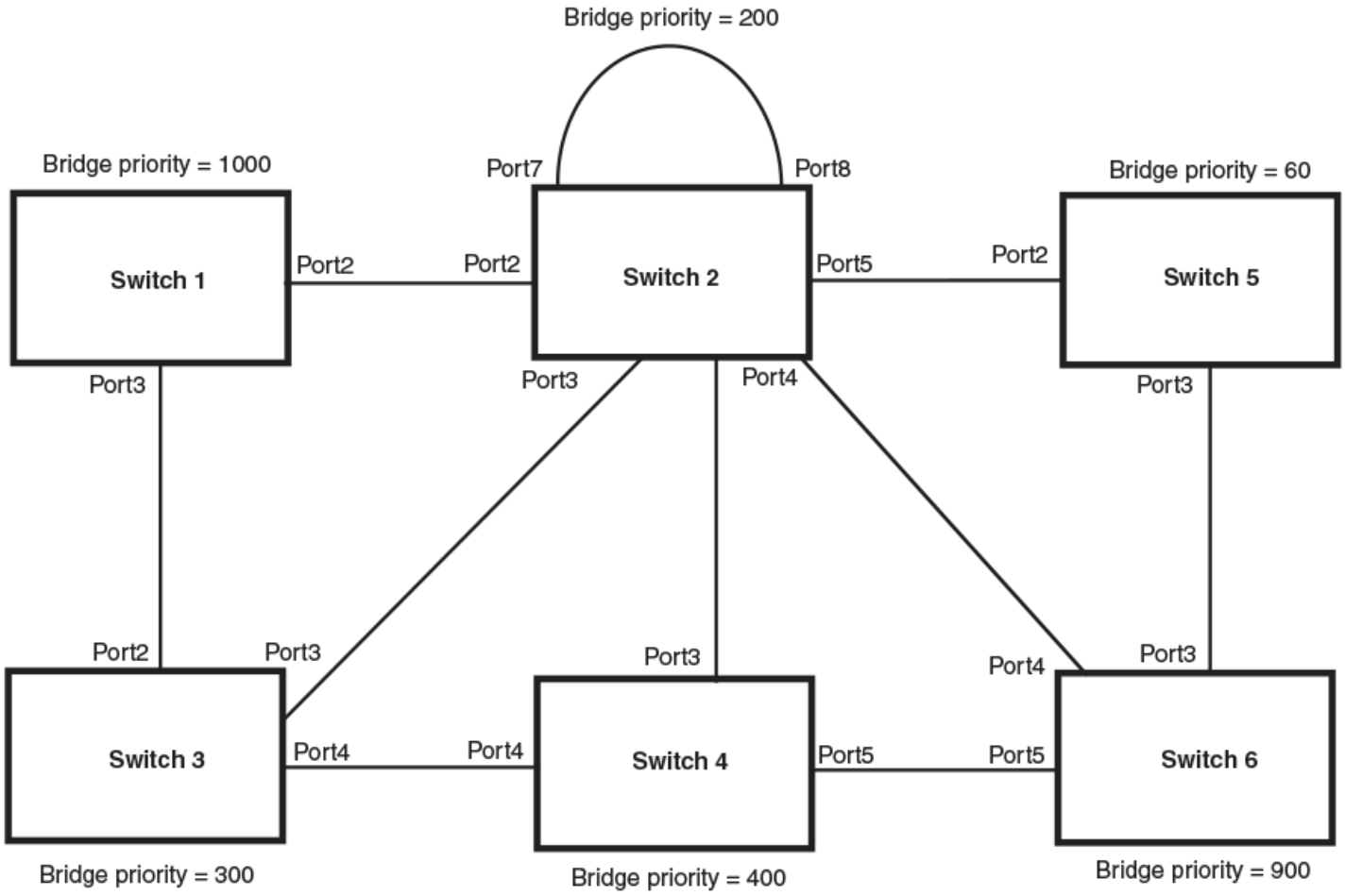
Now Port3/Switch 2 does not go into a forwarding state instantly like the Root port. It waits until the forward delay timer expires twice on that port while it is still in a Designated role, before it can proceed to the forwarding state. The wait, however, does not cause a denial of service, since the essential connectivity in the topology has already been established.

When fully restored, the topology is the same as that shown on Figure 61 on page 195.

Convergence in a complex 802.1W topology

The following figure illustrates a complex 802.1W topology.

FIGURE 64 Complex 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. 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.

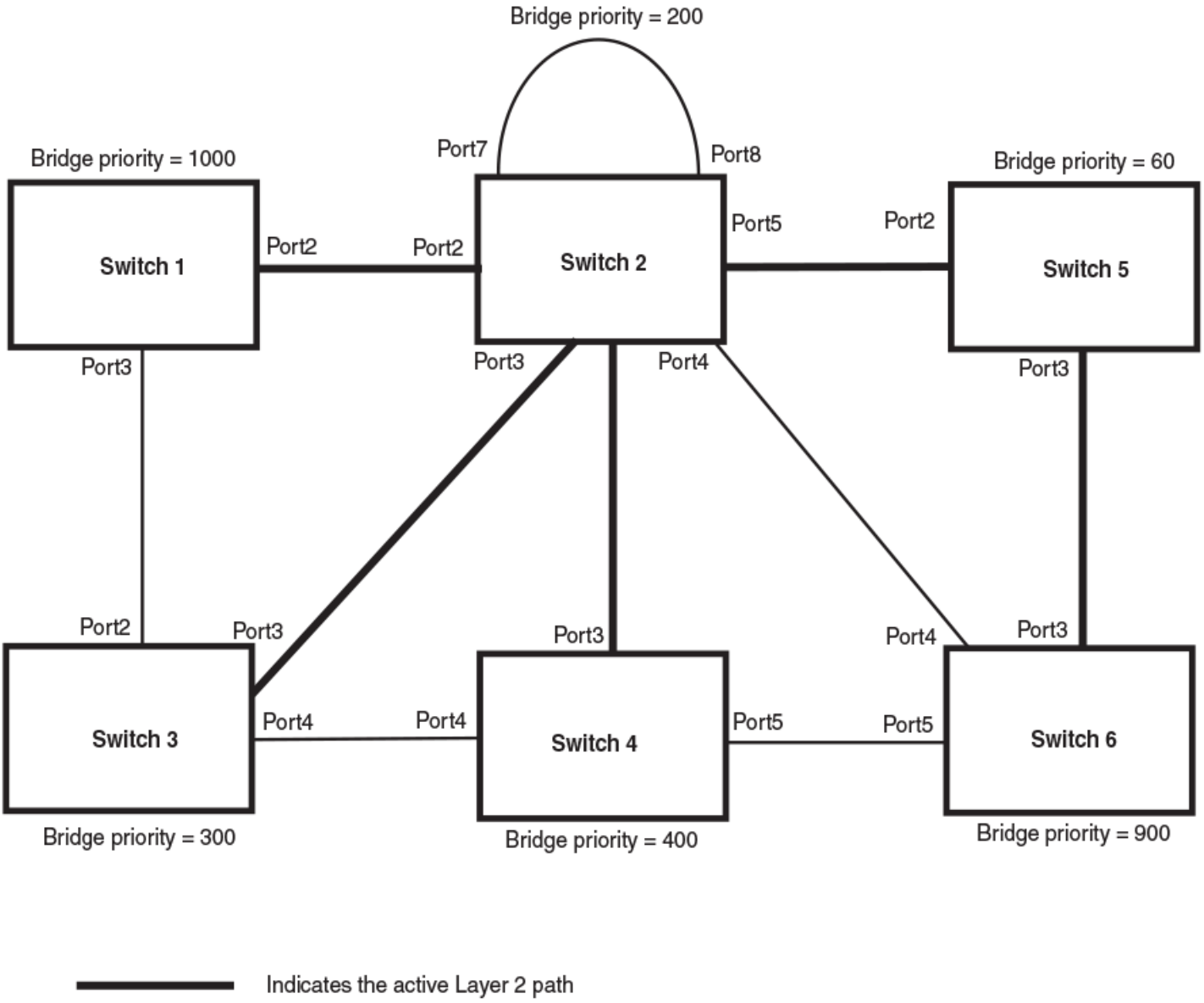
Port4/Switch 3 receives an RST BPDU that is inferior to what it can transmit; therefore, the port retains its Designated port role.

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 port. Only then will these port move into forwarding states.

The entire 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 in Figure 64.

FIGURE 65 Active Layer 2 path in complex topology



NOTE
Port numbers are simplified.

Propagation of topology change

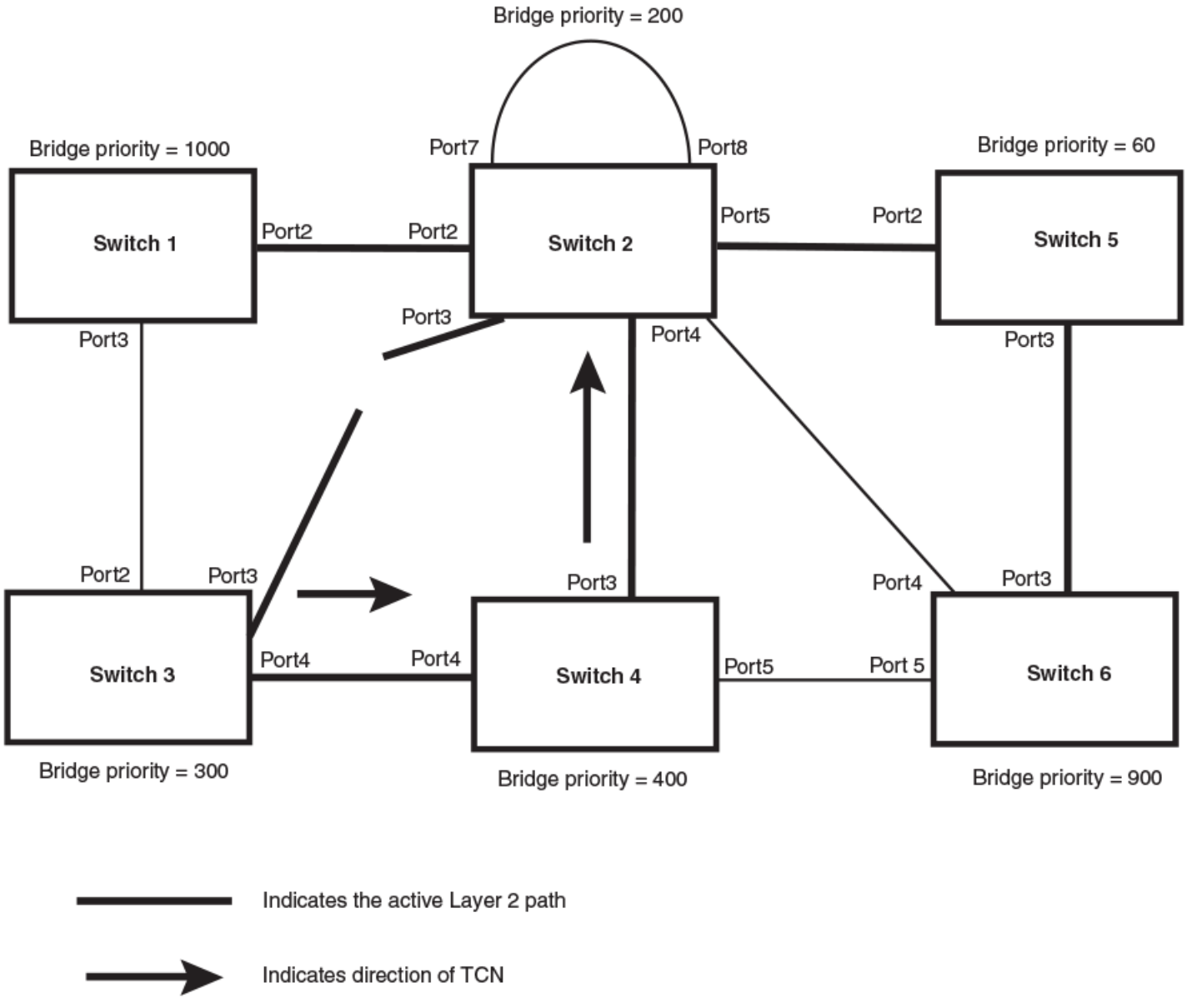
The Topology Change state machine generates and propagates the topology change notification messages on each port. When a Root port or a Designated port goes into a forwarding state, the Topology Change state machine on those ports send a topology change notice (TCN) to all the bridges in the topology 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, 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 66 Beginning of topology change notice

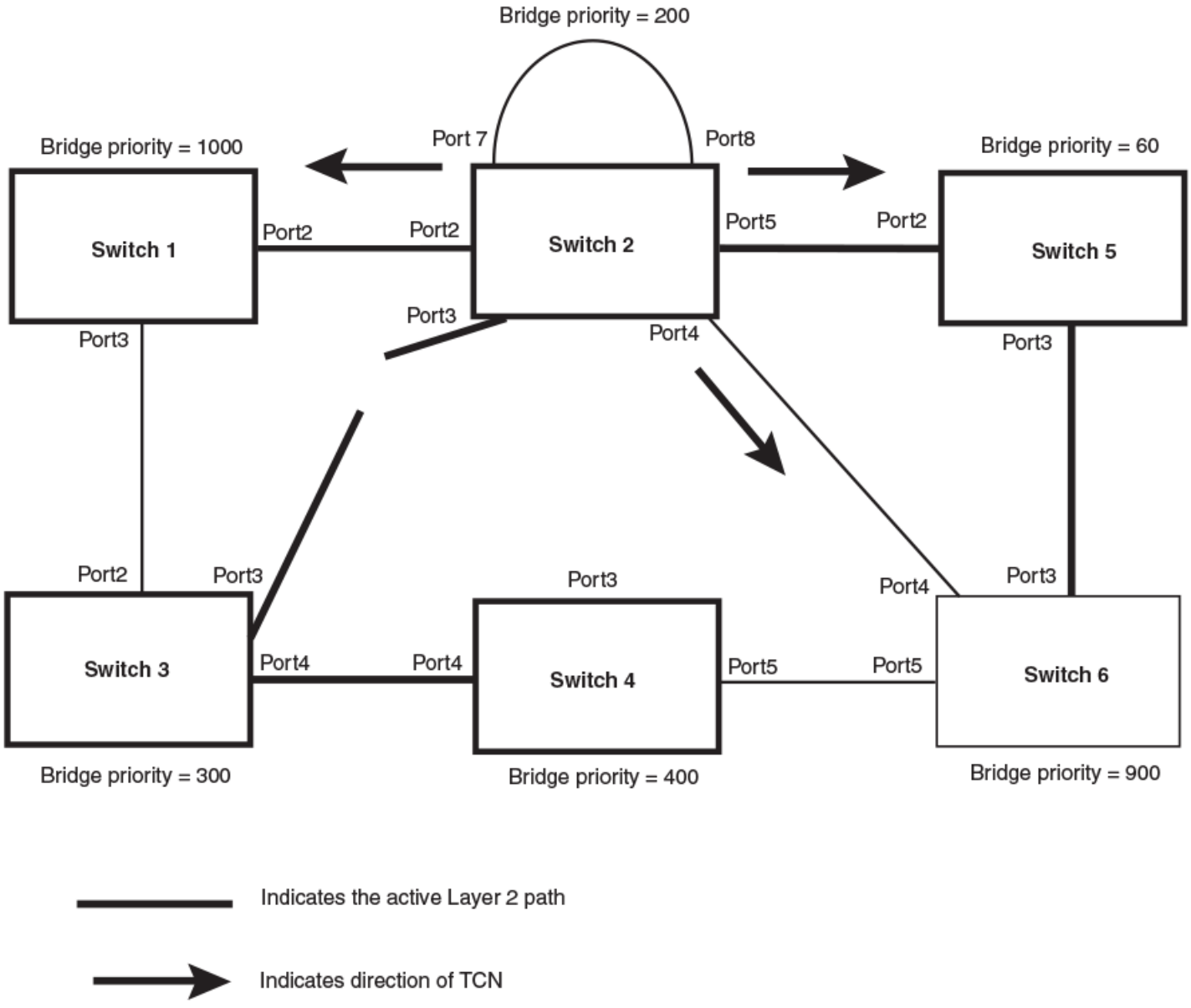


NOTE
Port numbers are simplified.

Switch 2 then starts the TCN timer on the Designated ports and sends RST BPDUs that contain the TCN as follows (Figure 67):

- 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

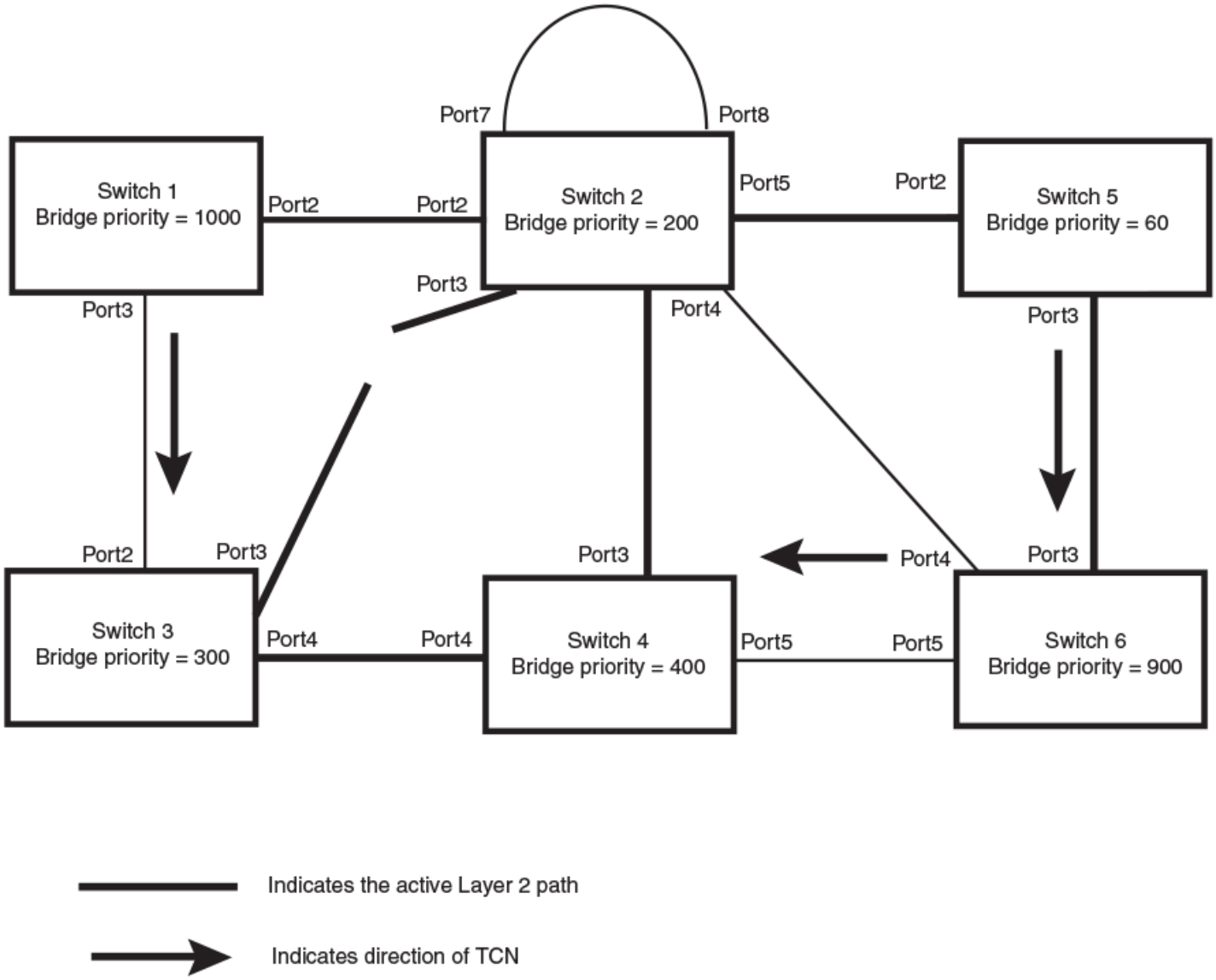
FIGURE 67 Sending TCN to bridges connected to Switch 2



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 68 Completing the TCN propagation



NOTE
Port numbers are simplified.

Compatibility of 802.1W with 802.1D

802.1W-enabled bridges are backward compatible with IEEE 802.1D bridges. This compatibility is managed on a per-port basis by the Port Migration state machine. **However, intermixing the two types of bridges in the network topology is not advisable if you want to take advantage of the rapid convergence feature.**

Compatibility with 802.1D means that an 802.1W-enabled port can send BPDUs in the STP or 802.1D format when one of the following events occur:

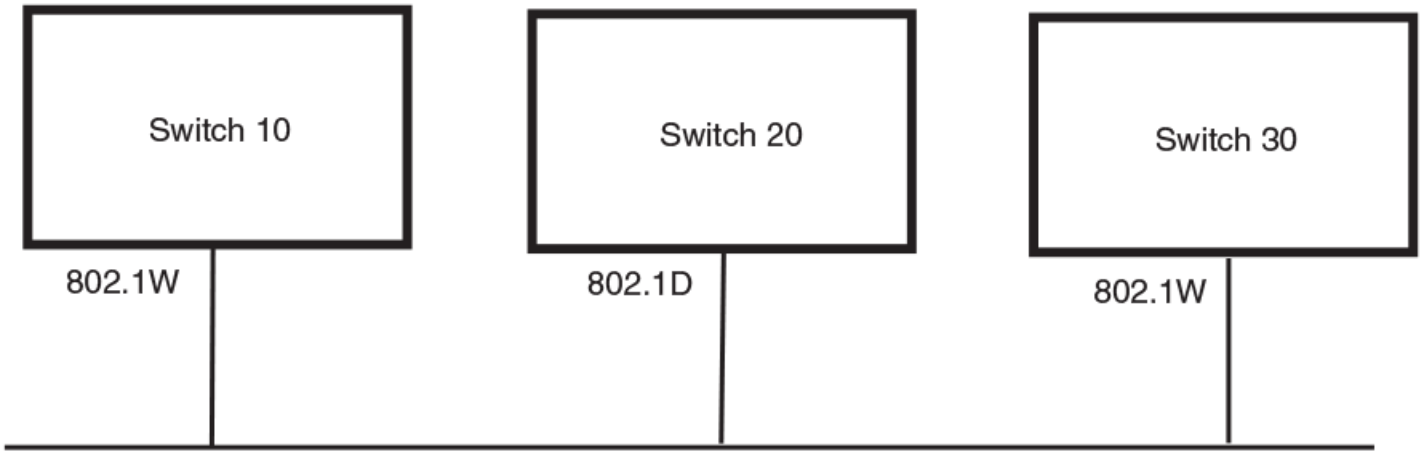
- The port receives a legacy BPDU. A legacy BPDU is an STP BPDU or a BPDU in an 802.1D format. The port that receives the legacy BPDU automatically configures itself to behave like a legacy port. It sends and receives legacy BPDUs only.

- The entire bridge is configured to operate in an 802.1D mode when an administrator sets the bridge parameter to zero at the CLI, forcing all ports on the bridge to send legacy BPDUs only.

Once a port operates in the 802.1D mode, 802.1D convergence times are used and rapid convergence is not realized.

For example, in the following figure, Switch 10 and Switch 30 receive legacy BPDUs from Switch 20. Ports on Switch 10 and Switch 30 begin sending BPDUs in STP format to allow them to operate transparently with Switch 20.

FIGURE 69 802.1W bridges with an 802.1D bridge



Once Switch 20 is removed from the LAN, Switch 10 and Switch 30 receive and transmit BPDUs in the STP format to and from each other. This state will continue until the administrator enables the **force-migration-check** command to force the bridge to send RSTP BPDU during a migrate time period. If ports on the bridges continue to hear only STP BPDUs after this migrate time period, those ports will return to sending STP BPDUs. However, when the ports receive RST BPDUs during the migrate time period, the ports begin sending RST BPDUs. The migrate time period is non-configurable. It has a value of three seconds.

NOTE
The IEEE standards state that 802.1W bridges need to interoperate with 802.1D bridges. IEEE standards set the path cost of 802.1W bridges to be between 1 and 200,000,000; whereas path cost of 802.1D bridges are set between 1 and 65,535. In order for the two bridge types to be able to interoperate in the same topology, the administrator needs to configure the bridge path cost appropriately. Path costs for either 802.1W bridges or 802.1D bridges need to be changed; in most cases, path costs for 802.1W bridges need to be changed.

Configuring 802.1W parameters on a Ruckus device

The remaining 802.1W sections explain how to configure the 802.1W protocol in a Ruckus device.

NOTE
With RSTP running, enabling static trunk on ports that are members of VLAN 4000 will keep the system busy for 20 to 25 seconds.

Ruckus devices are shipped from the factory with 802.1W disabled. Use the following methods to enable or disable 802.1W. You can enable or disable 802.1W at the following levels:

- Port-based VLAN - Affects all ports within the specified port-based VLAN. When you enable or disable 802.1W within a port-based VLAN, the setting overrides the global setting. Thus, you can enable 802.1W for the ports within a port-based

VLAN even when 802.1W is globally disabled, or disable the ports within a port-based VLAN when 802.1W is globally enabled.

- Individual port - Affects only the individual port. However, if you change the 802.1W state of the LAG virtual interface in a LAG, the change affects all ports in the LAG.

Enabling or disabling 802.1W in a port-based VLAN

Use the following procedure to disable or enable 802.1W on a device on which you have configured a port-based VLAN. Changing the 802.1W state in a VLAN affects only that VLAN.

To enable 802.1W for all ports in a port-based VLAN, enter commands such as the following.

```
device(config)#vlan 10
device(config-vlan-10)#spanning-tree 802-1w
```

Note regarding pasting 802.1W settings into the running configuration

If you paste 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 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 1001
end
```

To avoid this issue, 802.1W commands/settings that are pasted into the configuration should be in the following order.

1. Ports that are not yet connected
2. 802.1W RSTP settings
3. Ports that are already up

Example

```
configure terminal
vlan 120
untag ethernet 2/1/1
spanning-tree 802-1w
spanning-tree 802-1w priority 1001
tag ethernet 1/1/1 to 1/1/2
end
```

In the above configuration, untagged ethernet port 2/1/1 is added to VLAN 120 *before* the 802.1W RSTP settings, and ethernet ports 1/1/1 and 1/1/2 are added *after* the 802.1W RSTP settings. When these commands are pasted into the running configuration, the ports will properly operate in 802.1W RSTP mode.

Enabling or disabling 802.1W on a single spanning tree

To enable 802.1W for all ports of a single spanning tree, enter a command such as the following.

```
device(config-vlan-10)# spanning-tree single 802-1w
```

Disabling or enabling 802.1W on an individual port

The **spanning-tree 802-1w** or **spanning-tree single 802-1w** command must be used to initially enable 802.1W on ports. Both commands enable 802.1W on all ports that belong to the VLAN or to the single spanning tree.

Once 802.1W is enabled on a port, it can be disabled on individual ports. 802.1W that have been disabled on individual ports can then be enabled as required.

NOTE
If you change the 802.1W state of the LAG virtual interface in a LAG, the change affects all ports in that LAG.

To disable or enable 802.1W on an individual port, enter commands such as the following.

```
device(config)# interface ethernet 1/1/1
device(config-if-e1000-1/1/1)# no spanning-tree
```

Changing 802.1W bridge parameters

When you make changes to 802.1W bridge parameters, the changes are applied to individual ports on the bridge. To change 802.1W bridge parameters, use the following methods.

To designate a priority for a bridge, enter a command such as the following.

```
device(config)# spanning-tree 802-1w priority 10
```

The command in this example changes the priority on a device on which you have not configured port-based VLANs. The change applies to the default VLAN. If you have configured a port-based VLAN on the device, you can configure the parameters only at the configuration level for individual VLANs. Enter commands such as the following.

```
device(config)# vlan 20
device(config-vlan-20)# spanning-tree 802-1w priority 0
```

To make this change in the default VLAN, enter the following commands.

```
device(config)# vlan 1
device(config-vlan-1)# spanning-tree 802-1w priority 0
```

Changing port parameters

The 802.1W port commands can be enabled on individual ports or on multiple ports, such as all ports that belong to a VLAN.

The 802.1W port parameters are preconfigured with default values. If the default parameters meet your network requirements, no other action is required.

You can change the following 802.1W port parameters using the following method.

```
device(config)# vlan 10
device(config-vlan-10)# spanning-tree 802-1w ethernet 1/1/5 path-cost 15 priority 64
```

The **path-cost** *value* parameter specifies the cost of the port path to the root bridge. 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 19 Recommended path cost values of 802.1W

Link speed	Recommended (Default) 802.1W path cost values	Recommended 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

TABLE 19 Recommended path cost values of 802.1W (continued)

Link speed	Recommended (Default) 802.1W path cost values	Recommended 802.1W patch cost range
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

Example

Suppose you want to enable 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 information about 802-1w

To display a summary of 802-1w, use the following 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 802-1W, enter the show 802-1w detail command.

```
device# show 802-1w detail
========================================================================
VLAN 1 - MULTIPLE SPANNING TREE (MSTP - IEEE 802.1W) ACTIVE
========================================================================
BridgeId 800000e080541700, forceVersion 2, txHoldCount 3
Port 1/1/1 - Role: ROOT - State: FORWARDING
  PathCost 200000, Priority 128, AdminOperEdge F, AdminPt2PtMac F
  DesignatedPriority - Root: 0x800000e0804c9c00, Bridge: 0x800000e080541700
  ActiveTimers - rrWhile 4 rcvdInfoWhile 4
  MachineStates - PIM: CURRENT, PRT: ROOT_PORT, PST: FORWARDING
  TCM: ACTIVE, PPM: SENDING_STP, PTX: TRANSMIT_IDLE
  Received - RST BPDUs 0, Config BPDUs 1017, TCN BPDUs 0
Port 1/1/2 - Role: DESIGNATED - State: FORWARDING
  PathCost 200000, Priority 128, AdminOperEdge F, AdminPt2PtMac F
```

```
DesignatedPriority - Root: 0x800000e0804c9c00, Bridge: 0x800000e080541700
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
```

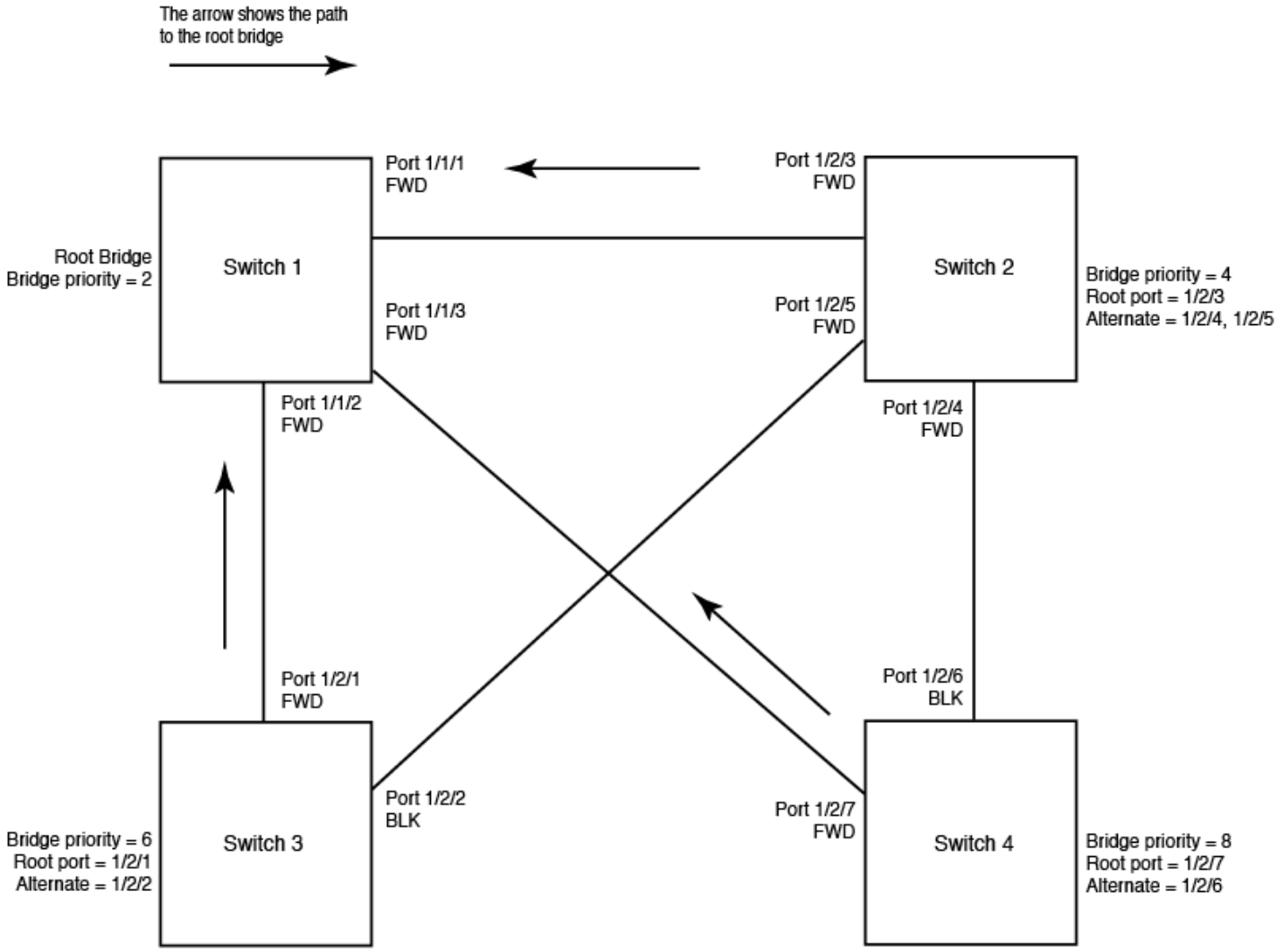
802.1W Draft 3

As an alternative to full 802.1W, you can configure 802.1W Draft 3. 802.1W Draft 3 provides a subset of the RSTP capabilities described in the 802.1W STP specification.

802.1W Draft 3 support is disabled by default. When the feature is enabled, if a root port on a Ruckus device that is not the root bridge becomes unavailable, the device can automatically Switch over to an alternate root port, without reconvergence delays. 802.1W Draft 3 does not apply to the root bridge, since all the root bridge ports are always in the forwarding state.

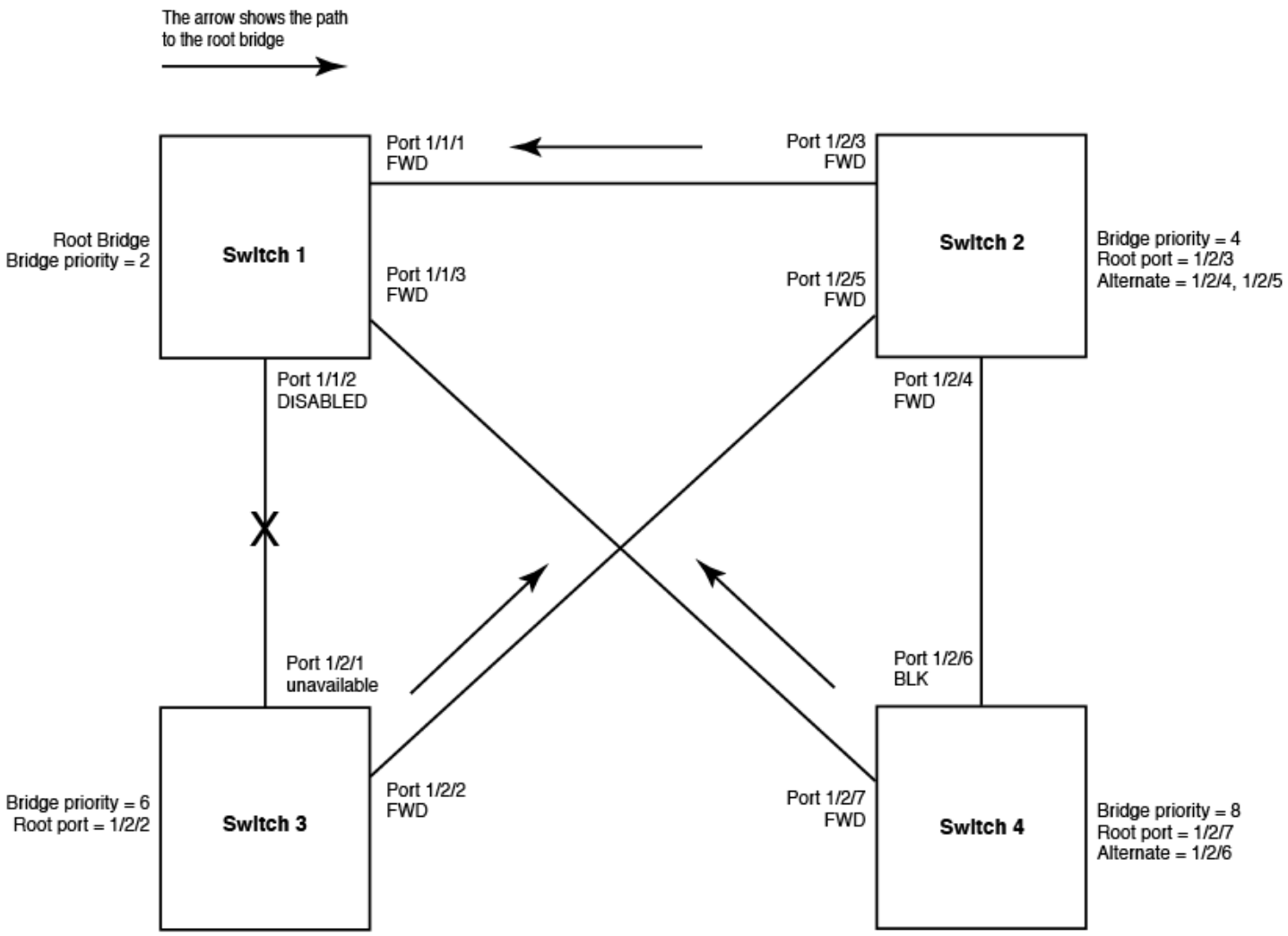
The following figure shows an example of an optimal STP topology. In this topology, all the non-root bridges have at least two paths to the root bridge (Switch 1 in this example). One of the paths is through the root port. The other path is a backup and is through the alternate port. While the root port is in the forwarding state, the alternate port is in the blocking state.

FIGURE 70 802.1W Draft 3 RSTP ready for failover



If the root port on a Switch becomes unavailable, 802.1W Draft 3 immediately fails over to the alternate port, as shown in the following figure.

FIGURE 71 802.1W Draft 3 RSTP failover to alternate root port



In this example, port 1/2/1 on Switch 3 has become unavailable. In standard STP (802.1D), if the root port becomes unavailable, the Switch must go through the listening and learning stages on the alternate port to reconverge with the spanning tree. Thus, port 1/2/2 must go through the listening and learning states before entering the forwarding state and thus reconverging with the spanning tree.

802.1W Draft 3 avoids the reconvergence delay by calculating an alternate root port, and immediately failing over to the alternate port if the root port becomes unavailable. The alternate port is in the blocking state as long as the root port is in the forwarding state, but moves immediately to the active state if the root port becomes unavailable. Thus, using 802.1W Draft 3, Switch 3 immediately fails over to port 1/2/2, without the delays caused by the listening and learning states.

802.1W Draft 3 selects the port with the next-best cost to the root bridge. For example, on Switch 3, port 1/2/1 has the best cost to the root bridge and thus is selected by STP as the root port. Port 1/2/2 has the next-best cost to the root bridge, and thus is selected by 802.1W Draft 3 as the alternate path to the root bridge.

Once a failover occurs, the Switch no longer has an alternate root port. If the port that was an alternate port but became the root port fails, standard STP is used to reconverge with the network. You can minimize the reconvergence delay in this case by setting

the forwarding delay on the root bridge to a lower value. For example, if the forwarding delay is set to 15 seconds (the default), change the forwarding delay to a value from 3 - 10 seconds.

During failover, 802.1W Draft 3 flushes the MAC addresses leaned on the unavailable root port, selects the alternate port as the new root port, and places that port in the forwarding state. If traffic is flowing in both directions on the new root port, addresses are flushed (moved) in the rest of the spanning tree automatically.

Spanning tree reconvergence time

Spanning tree reconvergence using 802.1W Draft 3 can occur within one second.

After the spanning tree reconverges following the topology change, traffic also must reconverge on all the bridges attached to the spanning tree. This is true regardless of whether 802.1W Draft 3 or standard STP is used to reconverge the spanning tree.

Traffic reconvergence happens after the spanning tree reconvergence, and is achieved by flushing the Layer 2 information on the bridges:

- Following 802.1W Draft 3 reconvergence of the spanning tree, traffic reconvergence occurs in the time it takes for the bridge to detect the link changes plus the STP maximum age set on the bridge.
- If standard STP reconvergence occurs instead, traffic reconvergence takes two times the forward delay plus the maximum age.

  NOTE
  802.1W Draft 3 does not apply when a failed root port comes back up. When this happens, standard STP is used.

802.1w configuration considerations

802.1w Draft 3 is disabled by default. To ensure optimal performance of the feature before you enable it, do the following:

- Configure the bridge priorities so that the root bridge is one that supports 802.1w Draft 3. (Use a Ruckus device or third-party device that supports 802.1w Draft 3.)
- Change the forwarding delay on the root bridge to a value lower than the default 15 seconds. Ruckus recommends a value from 3 - 10 seconds. The lower forwarding delay helps reduce reconvergence delays in cases where 802.1w Draft 3 is not applicable, such as when a failed root port comes back up.
- Configure the bridge priorities and root port costs so that each device has an active path to the root bridge if its root port becomes unavailable. For example, port 1/2/2 is connected to port 1/2/5 on Switch 2, which has the second most favorable bridge priority in the spanning tree.

  NOTE
  If reconvergence involves changing the state of a root port on a bridge that supports 802.1d STP but not 802.1w Draft 3, then reconvergence still requires the amount of time it takes for the ports on the 802.1d bridge to change state to forwarding (as needed), and receive BPDUs from the root bridge for the new topology.

Enabling 802.1w Draft 3

802.1w Draft 3 is disabled by default. The procedure for enabling the feature differs depending on whether single STP is enabled on the device.

NOTE
STP must be enabled before you can enable 802.1w Draft 3.

Enabling 802.1w Draft 3 when single STP is not enabled

By default, each port-based VLAN on the device has its own spanning tree. To enable 802.1w Draft 3 in a port-based VLAN, enter commands such as the following.

```
device(config)# vlan 10
device(config-vlan-10)# spanning-tree rstp
```

This command enables 802.1w Draft 3. You must enter the command separately in each port-based VLAN in which you want to run 802.1w Draft 3.

> **NOTE**
> This command does not also enable STP. To enable STP, first enter the **spanning-tree** command without the **rstp** parameter. After you enable STP, enter the **spanning-tree rstp** command to enable 802.1w Draft 3.

To disable 802.1w Draft 3, enter the following command.

```
device(config-vlan-10)# no spanning-tree rstp
```

Enabling 802.1w Draft 3 when single STP is enabled

To enable 802.1w Draft 3 on a device that is running single STP, enter the following command at the global CONFIG level of the CLI.

```
device(config)# spanning-tree single rstp
```

This command enables 802.1w Draft 3 on the whole device.

> **NOTE**
> This command does not also enable single STP. To enable single STP, first enter the **spanning-tree single** command without the **rstp** parameter. After you enable single STP, enter the **spanning-tree single rstp** command to enable 802.1w Draft 3.

To disable 802.1w Draft 3 on a device that is running single STP, enter the following command.

```
device(config)# no spanning-tree single rstp
```

Single Spanning Tree (SSTP)

By default, each port-based VLAN on a Ruckus device runs a separate spanning tree, which you can enable or disable on an individual VLAN basis.

Alternatively, you can configure a Ruckus device to run a single spanning tree across all ports and VLANs on the device. The Single STP feature (SSTP) is especially useful for connecting a Ruckus device to third-party devices that run a single spanning tree in accordance with the 802.1Q specification.

SSTP uses the same parameters, with the same value ranges and defaults, as the default STP support on Ruckus devices. Refer to STP parameters and defaults on page 159.

SSTP defaults

SSTP is disabled by default. When you enable the feature, all VLANs on which STP is enabled become members of a single spanning tree. All VLANs on which STP is disabled are excluded from the single spanning tree.

To add a VLAN to the single spanning tree, enable STP on that VLAN.To remove a VLAN from the single spanning tree, disable STP on that VLAN.

When you enable SSTP, all the ports that are in port-based VLANs with STP enabled become members of a single spanning tree domain. Thus, the ports share a single BPDU broadcast domain. The Ruckus device places all the ports in a non-configurable VLAN, 4094, to implement the SSTP domain. However, this VLAN does not affect port membership in the port-based VLANs you have configured. Other broadcast traffic is still contained within the individual port-based VLANs. Therefore, you can use SSTP while still using your existing VLAN configurations without changing your network. In addition, SSTP does not affect 802.1Q tagging. Tagged and untagged ports alike can be members of the single spanning tree domain.

NOTE
When SSTP is enabled, the BPDUs on tagged ports go out untagged.

If you disable SSTP, all VLANs that were members of the single spanning tree run MSTP instead. In MSTP, each VLAN has its own spanning tree. VLANs that were not members of the single spanning tree were not enabled for STP. Therefore, STP remains disabled on those VLANs.

Enabling SSTP

To enable SSTP, use one of the following methods.

NOTE
If the device has only one port-based VLAN (the default VLAN), then the device is already running a single instance of STP. In this case, you do not need to enable SSTP. You need to enable SSTP only if the device contains more than one port-based VLAN and you want all the ports to be in the same STP broadcast domain.

To configure the Ruckus device to run a single spanning tree, enter the following command at the global CONFIG level.

```
device(config)# spanning-tree single
```

NOTE
If the device has only one port-based VLAN, the CLI command for enabling SSTP is not listed in the CLI. The command is listed only if you have configured a port-based VLAN.

To change a global STP parameter, enter a command such as the following at the global CONFIG level.

```
device(config)# spanning-tree single priority 2
```

This command changes the STP priority for all ports to 2.

To change an STP parameter for a specific port, enter commands such as the following.

```
device(config)# spanning-tree single ethernet 1/2/1 priority 10
```

The commands shown above override the global setting for STP priority and set the priority to 10 for port 1/2/1.

NOTE
Both commands listed above are entered in the global configuration mode.

Displaying SSTP information

To verify that SSTP is in effect, enter the following commands at any level of the CLI.

```
device# show span
```

The **detail** parameter and its additional optional parameters display detailed information for individual ports. Refer to Displaying detailed STP information for each interface on page 166.

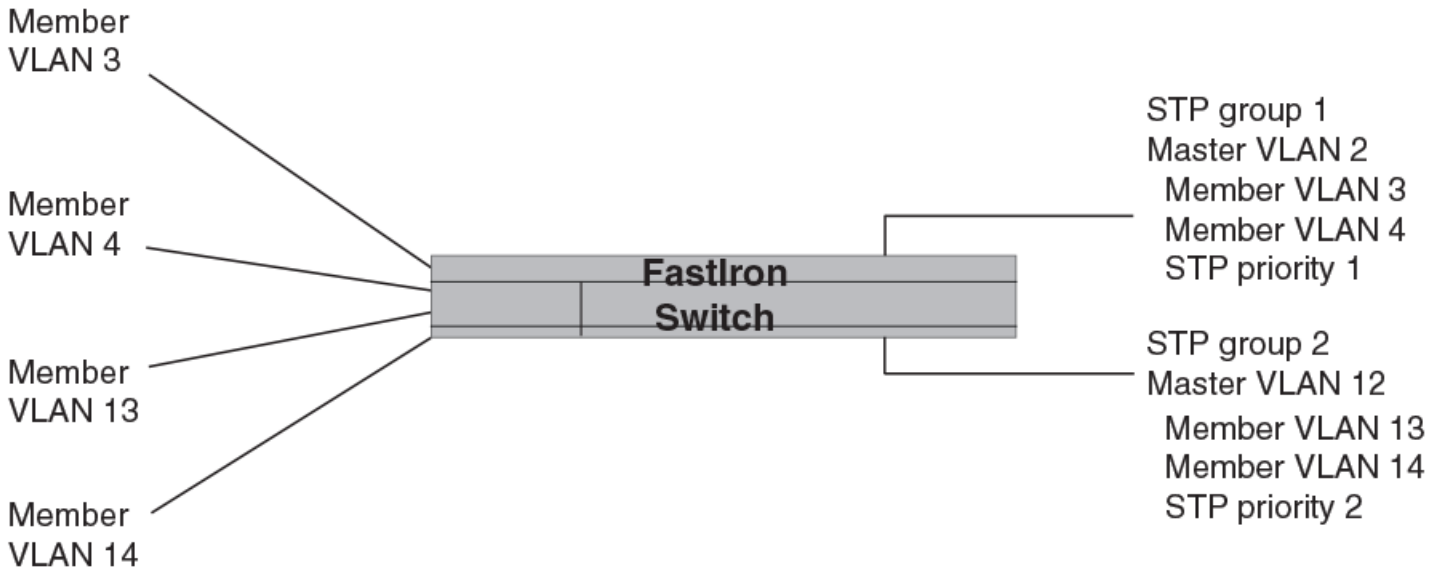
STP per VLAN group

STP per VLAN group is an STP enhancement that provides scalability while overcoming the limitations of the following scalability alternatives:

- Standard STP - You can configure up to 254 instances of standard STP on a Ruckus device. However, on Ruckus ICX 7150 device only 253 STP instances are supported. More instances of STP may be required in large configurations. Using STP per VLAN group, you can aggregate STP instances.
- Single STP - Single STP allows all the VLANs to run STP, but each VLAN runs the same instance of STP, resulting in numerous blocked ports that do not pass any Layer 2 traffic. STP per VLAN group uses all available links by load balancing traffic for different instances of STP on different ports. A port that blocks traffic for one spanning tree forwards traffic for another spanning tree.

STP per VLAN group allows you to group VLANs and apply the same STP parameter settings to all the VLANs in the group. The following figure shows an example of a STP per VLAN group implementation.

FIGURE 72 STP per VLAN group example



A master VLAN contains one or more member VLANs. Each of the member VLANs in the STP Group runs the same instance of STP and uses the STP parameters configured for the master VLAN. In this example, the FastIron switch is configured with VLANs 3, 4, 13, and 14. VLANs 3 and 4 are grouped in master VLAN 2, which is in STP group 1. VLANs 13 and 14 are grouped in master VLAN 12, which is in STP group 2. The VLANs in STP group 1 all share the same spanning tree. The VLANs in STP group 2 share a different spanning tree.

All the ports are tagged. The ports must be tagged so that they can be in both a member VLAN and the member's master VLAN. For example, ports 1/1/1 - 1/1/4 are in member VLAN 3 and also in master VLAN 2 (since master VLAN 2 contains member VLAN 3).

STP load balancing

Notice that the STP groups each have different STP priorities. In configurations that use the STP groups on multiple devices, you can use the STP priorities to load balance the STP traffic. By setting the STP priorities for the same STP group to different values on each device, you can cause each of the devices to be the root bridge for a different STP group. This type of configuration distributes the traffic evenly across the devices and also ensures that ports that are blocked in one STP group spanning tree are used by another STP group spanning tree for forwarding. Refer to Configuration example for STP load sharing on page 217 for an example using STP load sharing.

Configuring STP per VLAN group

To configure STP per VLAN group, perform the following tasks:

1. Configure the member VLANs.
2. Optionally, configure master VLANs to contain the member VLANs. This is useful when you have a lot of member VLANs and you do not want to individually configure STP on each one. Each of the member VLANs in the STP group uses the STP settings of the master VLAN.
3. Configure the STP groups. Each STP group runs a separate instance of STP.

   The following CLI commands implement the STP per VLAN group configuration shown in Figure 72 on page 216. The following commands configure the member VLANs (3, 4, 13, and 14) and the master VLANs (2 and 12). Notice that changes to STP parameters are made in the master VLANs only, not in the member VLANs.

```
device(config)# vlan 2
device(config-vlan-2)# spanning-tree priority 1
device(config-vlan-2)# tagged ethernet 1/1/1 to 1/1/4
device(config-vlan-2)# vlan 3
device(config-vlan-3)# tagged ethernet 1/1/1 to 1/1/4
device(config-vlan-3)# vlan 4
device(config-vlan-4)# tagged ethernet 1/1/1 to 1/1/4
device(config-vlan-4)# vlan 12
device(config-vlan-12)# spanning-tree priority 2
device(config-vlan-12)# tagged ethernet 1/1/1 to 1/1/4
device(config-vlan-12)# vlan 13
device(config-vlan-13)# tagged ethernet 1/1/1 to 1/1/4
device(config-vlan-13)# vlan 14
device(config-vlan-14)# tagged ethernet 1/1/1 to 1/1/4
device(config-vlan-14)# exit
```

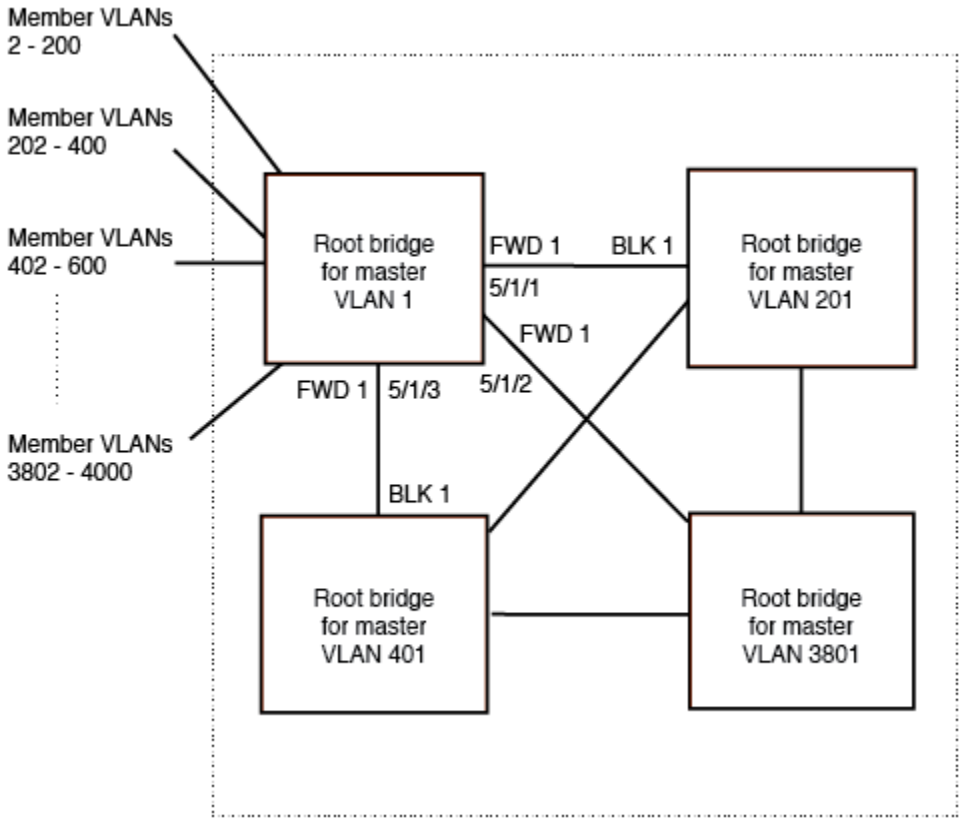
   The following commands configure the STP groups.

```
device(config)# stp-group 1
device(config-stp-group-1)# master-vlan 2
device(config-stp-group-1)# member-vlan 3 to 4
device(config-stp-group-1)# exit
device(config)#stp-group 2
device(config-stp-group-2)# master-vlan 12
device(config-stp-group-2)# member-vlan 13 to 14
```

Configuration example for STP load sharing

The following figure shows another example of a STP per VLAN group implementation.

FIGURE 73 More complex STP per VLAN group example



In this example, each of the devices in the core is configured with a common set of master VLANs, each of which contains one or more member VLANs. Each of the member VLANs in an STP group runs the same instance of STP and uses the STP parameters configured for the master VLAN.

The STP group ID identifies the STP instance. All VLANs within an STP group run the same instance of STP. The master VLAN specifies the bridge STP parameters for the STP group, including the bridge priority. In this example, each of the devices in the core is configured to be the default root bridge for a different master VLAN. This configuration ensures that each link can be used for forwarding some traffic. For example, all the ports on the root bridge for master VLAN 1 are configured to forward BPDUs for master VLAN spanning tree. Ports on the other devices block or forward VLAN 1 traffic based on STP convergence. All the ports on the root bridge for VLAN 2 forward VLAN 2 traffic, and so on.

All the ports are tagged. The ports must be tagged so that they can be in both a member VLAN and the member's master VLAN. For example, port 1/1/1 - and ports 5/1/1, 5/1/2, and 5/1/3 are in member VLAN 2 and master VLAN 1 (since master VLAN a contains member VLAN 2).

Here are the commands for configuring the root bridge for master VLAN 1 in figure Figure 72 on page 216 for STP per VLAN group. The first group of commands configures the master VLANs. Notice that the STP priority is set to a different value for each VLAN. In addition, the same VLAN has a different STP priority on each device. This provides load balancing by making each of the devices a root bridge for a different spanning tree.

```
device(config)# vlan 1
device(config-vlan-1)# spanning-tree priority 1
device(config-vlan-1)# tag ethernet 1/1/1 ethernet 5/1/1 to 5/1/3
device(config-vlan-1)# vlan 201
device(config-vlan-201)# spanning-tree priority 2
```

```
device(config-vlan-201)# tag ethernet 1/1/2 ethernet 5/1/1 to 5/1/3
device(config-vlan-201)# vlan 401
device(config-vlan-401)# spanning-tree priority 3
device(config-vlan-401)# tag ethernet 1/1/3 ethernet 5/1/1 to 5/1/3
...
device(config-vlan-3601)# vlan 3801
device(config-vlan-3801)# spanning-tree priority 20
device(config-vlan-3801)# tag ethernet 1/1/20 ethernet 5/1/1 to 5/1/3
device(config-vlan-3801)# exit
```

The next group of commands configures VLAN groups for the member VLANs. Notice that the VLAN groups do not contain the VLAN numbers assigned to the master VLANs. Also notice that no STP parameters are configured for the groups of member VLANs. Each group of member VLANs will inherit its STP settings from its master VLAN.

Set the bridge priority for each master VLAN to the highest priority (1) on one of the devices in the STP per VLAN group configuration. By setting the bridge priority to the highest priority, you make the device the default root bridge for the spanning tree. To ensure STP load balancing, make each of the devices the default root bridge for a different master VLAN.

```
device(config)# vlan-group 1 vlan 2 to 200
device(config-vlan-group-1)# tag ethernet 1/1/1 ethernet 5/1/1 to 5/1/3
device(config-vlan-group-1)# vlan-group 2 vlan 202 to 400
device(config-vlan-group-2)# tag ethernet 1/1/2 ethernet 5/1/1 to 5/1/3
device(config-vlan-group-2)# vlan-group 3 vlan 402 to 600
device(config-vlan-group-2)# tag ethernet 1/1/3 ethernet 5/1/1 to 5/1/3
...
device(config-vlan-group-19)# vlan-group 20 vlan 3082 to 3282
device(config-vlan-group-20)# tag ethernet 1/1/20 ethernet 5/1/1 to 5/1/3
device(config-vlan-group-20)# exit
```

The following group of commands configures the STP groups. Each STP group in this configuration contains one master VLAN, which contains a VLAN group. This example shows that an STP group also can contain additional VLANs (VLANs not configured in a VLAN group).

```
device(config)# stp-group 1
device(config-stp-group-1)# master-vlan 1
device(config-stp-group-1)# member-group 1
device(config-stp-group-1)# member-vlan 4001 4004 to 4010
device(config-stp-group-1)# stp-group 2
device(config-stp-group-2)# master-vlan 201
device(config-stp-group-2)# member-group 2
device(config-stp-group-2)# member-vlan 4002 4003 4011 to 4015
device(config-stp-group-2)# stp-group 3
device(config-stp-group-3)# master-vlan 401
device(config-stp-group-3)  #member-group 3
...
device(config-stp-group-19)# stp-group 20
device(config-stp-group-20)# master-vlan 3081
device(config-stp-group-20)# member-group 20
```

Spanning Tree Path Cost Support for 802.1D 2004

Support for the IEEE 802.1D 2004 defined set of spanning tree path costs can be configured globally for high-speed ports.

To ensure a loop-free topology, Spanning Tree Protocol (STP) determines a single active path between two network nodes by calculating a path cost from various properties, including the bridge ID, bridge priority, network segment cost, and port priority. The original IEEE 802.1D 1998 standard allows for a 16-bit path cost value. The introduction of higher port speeds such as 40 Gbps up to 10 Tbps led to a new IEEE 802.1D 2004 standard that allows a 32-bit path cost value for more granularity in assigning path costs using a redefined path cost formula.

STP Path Cost = 20,000,000,000/Link speed in Kbps

The challenge is to have all bridge devices in the network running the same set of path costs. A command line interface (CLI) command, **spanning-tree path-cost-method** with two keyword options of **long** or **short**, has been created to globally change

the STP path cost method for all ports running STP. The 802.1D 1998 set of path cost options is the default setting and is referred to as the **short** option. When configured, the **long** option upgrades all ports running STP to use the 802.1D 2004 defined set of spanning tree path costs. To revert to the 802.1D 1998 set of path cost options, use the **short** option in global configuration mode.

NOTE
Any path cost configured for an individual interface is overwritten when the STP path costs are configured globally.

NOTE
LAG formation follows the legacy rule where aggregation of ports is allowed only for ports with the same default speed and media type.

Configuring Spanning Tree Path Cost Support for 802.1D 2004

Devices running Spanning Tree Protocol (STP) with higher speed ports (above 40 Gbps) can be configured to use more granular path costs as outlined by the 802.1D 2004 standards. To avoid contention with ports running STP path costs using the 802.1D 1998 standards, a CLI command is entered in global configuration mode to force all ports running STP to use the 802.1D 2004 path costs.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Configure all ports running STP globally to use the 802.1D 2004 path costs.

```
device(config)# spanning-tree path-cost-method long
```

The following example configures all ports running STP globally to use the 802.1D 2004 path cost method.

```
device# configure terminal
device(config)# spanning-tree path-cost-method long
```

Displaying Spanning Tree 802.1D 2004 Path Cost Information

After the global configuration of the Spanning Tree 802.1D 2004 path cost method, you can display the defined path cost for high-speed ports.

Before performing this task, you must configure all ports running Spanning Tree Protocol (STP) globally to use the 802.1D 2004 path costs.

Use the **show span** command to display the port path costs after the 802.1D 2004 path cost method is configured globally. The Ethernet 1/1/5 and 1/1/6 port speeds are 1 Gbps.

```
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 8000cc4e246eb200 0    Root      8000 20  2   1   15  12739   2   cc4e246eb200

Port STP Parameters:

Port      Prio Path      State       Fwd    Design  Designated       Designated
Num       rity Cost                  Trans  Cost    Root             Bridge
          Hex
```

```
1/1/5     80   20000     FORWARDING  1       0       8000cc4e246eb200 8000cc4e246eb200
1/1/6     80   20000     FORWARDING  1       0       8000cc4e246eb200 8000cc4e246eb200
```

PVST/PVST+ compatibility

The FastIron family of switches support Cisco's Per VLAN Spanning Tree plus (PVST+), by allowing the device to run multiple spanning trees while also interoperating with IEEE 802.1Q devices[1].

> **NOTE**
> Ruckus 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. However, to support the IEEE 802.1Q BPDUs, you might need to enable dual-mode support.

Support for Cisco's Per VLAN Spanning Tree plus (PVST+), allows a Ruckus device to run multiple spanning trees (multiple spanning trees) while also interoperating with IEEE 802.1Q devices. Ruckus 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.

1 Cisco user documentation for PVST/PVST+ refers to the IEEE 802.1Q spanning tree as the Common Spanning Tree (CST).

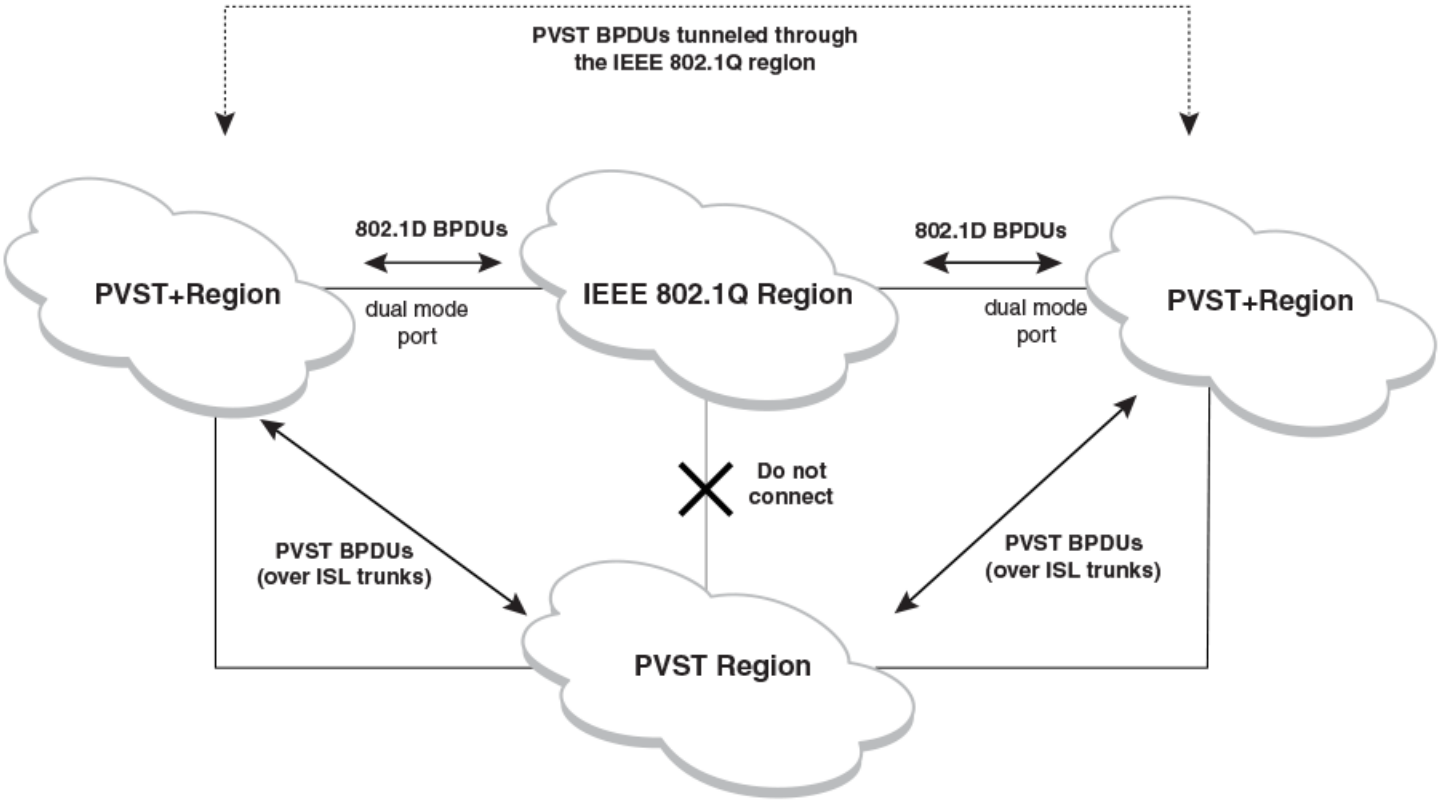
Overview of PVST and PVST+

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 tunnelled through 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.

FIGURE 74 Interaction of IEEE 802.1Q, PVST, and PVST+ regions



VLAN tags and dual mode

The dual-mode feature enables a port to send and receive both tagged and untagged frames. When the dual-mode feature is enabled on a port, the port is an untagged member of one of its VLANs and is at the same time a tagged member of all its other VLANs. The untagged frames are supported on the port Port Native VLAN .

The dual-mode feature must be enabled on a Ruckus port in order to interoperate with another vendor device. Some vendors use VLAN 1 by default to support the IEEE 802.1Q-based standard spanning tree protocols, such as 802.1d and 802.1w for sending untagged frames on VLAN 1. On Ruckus switches, by default, the Port Native VLAN is the same as the Default VLAN , which is VLAN 1. Thus, to support IEEE 802.1Q in a typical configuration, a port must be able to send and receive untagged frames for VLAN 1 and tagged frames for the other VLANs, and interoperate with other vendor devices using VLAN 1.

If you want to use tagged frames on VLAN 1, you can change the default VLAN ID to an ID other than 1. You also can specify the VLAN on which you want the port to send and receive untagged frames (the Port Native VLAN). The Port Native VLAN ID does not need to be the same as the default VLAN. Make sure that the untagged (native) VLAN is also changed on the interoperating vendor side to match that on the Ruckus side.

To support the IEEE 802.1Q with non-standard proprietary protocols such as PVST and PVST+, a port must always send and receive untagged frames on VLAN 1 on both sides. In this case, enable the dual-mode 1 feature to allow untagged BPDUs on VLAN 1and use Native VLAN 1 on the interoperating vendor side. You should not use VLAN 1 for tagged frames in this case.

Configuring PVST+ support

PVST+ support is automatically enabled when the port receives a PVST BPDU. You can manually enable the support at any time or disable the support if desired.

If you want a tagged port to also support IEEE 802.1Q BPDUs, you need to enable the dual-mode feature on the port. The dual-mode feature is disabled by default and must be enabled manually.

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 commands such as the following.

```
device(config)# interface ethernet 1/1/1
device(config-if-1/1/1)# pvst-mode
```

NOTE
If you disable PVST+ support, the software still automatically enables PVST+ support if the port receives a BPDU with PVST+ format.

NOTE
If 802.1W and pvst-mode (either by auto-detection or by explicit configuration) are enabled on a tagged VLAN port, 802.1W will treat the PVST BPDUs as legacy 802.1D BPDUs.

Enabling dual-mode support

To enable the dual-mode feature on a port, enter the following command at the interface configuration level for the port.

```
device(config-if-1/1/1)# dual-mode
```

For more information about the dual-mode feature, refer to Dual-mode VLAN ports on page 304.

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

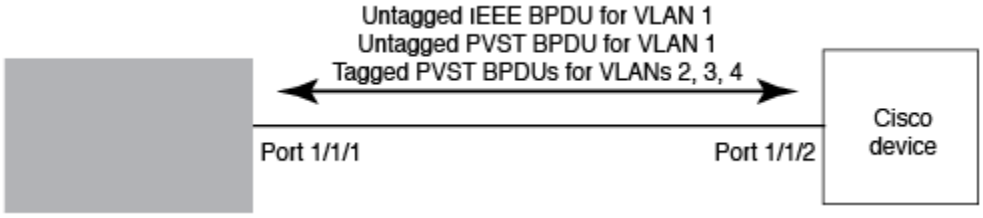
The following examples show configuration examples for two common configurations:

- Untagged IEEE 802.1Q BPDUs on VLAN 1 and tagged PVST+ BPDUs on other VLANs
- Tagged IEEE 802.1Q BPDUs on VLAN 1 and untagged BPDUs on another VLAN

Tagged port using default VLAN 1 as its port native VLAN

The following table 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 75 Default VLAN 1 for untagged BPDU



To implement this configuration, enter the following commands.

Commands on the 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)# dual-mode
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 the dual-mode feature and PVST+ support on the port. The 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 the 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 configuration leaves the default VLAN and the port Port Native VLAN unchanged. The default VLAN is 1 and the port Port Native VLAN also is 1. The dual-mode feature supports untagged frames on the default VLAN only. Thus, port 1/1/1 can send and receive untagged BPDUs for VLAN 1 and can send and receive tagged BPDUs for the other VLANs.

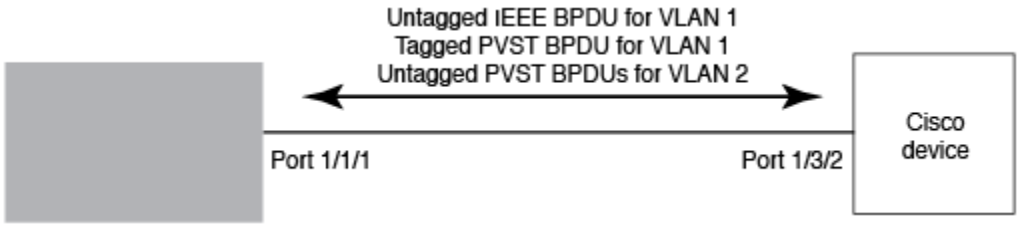
Port 1/1/1 will process BPDUs as follows:

- 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 Port Native VLAN is not VLAN 1. In this case, VLAN 1 uses tagged frames and VLAN 2 uses untagged frames.

FIGURE 76 Port Native VLAN 2 for Untagged BPDUs



To implement this configuration, enter the following commands.

Commands on the 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)#tagged ethernet 1/1/1
device(config-vlan-2)#exit
device(config)#interface ethernet 1/1/1
device(config-if-1/1/1)#dual-mode 2
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 VLANs 1 and 2, and enable the dual-mode feature and 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 allows the port to process tagged frames for VLAN 1. VLAN 2 is specified with the **dual-mode** command, which makes VLAN 2 the port Port Native VLAN. As a result, the port processes untagged frames and untagged PVST BPDUs on VLAN 2.

> **NOTE**
> Although VLAN 2 becomes the port untagged VLAN, the CLI still requires that you add the port to the VLAN as a tagged port, since the port is a member of more than one VLAN.

Port 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.

Note that when VLAN 1 is not the default VLAN, the ports must have the dual-mode feature enabled in order to process IEEE 802.1Q BPDUs.

For example, 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.

The following configuration is correct.

```
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)# dual-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)# dual-mode
device(config-if-1/1/2)# exit
```

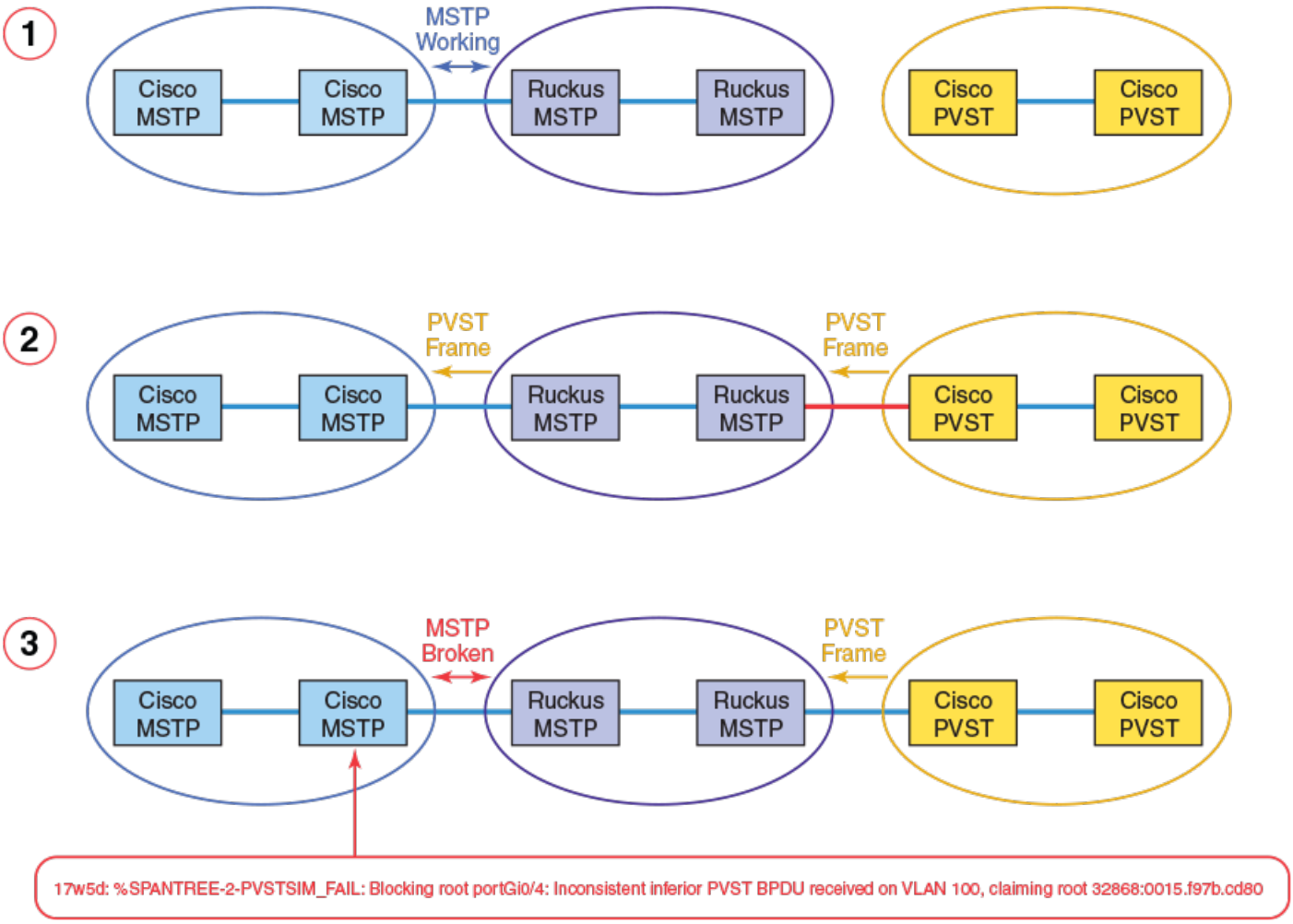
Setting the ports as dual-mode ensures that the untagged IEEE 802.1Q BPDUs reach the VLAN 1 instance.

PVST+ Protect

If a PVST+ packet is received on a port configured for MSTP, a Ruckus device floods it to all its ports in the VLAN so that it reaches other PVST+ devices across the VLAN. This flooding can sometime 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 77 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. 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 nondefault 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
 dual-mode
 pvstplus-protect
```

To display the status of PVST+ Protect on the Ethernet interface, including the number of dropped PVST+ BPDUs:

```
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
  Member of 7 L2 VLANs, port is dual mode 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 802.1w, it recognizes and processes these BPDUs and continues to operate in 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 BPDU protection by port

You can enable STP BPDU guard on individual interfaces. The feature is disabled by default.

To enable STP BPDU guard on a specific port, enter a command such as the following.

NOTE
Spanning tree must be enabled on the corresponding VLAN.

```
device(config) interface ethernet 1/1/2
device(config-if-e1000-1/1/2)# stp-bpdu-guard
```

You can also use the multiple interface command to enable this feature on multiple ports at once.

```
device(config)# interface ethernet 1/1/1 to 1/1/9
device(config-mif-1/1/1-1/1/9)# stp-bpdu-guard
device(config-mif-1/1/1-1/1/9)#
```

This will enable stp-bpdu-guard on ports 1/1/1 to 1/1/9

Re-enabling ports disabled by BPDU guard

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 232). In addition, the **show interface** command output will indicate that the port is errdisabled.

```
device# show interface ethernet 1/1/2
Gigabit Ethernet1/1/2 is ERR-DISABLED (bpduguard), line protocol is down
```

To re-enable a port that is in **errdisable** state, you must first disable the port then re-enable it. Enter commands such as the following.

```
device(config)# interface ethernet 1/1/2
device(config-if-e1000-1/1/2)# disable
device(config-if-e1000-1/1/2)# enable
```

If you attempt to enable an errdisabled port without first disabling it, the following error message will appear on the console.

```
device(config-if-e1000-1/1/2)# enable
Port 1/1/2 is errdisabled, do disable first and then enable to enable it
```

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:
Interface   Violation
Port 1/1/1   No
Port 1/1/2   No
Port 1/1/3   No
Port 1/1/4   No
Port 1/1/5   No
Port 1/1/6   No
Port 1/1/7   No
Port 1/1/8   No
Port 1/1/9   No
Port 1/1/10  No
Port 1/1/11  No
Port 1/1/12  Yes
Port 1/1/13  No
```

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:
Port
1
device(config-if-e1000-1/1/1)#
device(config-if-e1000-1/1/1)# show interfaces ethernet 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 249".

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:
Port 1/1/2
```

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
```

Packet InError Detection

Packet InError Detection identifies links that receive more number of bad frames than 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 236 and Setting the recovery interval on page 236.

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
```

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 fibre 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.

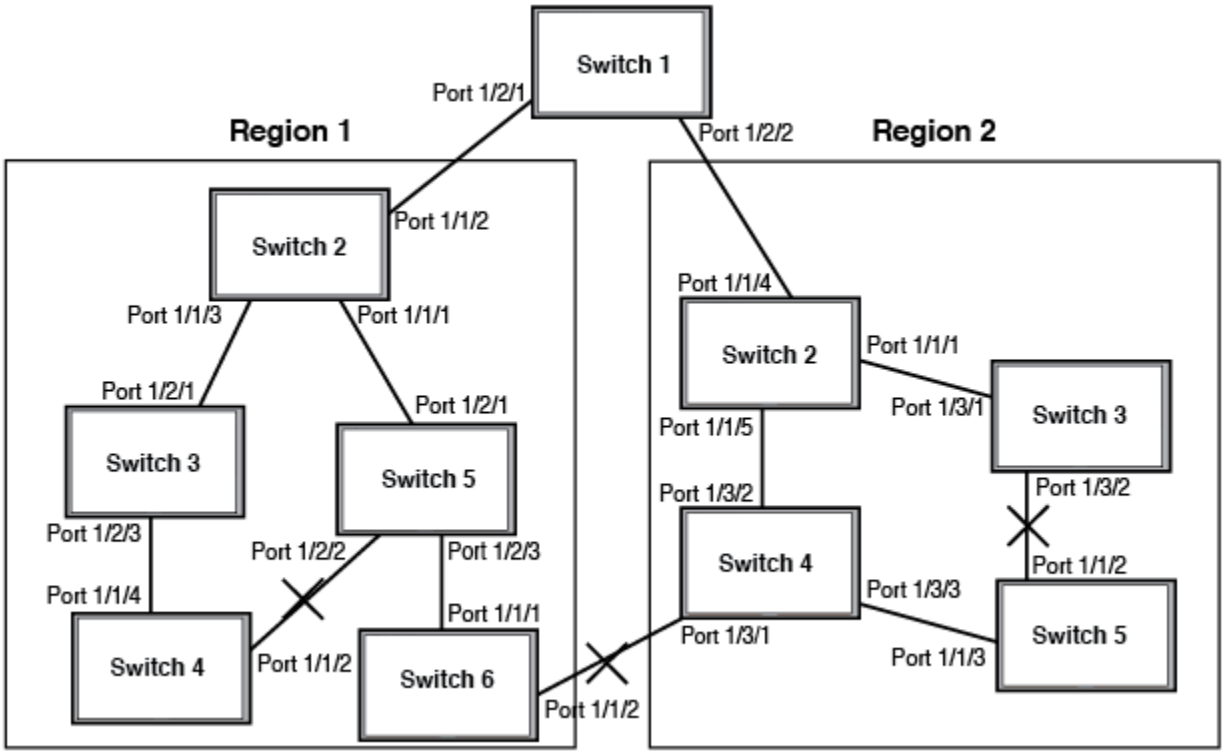
Multiple spanning-tree regions

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 78 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 78 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 tunnelled 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 GVRP configuration is 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
MSTP is not operational however until the **mstp start** command is issued as described in the “Forcing ports to transmit an MSTP BPDU” section.

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 whose 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 241.

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 additional MSTP parameters

To configure a switch for MSTP, you could configure the name and the revision on each switch that is being configured for MSTP. You must then create an MSTP Instance and assign an ID. VLANs are then assigned to MSTP instances. These instances must be configured on all switches that interoperate with the same VLAN assignments. Port cost, priority and global parameters can then be configured for individual ports and instances. In addition, operational edge ports and point-to-point links can be created and MSTP can be disabled on individual ports.

Each of the commands used to configure and operate MSTP are described in the following:

- Setting the MSTP name on page 242
- Setting the MSTP revision number on page 242
- Configuring an MSTP instance on page 242

- Configuring bridge priority for an MSTP instance on page 242
- Setting the MSTP global parameters on page 243
- Setting ports to be operational edge ports on page 243
- Setting automatic operational edge ports on page 243
- Setting point-to-point link on page 243
- Disabling MSTP on a port on page 245
- Forcing ports to transmit an MSTP BPDU on page 245
- Forcing ports to transmit an MSTP BPDU on page 245

Setting the MSTP name

Each switch that is running MSTP is configured with a name. It applies to the switch which can have many different VLANs that can belong to many different MSTP regions.

To configure an MSTP name, use a command such as the following at the Global Configuration level.

```
device(config)# mstp name Ruckus
```

Setting the MSTP revision number

Each switch that is running MSTP is configured with a revision number. It applies to the switch which can have many different VLANs that can belong to many different MSTP regions.

To configure an MSTP revision number, use a command such as the following at the Global Configuration level.

```
device(config)# mstp revision 4
```

Configuring an MSTP instance

An MSTP instance is configured with an MSTP ID for each region. Each region can contain 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. If pre-defined, a VLAN will be placed in the MSTI that it was assigned to immediately when the VLAN is created. Otherwise, the default operation is to condition of assign all new VLANs to the CIST. VLANs assigned to the CIST by default can be moved later to a specified MSTI.

To configure an MSTP instance and map one or more VLANs to that MSTI, use a command such as the following at the Global Configuration level.

```
device(config)# mstp instance 7 vlan 4 to 7
```

NOTE
The system does not allow an MSTI without any VLANs mapped to it. Consequently, removing all VLANs from an MSTI, deletes the MSTI from the system. The CIST by contrast will exist regardless of whether or not any VLANs are assigned to it or not. Consequently, if all VLANs are moved out of a CIST, the CIST will still exist and functional.

Configuring bridge priority for an MSTP instance

Priority can be configured for a specified instance. To configure priority for an MSTP instance, use a command such as the following at the Global Configuration level.

```
device(config)# mstp instance 1 priority 8192
```

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.

Setting the MSTP global parameters

MSTP has many of the options available in RSTP as well as some unique options. To configure MSTP Global parameters for all instances on a switch.

```
device(config)# mstp force-version 0 forward-delay 10 hello-time 4 max-age 12 max-hops 9
```

Setting ports to be operational edge ports

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. If a port is connected to an end device such as a PC, the port can be configured as an edge port. To configure ports as operational edge ports enter a command such as the following.

```
device(config)# mstp admin-edge-port ethernet 3/1/1
```

Setting automatic operational edge ports

You can 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. If the port receives a BPDU later, it is automatically reset to become an operational non-edge port. This feature is set globally to apply to all ports on a router where it is configured. This feature is configured as shown in the following.

```
device(config)# mstp edge-port-auto-detect
```

NOTE
If this feature is enabled, it takes the port about 3 seconds longer to come to the enable state.

Setting point-to-point link

You can set a point-to-point link between ports to increase the speed of convergence. To create a point-to-point link between ports, use a command such as the following at the Global Configuration level.

```
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+.

Protocol	Compatible with MSTP+
Spanning tree single (802.1D)	No
Rapid spanning tree single (802.1w)	No
MCT	No
Per-VLAN spanning tree (STP, 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
```

   Creates an MSTP instance on VLANs 4 to 7.

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.

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 239 before MSTP can be enabled.

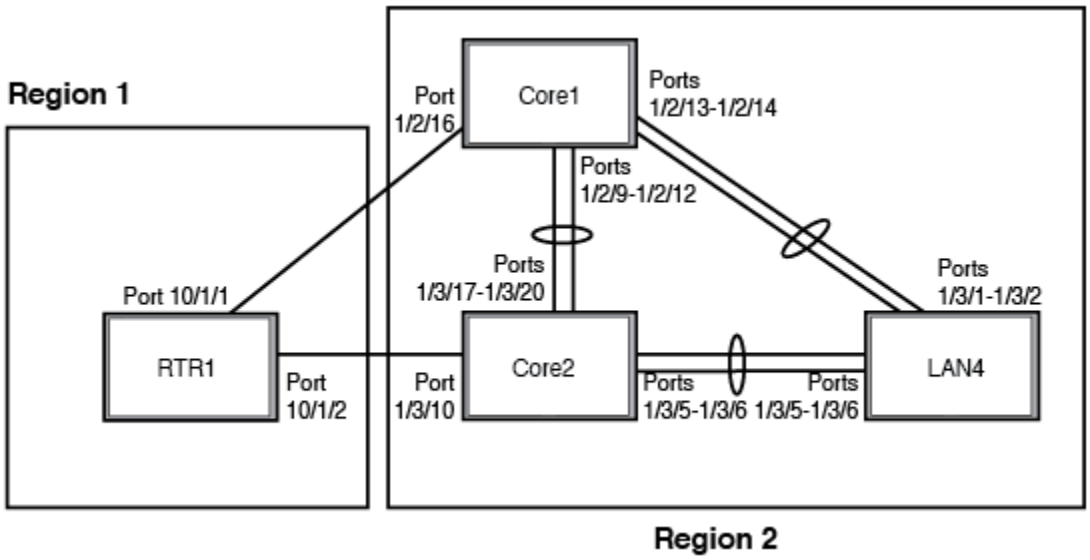
To enable MSTP on your switch, use the following at the Global Configuration level.

```
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 79 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 Instance (CIST). When you display information for this instance there are some differences with displaying 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 following command.

```
device# show mstp conf
MSTP CONFIGURATION
------------------
Name     : Reg1
Revision : 1
Version  : 3 (MSTP mode)
Status   : Started
Instance VLANs
-------- ----------------------------------------------------
0        4093
```

To display details about the MSTP that is configured on the device, enter the following command.

```
device# show mstp detail
MSTP Instance 0 (CIST) - VLANs: 4093
--------------------------------------------------------------------------------
Bridge: 800000b000c00000 [Priority 32768, SysId 0, Mac 00b000c00000]
FwdDelay 15, HelloTime 2, MaxHops 20, TxHoldCount 6
Port 1/1/4 - Role: DESIGNATED - State: FORWARDING
```

```
PathCost 20000, Priority 128, OperEdge T, OperPt2PtMac F, Boundary T
Designated - Root 800000b000c00000, RegionalRoot 800000b000c00000,
Bridge 800000b000c00000, ExtCost 0, IntCost 0
ActiveTimers - helloWhen 1
MachineState - PRX-DISCARD, PTX-IDLE, PPM-SENDING_RSTP, PIM-CURRENT
PRT-ACTIVE_PORT, PST-FORWARDING, TCM-INACTIVE
BPDUs - Rcvd MST 0, RST 0, Config 0, TCN 0
Sent MST 6, 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, see Root guard on page 232.

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.

Enabling MSTP root guard on a port

To enable MSTP root guard on a port, use the **spanning-tree root-protect** command on the interface as in the following example.

```
device# configure terminal
dcevice(config)# interface ethernet 1/1/5
device(config-if-e10000-1/1/5)# spanning-tree root-protect
```

Setting MSTP root guard timeout value

To configure root protection timeout value for MSTP root guard at the global level, use the **mstp root-protect timeout** command as in the following example.

```
device# configure terminal
device(config)# mstp root-protect timeout 60
```

Disabling MSTP root guard on a port

To disable MSTP root guard on the port, use the **no spanning-tree root-protect** command on the interface as in the following example.

```
device# configure terminal
dcevice(config)# interface ethernet 1/1/5
device(config-if-e10000-1/1/5)# no spanning-tree root-protect
```

Verifying the MSTP root guard configuration

To verify the MSTP root guard configuration, use the **show mstp** command as in the following examples. A root-inconsistent port is indicated by **RT-INCO**.

```
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
200000000000000a 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
```

To verify whether MSTP instances are in consistent state or in Inconsistent state, use 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)
```

VLANs

- VLAN overview....253
- Routing between VLANs....272
- Enabling port-based VLANs....276
- VLAN-based static MAC entries configuration....277
- Routing between VLANs using virtual routing interfaces (Layer 3 Switches only)....277
- IP subnet address on multiple port-based VLAN configuration....282
- VLAN groups and virtual routing interface group....285
- Topology groups....290
- Super-aggregated VLAN configuration....293
- 802.1ad tagging configuration....300
- Dual-mode VLAN ports....304
- Private VLAN configuration....307
- Displaying VLAN information....316

VLAN overview

The following sections provide details about the VLAN types and features supported on the FastIron family of switches.

VLAN support on FastIron devices

You can configure the following type of VLAN on FastIron 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. When you configure a port-based VLAN, the device automatically removes the ports you add to the VLAN from the default VLAN.

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, 4090, and 4093 are reserved for Ruckus internal use only. VLAN 4094 is reserved for use by Single STP. If you want to use VLANs 4091 and 4092 as configurable VLANs, you can assign them to different VLAN IDs. For more information, refer to Assigning different VLAN IDs to reserved VLANs 4091 and 4092 on page 273

NOTE

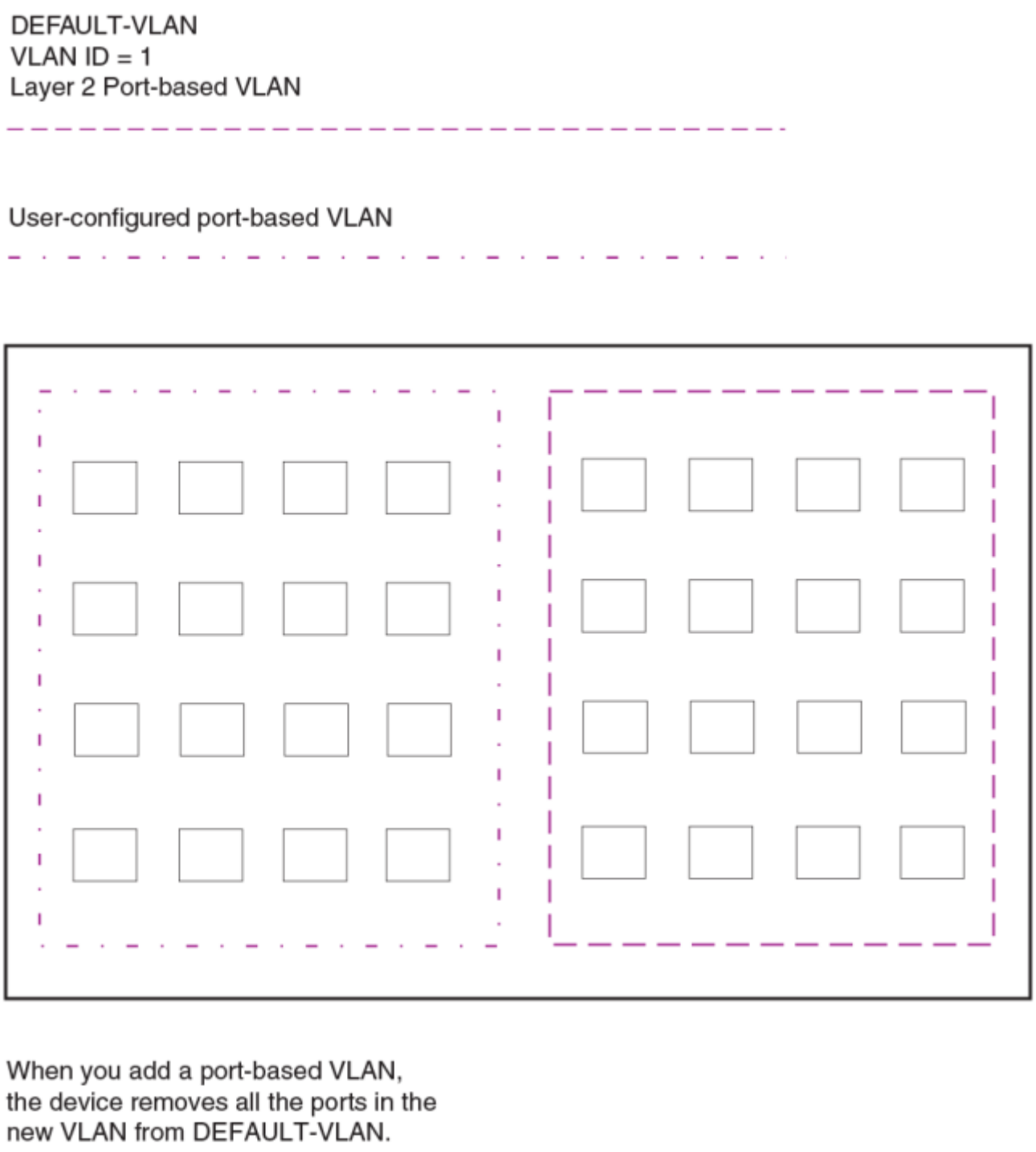
Each port-based VLAN can contain either tagged or untagged ports. A port cannot be a member of more than one port-based VLAN unless the port is tagged. 802.1Q tagging allows the port to add a four-byte tag field, which contains the VLAN ID, to each packet sent on the port. You also can configure port-based VLANs that span multiple devices by tagging the ports within the VLAN. The tag enables each device that receives the packet to determine the VLAN the packet belongs to. 802.1Q tagging applies only to Layer 2 VLANs, not to Layer 3 VLANs.

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.

The following figure shows an example of a Ruckus device on which a Layer 2 port-based VLAN has been configured.

FIGURE 80 Ruckus device containing user-defined Layer 2 port-based 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.

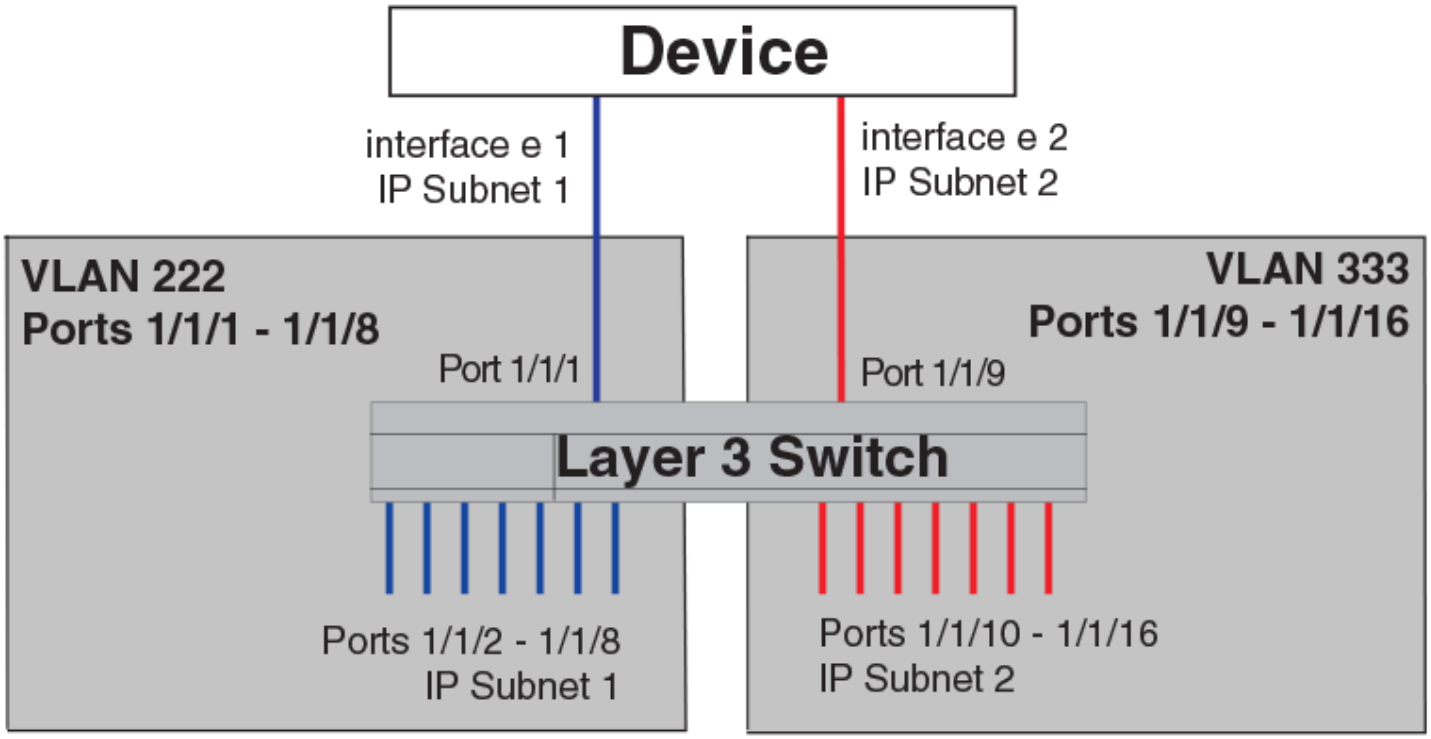
This section describes how to perform the following tasks for port-based VLANs using the CLI:

- Create a VLAN
- Delete a VLAN
- Modify a VLAN
- Change a VLAN priority
- Enable or disable STP on the VLAN

1--Simple port-based VLAN configuration

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 81 Port-based VLANs 222 and 333



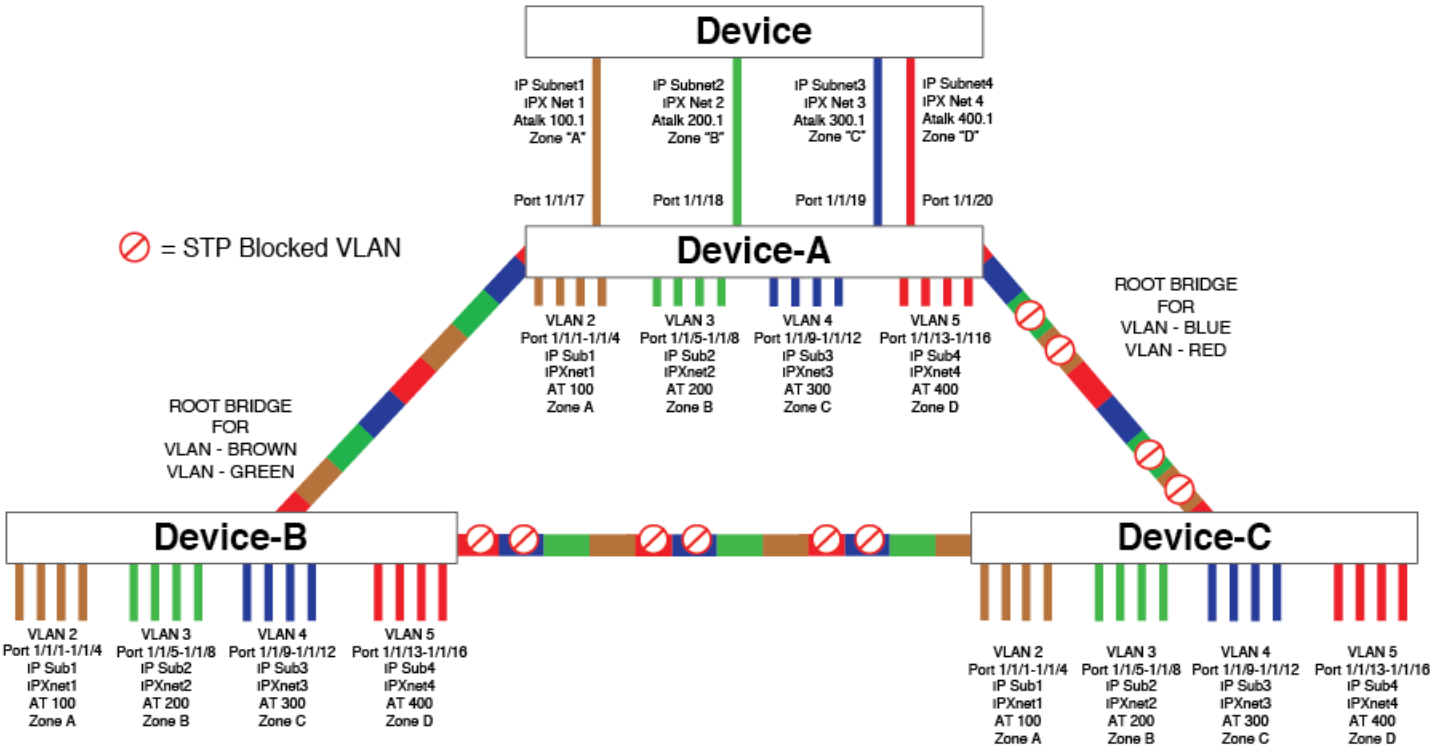
To create the two port-based VLANs shown in the above figure, enter the following commands.

```
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
```

2--More complex port-based VLAN configuration

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 is tagged. 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 82 More complex port-based VLAN



To configure the Port-based VLANs on the Layer 2 Switches in the above figure, use the following method.

Configuring port-based VLANs on Device-A

Enter the following commands to configure Device-A.

```
device> enable
device# configure terminal
device(config)# hostname Device-A
device-A(config)# vlan 2 name BROWN
device-A(config-vlan-2)# untagged ethernet 1/1/1 to 1/1/4 ethernet 1/1/17
device-A(config-vlan-2)# tagged ethernet 1/1/25 to 1/1/26
device-A(config-vlan-2)# spanning-tree
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
```

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
```

Syntax: vlan *vlan-id* **by port**

Syntax: untagged ethernet *unit*/*slotnum*/*portnum* [**to** *unit*/*slotnum*/*portnum* | **ethernet** *unit*/*slotnum*/*portnum*]

Syntax: tagged ethernet *unit*/*slotnum*/*portnum* [**to <** *unit*/*slotnum*/*portnum*> | **ethernet** *unit*/*slotnum*/*portnum*]

Syntax: [no] spanning-tree

Syntax: spanning-tree [**ethernet** *unit*/*slotnum*/*portnum* **path-cost** *value* **priority** *value*] **forward-delay** *value* **hello-time** *value* **maximum-age** *time* **priority** *value*

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 82 on page 257. 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.

Syntax: [no] vlan *vlan-id* **by port**

Removing a port from a VLAN

Suppose you want to remove port 11 from VLAN 4 on Device-A shown in Figure 82 on page 257. 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. Access the level of the CLI for configuring port-based VLAN 4 by entering the following command.

```
device-A(config)# vlan 4
```

3. Enter the following command to remove port 11 from VLAN 4.

```
device-A(config-vlan-4)# no untagged ethernet 11
deleted port ethernet 11 from port-vlan 4.
```

4. Enter the following commands to exit the VLAN configuration mode and save the configuration to the system-config file on flash memory.

```
device-A(config-vlan-4)# end
device-A# write memory
```

You can remove all the ports from a port-based VLAN without losing the rest of the VLAN configuration. However, you cannot configure an IP address on a virtual routing interface unless the VLAN contains ports. 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.

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 remove-vlan 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 remove-vlan feature is supported on the Ruckus ICX 7250, Ruckus ICX 7450, and Ruckus ICX 7750 devices only.

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 **remove-vlan all** command to remove all VLANs from the Ethernet port.
   - Enter the **remove-vlan vlan** 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)# remove-vlan all
(port(s) will be removed from 300 VLANs in single execution)

device(config-if-e40000-1/1/1)# remove-vlan vlan 2000 to 2005
(port(s) will be removed from 300 VLANs using this command)
```

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)# remove-vlan 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)# remove-vlan all
Port(s) ethernet 1/1/1 cannot be removed from VLANs 1 1001 to 1005
```

To remove a range of 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)# remove-vlan vlan 2001 to 2005
```

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)# remove-vlan all
```

Multi-range VLAN

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.

NOTE
The maximum number of VLANs you can create or configure with a single command is 64.

Creating a multi-range VLAN

To create more than one VLAN with a single command, you can specify the VLAN number and range.

Syntax: [no] vlan *num* **to** *num*

The *num* parameter specifies the VLAN ID.

To create a continuous range of VLANs, enter command such as the following.

```
device(config)#vlan 2 to 7
device(config-mvlan-2-7)#
```

Syntax: [no] vlan *num* **to** *num*

To create discontinuous VLANs, enter command such as the following.

```
device(config)#vlan 2 4 7
device(config-mvlan-2*7)#exit
```

Syntax: [no] vlan *num num num*

You can also create continuous and discontinuous VLANs. To create continuous and discontinuous VLANs, enter command such as the following.

```
device(config)#vlan 2 to 7 20 25
device(config-mvlan-2*25)#
```

Syntax: [no] vlan *num* **to** *num num*

Deleting a multi-range VLAN

You can also delete multiple VLANs with a single command.

To delete a continuous range of VLANs, enter command such as the following.

```
device(config)#no vlan 2 to 7
```

Syntax: [no] vlan *num* **to** *num*

To delete discontinuous VLANs, enter command such as the following.

```
device(config)#no vlan 2 4 7
```

Syntax: [no] vlan *num num num*

You can also delete continuous and discontinuous VLANs. To delete continuous and discontinuous VLANs, enter command such as the following.

```
device(config)#no vlan 2 to 7 20 25
```

Syntax: [no] vlan *num* **to** *num num*

If a single multi-range VLAN command contains more than 64 VLANs, the CLI does not add the VLAN IDs but instead displays an error message. An example is given below.

```
device(config)#vlan 100 to 356
ERROR -can't have more than 64 vlans at a time in a multi-range vlan command
```

Configuring a multi-range VLAN

You can configure multiple VLANs with a single command from the multi-range VLAN configuration level. For example, if you want to add tagged ethernet port 1/1/1 in the VLAN 16 17 20 21 22 23 24, enter the following commands.

```
device(config)#vlan 16 17 20 to 24
device(config-mvlan-16*24)#tag e 1/1/1
device(config-mvlan-16*24)#
```

The first command will take you to the multi-range VLAN configuration mode. The second command will add tagged ethernet port 1/1/1 in the specified VLANs, VLAN 16 17 20 21 22 23 and 24.

The following VLAN parameters can be configured with the specified VLAN range.

Command	Explanation
end	End Configuration level and goto Privileged level
exit	Exit current level
mac-vlan-permit	Define port to be used for MAC Based VLan
monitor	Monitor Ingress Traffic on this VLAN(Enable VLAN
multicast	IGMP snooping on this VLAN Mirroring)
no	Undo/disable commands
quit	Exit to User level
show	Show system information
spanning-tree	Set spanning tree for this VLAN
static-mac-address	Configure static MAC for this VLAN
tagged	802.1Q tagged port
vsrp	Configure VSRP

Command	Explanation
vsrp-aware	Configure VSRP Aware parameters
write	Write running configuration to flash or terminal

The VLAN parameters configured for the VLAN range are written in the configuration file of the individual VLANs. These VLAN parameters can also be removed or modified from the individual VLANs. In the following example, as the first step, create VLANs 16 17 20 21 22 23 24. Further, as the second step, add Ethernet port 1/1/1 in all the VLANs. As the third step, enabled 802.1w spanning tree on all these VLANs.

```
device(config)#vlan 16 17 20 to 24
device(config-mvlan-16*24)#tag ethernet 1/1/1
device(config-mvlan-16*24)#
Added tagged port(s) ethernet 1/1/1 to port-vlan16.
Added tagged port(s) ethernet 1/1/1 to port-vlan 17.
Added tagged port(s) ethernet 1/1/1 to port-vlan 20.
Added tagged port(s) ethernet 1/1/1 to port-vlan 21.
Added tagged port(s) ethernet 1/1/1 to port-vlan 22.
Added tagged port(s) ethernet 1/1/1 to port-vlan 23.
Added tagged port(s) ethernet 1/1/1 to port-vlan 24.
device(config-mvlan-16*24)#span 802-1w
```

The Ethernet port ethernet 1/1/1 and spanning tree 802.1w is added to the database of each VLAN separately. You can verify the configuration with the **show running-config** command. See the example below.

```
device(config-mvlan-16*24)#show run
Current configuration:
!
!
output omitted

!
!
vlan 1 name DEFAULT-VLAN by port
!
vlan 16 by port
tagged ethernet 1/1/1
spanning-tree 802-1w
!
vlan 17 by port
tagged ethernet 1/1/1
spanning-tree 802-1w
!
vlan 20 by port
tagged ethernet 1/1/1
spanning-tree 802-1w
!
vlan 21 by port
tagged ethernet 1/1/1
spanning-tree 802-1w
!
vlan 22 by port
tagged ethernet 1/1/1
spanning-tree 802-1w
!
vlan 23 by port
tagged ethernet 1/1/1
spanning-tree 802-1w
!
vlan 24 by port
tagged ethernet 1/1/1
spanning-tree 802-1w
!
!
output omitted
```

```
!
!
```

Now you can modify any one or some of the VLANs. See the example below.

In the following example, disable the spanning tree 802.1w on VLANs 22,23 and 24, And, verify with **show running-config** output that the spanning tree 802.1w is disabled on specified VLANs, VLAN 22, 23 and 24 and not on the VLANs 16, 17, 20 and 21.

```
device(config)#vlan 22 to 24
device(config-mvlan-22-24)#no span 8
device(config-mvlan-22-24)#exit
device(config)#show run
Current configuration:
output omitted
!
!
vlan 1 name DEFAULT-VLAN by port
!
vlan 16 by port
tagged ethernet 1/1/1
spanning-tree 802-1w
!
vlan 17 by port
tagged ethernet 1/1/1
spanning-tree 802-1w
!
vlan 20 by port
tagged ethernet 1/1/1
spanning-tree 802-1w
!
vlan 21 by port
tagged ethernet 1/1/1
spanning-tree 802-1w
vlan 22 by port
tagged ethernet 1/1/1
!
vlan 23 by port
tagged ethernet 1/1/1
!
vlan 24 by port
tagged ethernet 1/1/1
output omitted
```

Multi-range VLAN show commands

This section describes the show commands for multi-range VLAN parameters.

In the multi-range VLAN mode, some of the Show commands are also available. The output of the Show commands in multi-range VLAN mode displays the information related to the specific VLANs only. See the example below.

In the following example, the first command will change the interface configuration level to the multi-range VLAN mode for the VLANs 4, 5 and 6. In the multi-range VLAN mode, enter the command **show 802.1w**. The output will display the information of STP for 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
```

The following **show** parameters can be viewed for the specified VLAN range from the multi-range VLAN configuration mode. The output of these commands displays information about the specified VLANs only.

TABLE 20 VLAN show parameters

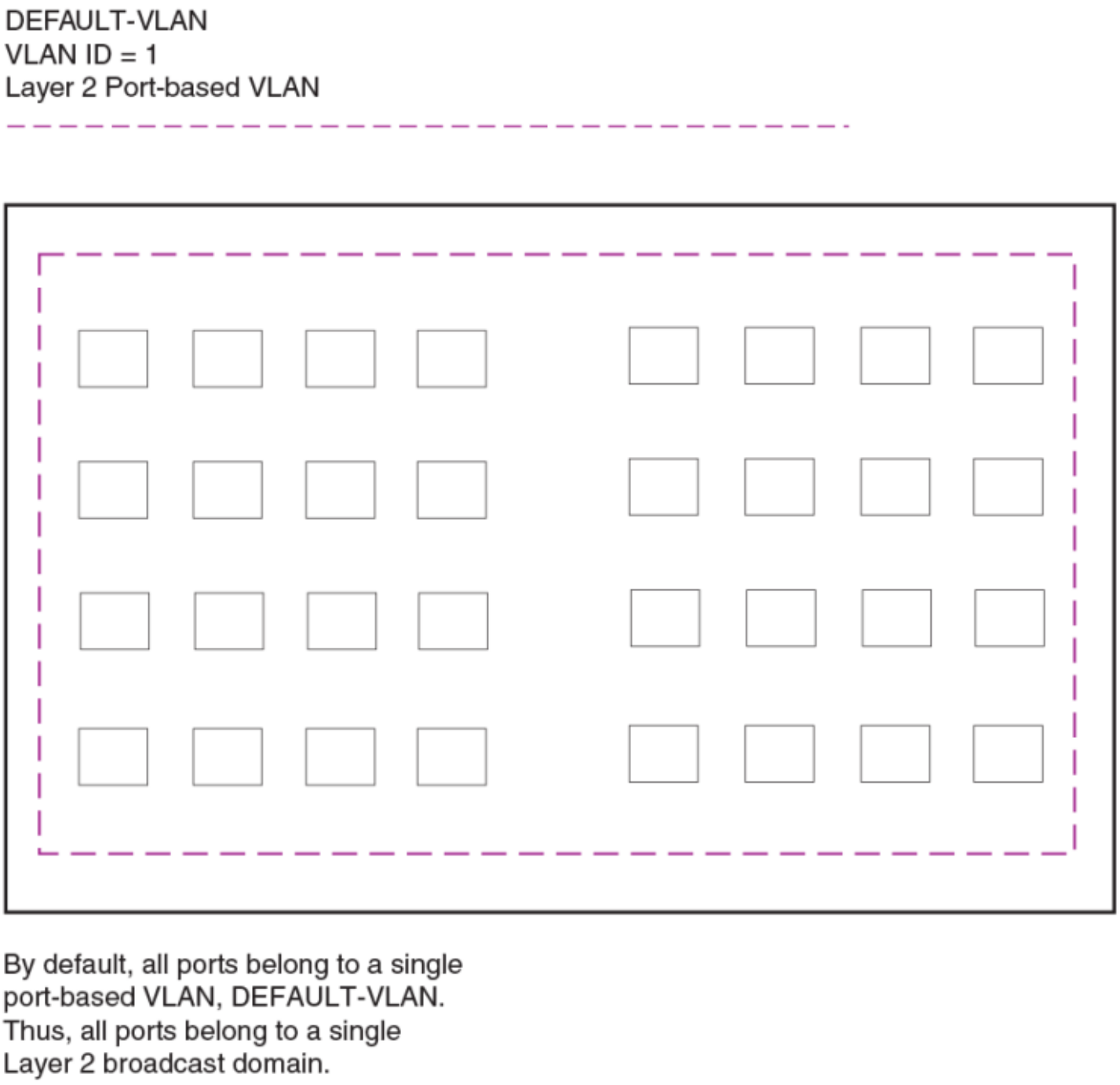
Command	Definition
802-1w	Rapid Spanning tree IEEE 802.1w status
mac-address	MAC address table
span	Spanning tree status
vlan	VLAN status
vsrp	Show VSRP commands

Default VLAN

By default, all the ports on a FastIron device are in a single port-based VLAN. This VLAN is called the DEFAULT-VLAN and is VLAN number 1. FastIron devices do not contain any protocol VLANs or IP subnet VLANs by default.

The following figure shows an example of the default Layer 2 port-based VLAN.

FIGURE 83 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. When you configure the VLAN, the Ruckus device automatically removes the ports that you place in the VLAN from DEFAULT-VLAN. By removing the ports from the default VLAN, the Ruckus device ensures that each port resides in only one Layer 2 broadcast domain.

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 the default VLAN on page 273.

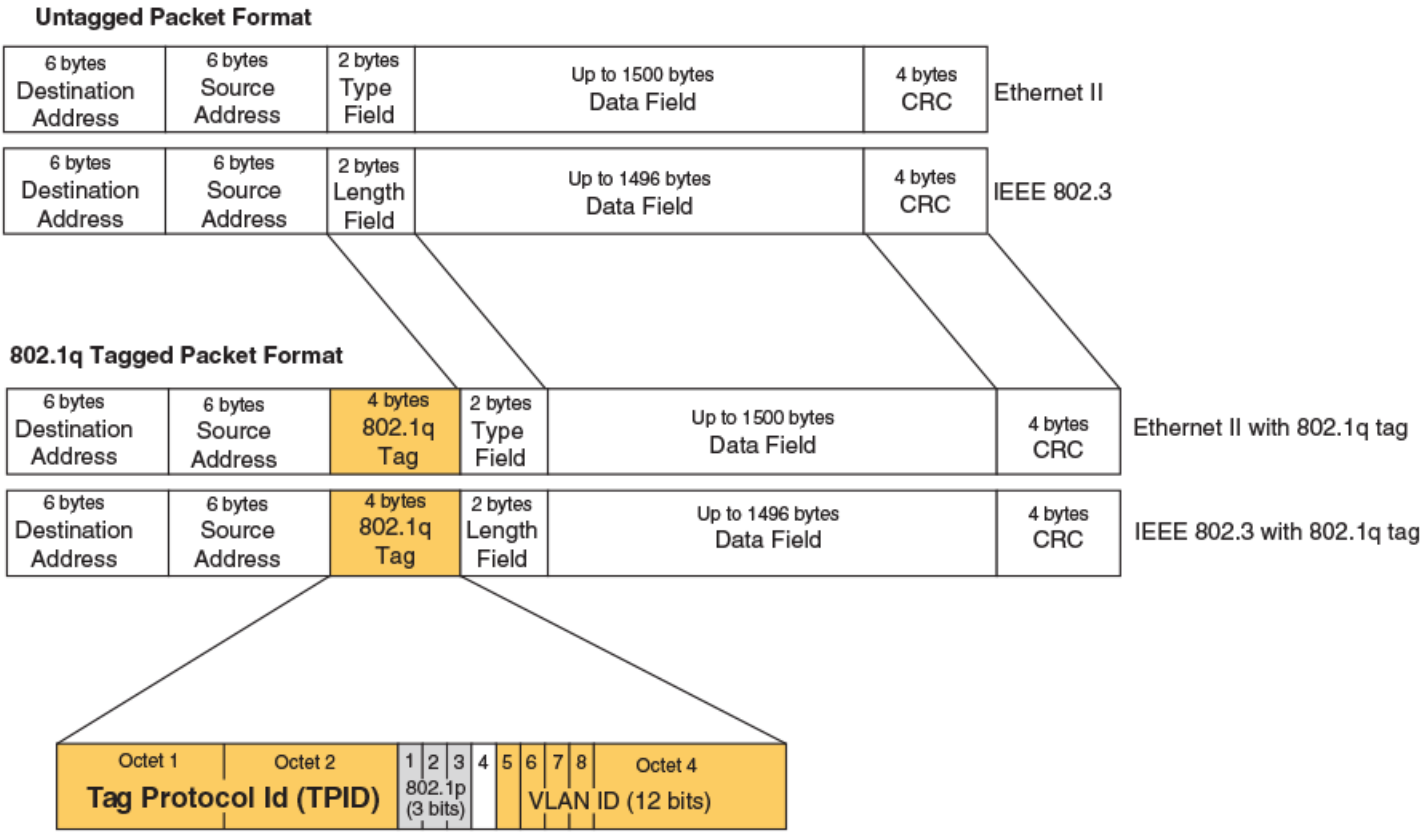
802.1Q tagging

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 84 Packet containing a Ruckus 802.1Q VLAN tag

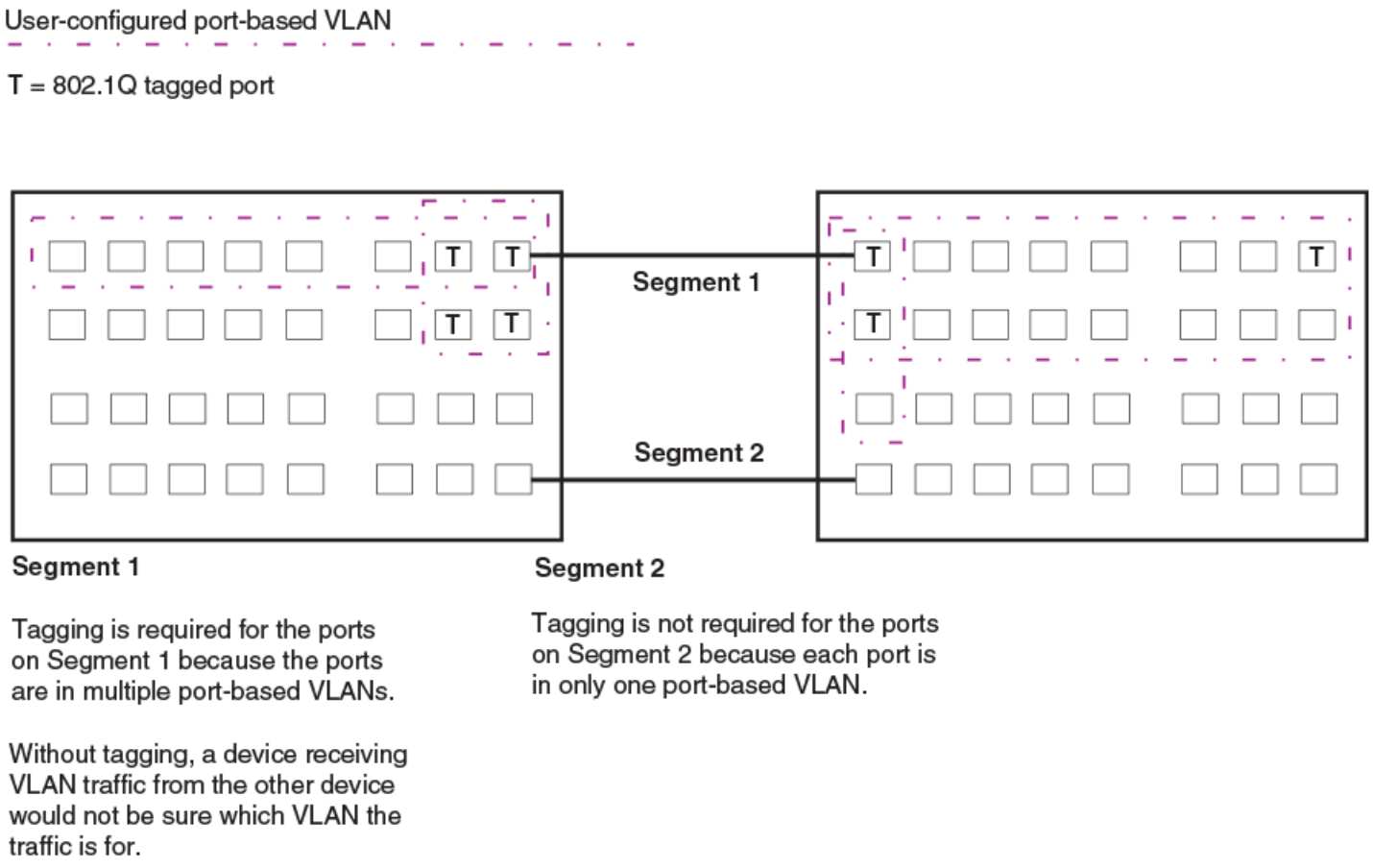


If you configure a VLAN that spans multiple devices, you need to use tagging only if a port connecting one of the devices to the other is a member of more than one port-based VLAN. If a port connecting one device to the other is a member of only a single port-based VLAN, tagging is not required.

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 85 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-types 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-types, but are not very flexible with the tag-types they accept.

NOTE
Layer 2 control protocol PDU tunneling over 802.1ad (Q-in-Q) is not supported on FastIron devices.

802.1 ad tagging for ICX 7xxx devices

The following enhancements allow the ICX 7xxx devices, including those in an IronStack, to use Q-in-Q and SAV, by allowing the changing of a tag profile for ports:

- In addition to the default tag type 0x8100, you can now configure one additional global tag profile with a number from 0xffff.
- 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 the configuration mode.

```
device(config)# tag-profile 9500
```

Syntax: [no] tag-profile *tag-no*

tag-no - the number of the tag, can be 0x8100 (default), or 0xffff

To direct individual ports or on a range of ports to this tag profile, enter commands similar to the following.

```
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 (STP)

The default state of STP depends on the device type:

- STP is disabled by default on Ruckus Layer 3 Switches.
- STP is enabled by default on Ruckus Layer 2 Switches.

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. Thus, 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. Thus, you cannot enable or disable STP for individual protocol VLANs or for IP subnet VLANs. The STP state of a port-based VLAN containing these other types of VLANs 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.

It is possible that STP will block one or more ports in a protocol VLAN that uses a virtual routing interface to route to other VLANs. For IP protocol and IP subnet VLANs, even though some of the physical ports of the virtual routing interface are blocked, the virtual routing interface can still route so long as at least one port in the virtual routing interface protocol VLAN is not blocked by STP.

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 Spanning Tree Protocol on page 159.

Virtual routing interfaces

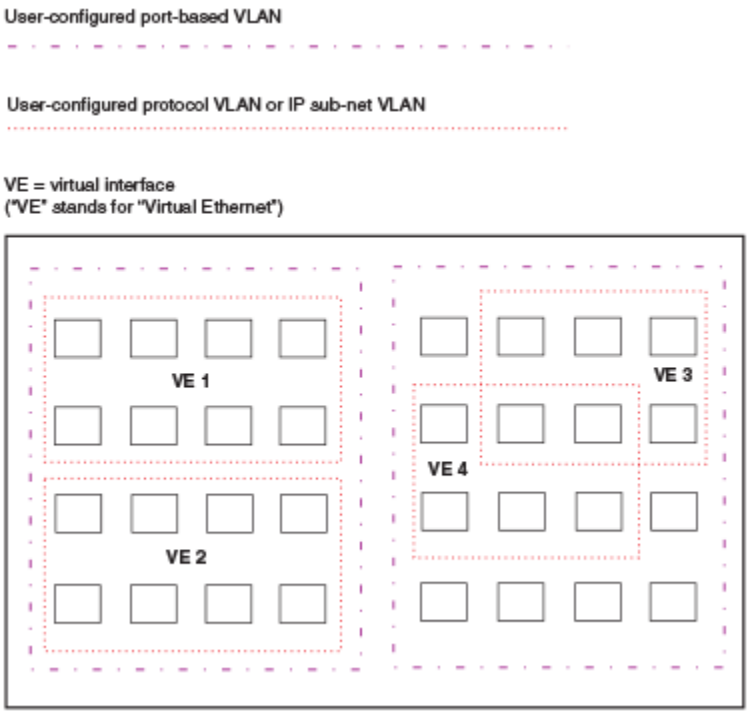
A virtual routing interface is a logical routing interface that Ruckus Layer 3 Switches use to route Layer 3 protocol traffic between protocol VLANs.

Ruckus devices send Layer 3 traffic at Layer 2 within a protocol VLAN. However, Layer 3 traffic from one protocol VLAN to another must be routed.

If you want the device to be able to send Layer 3 traffic from one protocol VLAN to another, you must configure a virtual routing interface on each protocol VLAN, then configure routing parameters on the virtual routing interfaces. For example, to enable a Layer 3 Switch to route IP traffic from one IP subnet VLAN to another, you must configure a virtual routing interface on each IP subnet VLAN, then configure the appropriate IP routing parameters on each of the virtual routing interfaces.

The following figure shows an example of Layer 3 protocol VLANs that use virtual routing interfaces for routing.

FIGURE 86 Use virtual routing interfaces for routing between Layer 3 protocol 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.

For configuration information, refer to VLAN groups and virtual routing interface group on page 285.

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 293.

Trunk group ports and VLAN membership

A trunk group is a set of physical ports that are configured to act as a single physical interface. Each trunk group port configuration is based on the configuration of the lead port, which is the lowest numbered port in the group.

If you add a trunk group lead port to a VLAN, all of the ports in the trunk group become members of that VLAN.

Summary of VLAN configuration rules

A hierarchy of VLANs exists between the Layer 2 and Layer 3 protocol-based VLANs:

- Port-based VLANs are at the lowest level of the hierarchy.
- Layer 3 protocol-based VLANs, IP, IPv6, Decnet, and NetBIOS are at the middle level of the hierarchy.
- IP subnet VLANs are at the top of the hierarchy.

  NOTE
  You cannot have a protocol-based VLAN and a subnet or network VLAN of the same protocol type in the same port-based VLAN.

As a Ruckus device receives packets, the VLAN classification starts from the highest level VLAN first. Therefore, if an interface is configured as a member of both a port-based VLAN and an IP protocol VLAN, IP packets coming into the interface are classified as members of the IP protocol VLAN because that VLAN is higher in the VLAN hierarchy.

Multiple VLAN membership rules

- A port can belong to multiple, unique, overlapping Layer 3 protocol-based VLANs without VLAN tagging.
- 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.
- When both port and protocol-based VLANs are configured on a given device, all protocol VLANs must be strictly contained within a port-based VLAN. A protocol VLAN cannot include ports from multiple port-based VLANs. This rule is required to ensure that port-based VLANs remain loop-free Layer 2 broadcast domains.
- IP protocol VLANs and IP subnet VLANs cannot operate concurrently on the system or within the same port-based VLAN.
- One of each type of protocol VLAN is configurable within each port-based VLAN on the Layer 2 Switch.
- Multiple IP subnet VLANs are configurable within each port-based VLAN on the Layer 2 Switch.
- Removing a configured port-based VLAN from a Ruckus Networks Layer 2 Switch or Layer 3 Switch automatically removes any protocol-based VLAN, IP subnet VLAN, or any Virtual Ethernet router interfaces defined within the Port-based VLAN.

Routing between VLANs

Ruckus Layer 3 Switches can locally route IP between VLANs defined within a single router. All other routable protocols or protocol VLANs (for example, DecNet) must be routed by another external router capable of routing the protocol.

Virtual routing interfaces (Layer 2 Switches only)

You need to configure virtual routing interfaces if an IP VLAN, IP subnet VLAN needs to route protocols to another port-based VLAN on the same router. A virtual routing interface can be associated with the ports in only a single port-based VLAN. Virtual router interfaces must be defined at the highest level of the VLAN hierarchy.

If you do not need to further partition the port-based VLAN by defining separate Layer 3 VLANs, you can define a single virtual routing interface at the port-based VLAN level and enable IP routing on a single virtual routing interface.

Some configurations may require simultaneous switching and routing of the same single protocol across different sets of ports on the same router. When IP routing is enabled on a Ruckus Layer 3 Switch, you can route these protocols on specific interfaces while bridging them on other interfaces. In this scenario, you can create two separate backbones for the same protocol, one bridged and one routed.

To bridge IP protocols, you need to configure an IP protocol, IP subnet VLAN and not assign a virtual routing interface to the VLAN. Packets for these protocols are bridged or switched at Layer 2 across ports on the router that are included in the Layer 3 VLAN. If these VLANs are built within port-based VLANs, they can be tagged across a single set of backbone fibers to create separate Layer 2 switched and Layer 3 routed backbones for the same protocol on a single physical backbone.

Routing between VLANs using virtual routing interfaces (Layer 3 Switches only)

Ruckus calls the ability to route between VLANs with virtual routing interfaces Integrated Switch Routing (ISR) . There are some important concepts to understand before designing an ISR backbone.

Virtual router interfaces can be defined on port-based, IP protocol, IP subnet VLANs.

To create any type of VLAN on a Ruckus Layer 3 Switch, Layer 2 forwarding must be enabled. When Layer 2 forwarding is enabled, the Layer 3 Switch becomes a Switch on all ports for all non-routable protocols.

If the router interfaces for IP is configured on physical ports, then routing occurs independent of the Spanning Tree Protocol (STP). However, if the router interfaces are defined for any type VLAN, they are virtual routing interfaces and are subject to the rules of STP.

If your backbone consists of virtual routing interfaces all within the same STP domain, it is a bridged backbone, not a routed one. This means that the set of backbone interfaces that are blocked by STP will be blocked for routed protocols as well. The routed protocols will be able to cross these paths only when the STP state of the link is FORWARDING. This problem is easily avoided by proper network design.

When designing an ISR network, pay attention to your use of virtual routing interfaces and the spanning-tree domain. If Layer 2 switching of your routed protocols (IP) is not required across the backbone, then the use of virtual routing interfaces can be limited to edge switch ports within each router. Full backbone routing can be achieved by configuring routing on each physical interface that connects to the backbone. Routing is independent of STP when configured on a physical interface.

If your ISR design requires that you switch IP at Layer 2 while simultaneously routing the same protocols over a single backbone, then create multiple port-based VLANs and use VLAN tagging on the backbone links to separate your Layer 2 switched and Layer 3 routed networks.

There is a separate STP domain for each port-based VLAN. Routing occurs independently across port-based VLANs or STP domains. You can define each end of each backbone link as a separate tagged port-based VLAN. Routing will occur independently across the port-based VLANs. Because each port-based VLAN STP domain is a single point-to-point backbone connection, you are guaranteed to never have an STP loop. STP will never block the virtual router interfaces within the tagged port-based VLAN, and you will have a fully routed backbone.

Dynamic port assignment (Layer 2 Switches and Layer 3 Switches)

All Switch ports are dynamically assigned to any Layer 3 VLAN on Ruckus Layer 2 Switches and any non-routable VLAN on Ruckus Layer 3 Switches. To maintain explicit control of the VLAN, you can explicitly exclude ports when configuring any Layer 3 VLAN on a Ruckus Layer 2 Switch or any non-routable Layer 3 VLAN on a Ruckus Layer 3 Switch.

If you do not want the ports to have dynamic membership, you can add them statically. This eliminates the need to explicitly exclude the ports that you do not want to participate in a particular Layer 3 VLAN.

Assigning a different VLAN ID to the default VLAN

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.

To reassign the default VLAN to a different VLAN ID, enter the following command.

```
device(config)# default-vlan-id 4095
```

Syntax: [no] default-vlan-id *vlan-id*

You must specify a valid VLAN ID that is not already in use. For example, if you have already defined VLAN 10, do not try to use "10" as the new VLAN ID for the default VLAN. Valid VLAN IDs are numbers from 1 - 4095.

> **NOTE**
> This command does not change the properties of the default VLAN. Changing the name allows you to use the VLAN ID "1" as a configurable VLAN.

Assigning different VLAN IDs to reserved VLANs 4091 and 4092

If you want to use VLANs 4091 and 4092 as configurable VLANs, you can assign them to different VLAN IDs.

For example, to reassign reserved VLAN 4091 to VLAN 10, enter the following commands.

```
device(config)# reserved-vlan-map vlan 4091 new-vlan 10
Reload required. Please write memory and then reload or power cycle.
device(config)# write mem
device(config)# exit
device# reload
```

> **NOTE**
> You must save the configuration (write mem) and reload the software to place the change into effect.

The above configuration changes the VLAN ID of 4091 to 10. After saving the configuration and reloading the software, you can configure VLAN 4091 as you would any other VLAN.

Syntax: [no] reserved-vlan-map vlan 4091 | **4092 new-vlan** *vlan-id*

For *vlan-id* , enter a valid VLAN ID that is not already in use. For example, if you have already defined VLAN 20, do not try to use "20 as the new VLAN ID. Valid VLAN IDs are numbers from 1 - 4090, 4093, and 4095. VLAN ID 4094 is reserved for use by the Single Spanning Tree feature.

Viewing reassigned VLAN IDs for reserved VLANs 4091 and 4092

To view the assigned VLAN IDs for reserved VLANs 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
  CPU VLAN                    4091             10              10
  All Ports VLAN              4092             33              33
```

Syntax: show reserved-vlan-map

The following table defines the fields in the output of the **show reserved-vlan-map** command.

TABLE 21 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.[1]
Current	The current VLAN ID for the reserved VLAN.[1]

1. 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 trunk group ports

When a "lead" trunk group port is assigned to a VLAN, all other members of the trunk group are automatically added to that VLAN. A lead port is the first port of a trunk group port range; for example, "1" in 1 - 4 or "5" in 5 - 8.

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> enable
No password has been assigned yet...
device-A# configure terminal
device-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 3
device-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-tree
device-B(config-vlan-3)#
```

4. Enter the following commands to exit the VLAN CONFIG mode and save the configuration to the system-config file on flash memory.

```
device-B(config-vlan-3)#
device-B(config-vlan-3)# end
device-B# write memory
device-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.

Syntax: vlan *vlan-id* **by port**

Syntax: [no] spanning-tree

Syntax: spanning-tree [**ethernet** *unit**/**slotnum**/** portnum* **path-cost** *value* **priority** *value*] **forward-delay** *value* **hello-time** *value* **maximum-age** *time* **priority** *value*

Bridge STP parameters (applied to all ports within a VLAN):

- Forward Delay - the period of time a bridge will wait (the listen and learn period) before forwarding data packets. Possible values: 4 - 30 seconds. Default is 15.
- Maximum Age - the interval a bridge will wait for receipt of a hello packet before initiating a topology change. Possible values: 6 - 40 seconds. Default is 20.
- Hello Time - the interval of time between each configuration BPDU sent by the root bridge. Possible values: 1 - 10 seconds. Default is 2.
- Priority - a parameter used to identify the root bridge in a network. The bridge with the lowest value has the highest priority and is the root. Possible values: 1 - 65,535. Default is 32,678.

Port parameters (applied to a specified port within a VLAN):

- Path Cost - a parameter used to assign a higher or lower path cost to a port. Possible values: 1 - 65535. Default is (1000/Port Speed) for Half-Duplex ports and is (1000/Port Speed)/2 for Full-Duplex ports.
- Priority - value determines when a port will be rerouted in relation to other ports. Possible values: 0 - 255. Default is 128.

Enabling port-based VLANs

When using the CLI, port and protocol-based VLANs are created by entering one of the following commands at the global CONFIG level of the CLI.

To create a port-based VLAN, enter commands such as the following.

```
device(config)#vlan 222 by port
device(config)#vlan 222 name Mktg
```

The num parameter specifies the VLAN ID. The valid range for VLAN IDs starts at 1 on all systems but the upper limit of the range differs depending on the device. In addition, you can change the upper limit on some devices using the **system max-vlans...** command.

The string parameter is the VLAN name and can be a string up to 32 characters. You can use blank spaces in the name if you enclose the name in double quotes (for example, "Product Marketing".)

Depending on device support, you can configure up to 4000 port-based VLANs. 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. 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, 4090, and 4093 are reserved for Ruckus internal use only. VLAN 4094 is reserved for use by Single STP. Also, if you are running an earlier release, VLAN IDs 4091 and 4092 may be reserved for Ruckus internal use only. If you want to use VLANs 4091 and 4092 as configurable VLANs, you can assign them to different VLAN IDs. For more information, refer to Assigning different VLAN IDs to reserved VLANs 4091 and 4092 on page 273

NOTE
The second command is optional and also creates the VLAN if the VLAN does not already exist. You can enter the first command after you enter the second command if you first exit to the global CONFIG level of the CLI.

Assigning IEEE 802.1Q tagging to a port

When a port is tagged, it allows communication among the different VLANs to which it is assigned. A common use for this might be to place an email server that multiple groups may need access to on a tagged port, which in turn, is resident in all VLANs that need access to the server.

NOTE
Tagging does not apply to the default VLAN.

When using the CLI, ports are defined as either tagged or untagged at the VLAN level.

Command syntax for assigning 802.1Q tagging to a port

Suppose you want to make port 5 a member of port-based VLAN 4, a tagged port. To do so, enter the following.

```
device(config)#vlan 4
device(config-vlan-4)#tagged e 5
```

Syntax: tagged ethernet [*stack**/**slot**/**port*] *portnum* [**to** [*stack**/**port*] *portnum* [**ethernet** [*slotnum/*] *portnum...*]]

The *slotnum* parameter is required on chassis devices.

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
```

Syntax: [no] static-mac-address *mac-addr* **drop**

Use the **no** form of the command to remove the static MAC address drop configuration.

Routing between VLANs using virtual routing interfaces (Layer 3 Switches only)

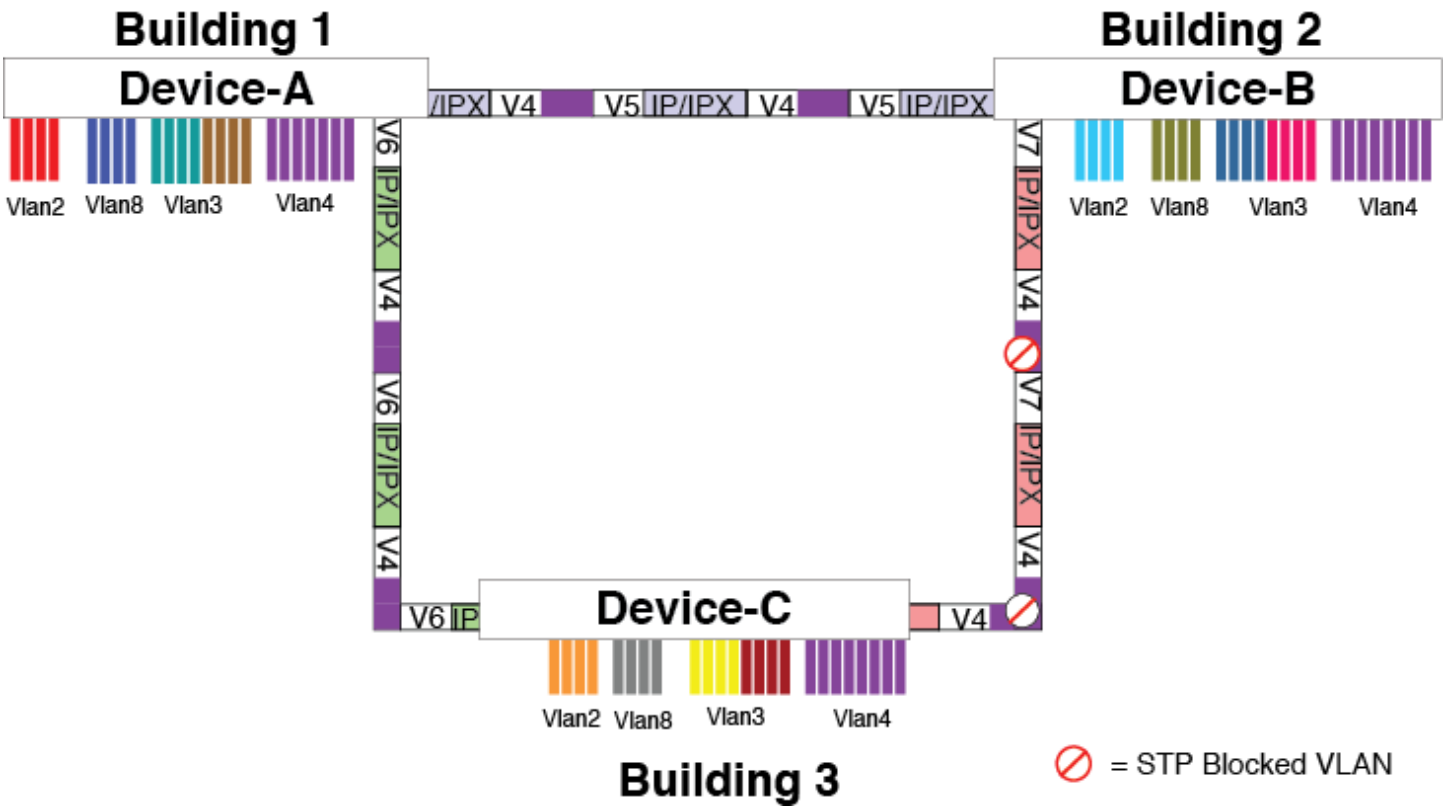
Ruckus Layer 3 Switches offer the ability to create a virtual routing interface within a Layer 2 STP port-based VLAN. This combination of multiple Layer 2 or Layer 3 broadcast domains, or both, and virtual routing interfaces are the basis for Ruckus Networks' very powerful Integrated Switch Routing (ISR) technology. ISR is very flexible and can solve many networking problems. The following example is meant to provide ideas by demonstrating some of the concepts of ISR.

Suppose you want to move routing out to each of three buildings in a network. Remember that the only protocols present on VLAN 2 and VLAN 3 are IP. Therefore, you can eliminate tagged ports 1/1/25 and 1/1/26 from both VLAN 2 and VLAN 3 and create new tagged port-based VLANs to support separate IP subnets for each backbone link.

You also need to create unique IP subnets within VLAN 2 and VLAN 3 at each building. This will create a fully routed IP backbone for VLAN 2 and VLAN 3. However, VLAN 4 has no protocol restrictions across the backbone. In fact there are requirements for NetBIOS and DecNet to be bridged among the three building locations. The IP subnet that exists within VLAN 4 must remain a flat Layer 2 switched STP domain. You enable routing for IP on a virtual routing interface only on Device-A. This will provide the flat IP segment with connectivity to the rest of the network. Within VLAN 4 IP will follow the STP topology. All other IP subnets will be fully routed and have use of all paths at all times during normal operation.

The following figure shows the configuration described above.

FIGURE 87 Routing between protocol-based VLANs



To configure the Layer 3 VLANs and virtual routing interfaces on the Layer 3 Switch in the above figure, use the following procedure.

Configuring Layer 3 VLANs and virtual routing interfaces on the Device-A

Enter the following commands to configure Device-A. The following commands enable OSPF or RIP routing.

```
device>enable
No password has been assigned yet...
device# configure terminal
device(config)# hostname Device-A
device-A(config)# router ospf
device-A(config-ospf-router)# area 0.0.0.0 normal
Please save configuration to flash and reboot.
device-A(config-ospf-router)#
```

The following commands create the port-based VLAN 2. In the previous example, an external device defined the router interfaces for VLAN 2. With ISR, routing for VLAN 2 is done locally within each device. Therefore, there are two ways you can solve this problem. One way is to create a unique IP subnet VLAN, each with its own virtual routing interface and unique IP address within VLAN 2 on each device. In this example, this is the configuration used for VLAN 3. The second way is to split VLAN 2 into two separate port-based VLANs and create a virtual router interface within each port-based VLAN. Later in this example, this second option is used to create a port-based VLAN 8 to show that there are multiple ways to accomplish the same task with ISR.

You also need to create the Other-Protocol VLAN within port-based VLAN 2 and 8 to prevent unwanted protocols from being Layer 2 switched within port-based VLAN 2 or 8. Note that the only port-based VLAN that requires STP in this example is VLAN 4. You will need to configure the rest of the network to prevent the need to run STP.

```
device-A(config-ospf-router)# vlan 2 name IP-Subnet_10.1.2.0/24
device-A(config-vlan-2)# untagged ethernet 1/1/1 to 1/1/4
device-A(config-vlan-2)# no spanning-tree
device-A(config-vlan-2)# router-interface ve1
device-A(config-vlan-2)# other-proto name block_other_protocols
device-A(config-vlan-other-proto)# no dynamic
device-A(config-vlan-other-proto)# exclude ethernet 1/1/1 to 1/1/4
```

Once you have defined the port-based VLAN and created the virtual routing interface, you need to configure the virtual routing interface just as you would configure a physical interface.

```
device-A(config-vlan-other-proto)# interface ve1
device-A(config-vif-1)# ip address 10.1.2.1/24
device-A(config-vif-1)# ip ospf area 0.0.0.0
```

Do the same thing for VLAN 8.

```
device-A(config-vif-1)# vlan 8 name Network2
device-A(config-vlan-8)# untagged ethernet 1/1/5 to 1/1/8
device-A(config-vlan-8)# no spanning-tree
device-A(config-vlan-8)# router-interface ve 2
device-A(config-vlan-8)# other-proto name block-other-protocols
device-A(config-vlan-other-proto)# no dynamic
device-A(config-vlan-other-proto)# exclude ethernet 1/1/5 to 1/1/8
device-A(config-vlan-other-proto)# interface ve2
device-A(config-vif-1)# ip address 10.1.2.2/24
device-A(config-vif-1)# ip ospf area 0.0.0.0
```

The next thing you need to do is create VLAN 3. This is very similar to the previous example with the addition of virtual routing interfaces to the IP subnet VLANs. Also there is no need to exclude ports from the IP subnet VLANs on the router.

```
device-A(config-vif-2)# vlan 3 name IP_Sub_&_Net_VLAN
device-A(config-vlan-3)# untagged ethernet 1/1/9 to 1/1/16
device-A(config-vlan-3)# no spanning-tree
device-A(config-vlan-3)# ip-subnet 10.1.1.0/24
device-A(config-vlan-ip-subnet)# static ethernet 1/1/9 to 1/1/12
device-A(config-vlan-ip-subnet)# router-interface ve3
device-A(config-vlan-ip-subnet)# other-proto name block-other-protocols
device-A(config-vlan-other-proto)# exclude ethernet 1/1/9 to 1/1/16
device-A(config-vlan-other-proto)# no dynamic
device-A(config-vlan-other-proto)# interface ve 3
device-A(config-vif-3)# ip addr 10.1.1.1/24
device-A(config-vif-3)# ip ospf area 0.0.0.0
```

Now configure VLAN 4. Remember this is a flat segment that, in the previous example, obtained its IP default gateway services from an external device. In this example, Device-A will provide the routing services for VLAN 4. You also want to configure the STP priority for VLAN 4 to make Device-A the root bridge for this VLAN.

```
device-A(config-vif-4)# vlan 4 name Bridged_ALL_Protocols
device-A(config-vlan-4)# untagged ethernet 1/1/17 to 1/1/24
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)# router-interface ve5
device-A(config-vlan-4)# interface ve5
device-A(config-vif-5)# ip address 10.1.3.1/24
device-A(config-vif-5)# ip ospf area 0.0.0.0
```

It is time to configure a separate port-based VLAN for each of the routed backbone ports (Ethernet 1/1/25 and 1/1/26). If you do not create a separate tagged port-based VLAN for each point-to-point backbone link, you need to include tagged interfaces for Ethernet 1/1/25 and 1/1/26 within VLANs 2, 3, and 8. This type of configuration makes the entire backbone a single STP domain

for each VLAN 2, 3, and 8. In this scenario, the virtual routing interfaces within port-based VLANs 2, 3, and 8 will be accessible using only one path through the network. The path that is blocked by STP is not available to the routing protocols until it is in the STP FORWARDING state.

```
device-A(config-vif-5)# vlan 5 name Rtr_BB_to_Bldg.2
device-A(config-vlan-5)# tagged ethernet 1/1/25
device-A(config-vlan-5)# no spanning-tree
device-A(config-vlan-5)# router-interface ve6
device-A(config-vlan-5)# vlan 6 name Rtr_BB_to_Bldg.3
device-A(config-vlan-6)# tagged ethernet 1/1/26
device-A(config-vlan-6)# no spanning-tree
device-A(config-vlan-6)# router-interface ve7
device-A(config-vlan-6)# interface ve6
device-A(config-vif-6)# ip addr 10.1.4.1/24
device-A(config-vif-6)# ip ospf area 0.0.0.0
device-A(config-vif-6)# interface ve7
device-A(config-vif-7)# ip addr 10.1.5.1/24
device-A(config-vif-7)# ip ospf area 0.0.0.0
device-A(config-vif-7)#
```

This completes the configuration for Device-A. The configuration for Device-B and Device-C is very similar except for a few issues which are as follows:

- IP subnets configured on Device-B and Device-C must be unique across the entire network, except for the backbone port-based VLANs 5, 6, and 7 where the subnet is the same but the IP address must change.
- There is no need to change the default priority of STP within VLAN 4.
- There is no need to include a virtual router interface within VLAN 4.
- The backbone VLAN between Device-B and Device-C must be the same at both ends and requires a new VLAN ID. The VLAN ID for this port-based VLAN is VLAN 7.

Configuring Layer 3 VLANs and virtual routing interfaces for Device-B

Enter the following commands to configure Device-B.

```
device> enable
No password has been assigned yet...
device# config terminal
device(config)# hostname Device-B
device-B(config)# router ospf
device-B(config-ospf-router)# area 0.0.0.0 normal
device-B(config-ospf-router)# vlan 2 name IP-Subnet_10.1.6.0/24
device-B(config-vlan-2)# untagged ethernet 1/1/1 to 1/1/4
device-B(config-vlan-2)# no spanning-tree
device-B(config-vlan-2)# router-interface ve1
device-B(config-vlan-2)# other-proto name block-other-protocols
device-B(config-vlan-other-proto)# no dynamic
device-B(config-vlan-other-proto)# exclude ethernet 1/1/1 to 1/1/4
device-B(config-vlan-other-proto)# interface ve1
device-B(config-vif-1)# ip addr 10.1.6.1/24
device-B(config-vif-1)# ip ospf area 0.0.0.0
device-B(config-vif-1)# other-proto name block-other-protocols
device-B(config-vlan-other-proto)# no dynamic
device-B(config-vlan-other-proto)# exclude ethernet 1/1/5 to 1/1/8
device-B(config-vlan-other-proto)# interface ve2
device-B(config-vif-2)# vlan 3 name IP_Sub_&_Net_VLAN
device-B(config-vlan-3)# untagged ethernet 1/1/9 to 1/1/16
device-B(config-vlan-3)# no spanning-tree
device-B(config-vlan-3)# ip-subnet 10.1.7.0/24
device-B(config-vlan-ip-subnet)# static ethernet 1/1/9 to 1/1/12
device-B(config-vlan-ip-subnet)# router-interface ve3
device-B(config-vlan-ip-subnet)# other-proto name block-other-protocols
device-B(config-vlan-other-proto)# exclude ethernet 1/1/9 to 1/1/16
```

```
device-B(config-vlan-other-proto)# no dynamic
device-B(config-vlan-other-proto)# interface ve 3
device-B(config-vif-3)# ip addr 10.1.7.1/24
device-B(config-vif-3)# ip ospf area 0.0.0.0
device-B(config-vif-3)# interface ve4
device-B(config-vif-4)# vlan 4 name Bridged_ALL_Protocols
device-B(config-vlan-4)# untagged ethernet 1/1/17 to 1/1/24
device-B(config-vlan-4)# tagged ethernet 1/1/25 to 1/1/26
device-B(config-vlan-4)# spanning-tree
device-B(config-vlan-4)# vlan 5 name Rtr_BB_to_Bldg.1
device-B(config-vlan-5)# tagged ethernet 1/1/25
device-B(config-vlan-5)# no spanning-tree
device-B(config-vlan-5)# router-interface ve5
device-B(config-vlan-5)# vlan 7 name Rtr_BB_to_Bldg.3
device-B(config-vlan-7)# tagged ethernet 1/1/26
device-B(config-vlan-7)# no spanning-tree
device-B(config-vlan-7)# router-interface ve6
device-B(config-vlan-7)# interface ve5
device-B(config-vif-5)# ip addr 10.1.4.2/24
device-B(config-vif-5)# ip ospf area 0.0.0.0
device-B(config-vif-5)# interface ve6
device-B(config-vif-6)# ip addr 10.1.8.1/24
device-B(config-vif-6)# ip ospf area 0.0.0.0
device-B(config-vif-6)#
```

Configuring Layer 3 VLANs and virtual routing interfaces for Device-C

Enter the following commands to configure Device-C.

```
device> enable
No password has been assigned yet...
device# config terminal
device(config)# hostname Device-C
device-C(config)# router ospf
device-C(config-ospf-router)# area 0.0.0.0 normal
device-C(config-ospf-router)# vlan 2 name IP-Subnet_10.1.9.0/24
device-C(config-vlan-2)# untagged ethernet 1/1/1 to 1/1/4
device-C(config-vlan-2)# no spanning-tree
device-C(config-vlan-2)# router-interface ve1
device-C(config-vlan-2)# other-proto name block-other-protocols
device-C(config-vlan-other-proto)# no dynamic
device-C(config-vlan-other-proto)# exclude ethernet 1/1/1 to 1/1/4
device-C(config-vlan-other-proto)# interface ve1
device-C(config-vif-1)# ip addr 10.1.9.1/24
device-C(config-vif-1)# ip ospf area 0.0.0.0
device-C(config-vif-1)# vlan 8 name Network9
device-C(config-vlan-8)# untagged ethernet 1/1/5 to 1/1/8
device-C(config-vlan-8)# no span
device-C(config-vlan-8)# router-interface ve2
device-C(config-vlan-8)# other-proto name block-other-protocols
device-C(config-vlan-other-proto)# no dynamic
device-C(config-vlan-other-proto)# exclude ethernet 1/1/5 to 1/1/8
device-C(config-vlan-other-proto)# interface ve2
device-C(config-vif-1)# ip addr 10.1.9.2/24
device-C(config-vif-1)# ip ospf area 0.0.0.0
device-C(config-vif-2)# vlan 3 name IP_Sub_&_Net_VLAN
device-C(config-vlan-3)# untagged ethernet 1/1/9 to 1/1/16
device-C(config-vlan-3)# no spanning-tree
device-C(config-vlan-3)# ip-subnet 10.1.10.0/24
device-C(config-vlan-ip-subnet)# static ethernet 1/1/9 to 1/1/12
device-C(config-vlan-ip-subnet)# router-interface ve3
device-C(config-vlan-ip-subnet)# other-proto name block-other-protocols
device-C(config-vlan-other-proto)# exclude ethernet 1/1/9 to 1/1/16
device-C(config-vlan-other-proto)# no dynamic
device-C(config-vlan-other-proto)# interface ve 3
device-C(config-vif-3)# ip addr 10.1.10.1/24
device-C(config-vif-3)# ip ospf area 0.0.0.0
```

```
device-C(config-vif-3)# interface ve4
device-C(config-vif-4)# vlan 4 name Bridged_ALL_Protocols
device-C(config-vlan-4)# untagged ethernet 1/1/17 to 1/1/24
device-C(config-vlan-4)# tagged ethernet 1/1/25 to 1/1/26
device-C(config-vlan-4)# spanning-tree
device-C(config-vlan-4)# vlan 7 name Rtr_BB_to_Bldg.2
device-C(config-vlan-7)# tagged ethernet 1/1/25
device-C(config-vlan-7)# no spanning-tree
device-C(config-vlan-7)# router-interface ve5
device-C(config-vlan-7)# vlan 6 name Rtr_BB_to_Bldg.1
device-C(config-vlan-6)# tagged ethernet 1/1/26
device-C(config-vlan-6)# no spanning-tree
device-C(config-vlan-6)# router-interface ve6
device-C(config-vlan-6)# interface ve5
device-C(config-vif-5)# ip addr 10.1.8.2/24
device-C(config-vif-5)# ip ospf area 0.0.0.0
device-C(config-vif-5)# interface ve6
device-C(config-vif-6)# ip addr 10.1.5.2/24
device-C(config-vif-6)# ip ospf area 0.0.0.0
device-C(config-vif-6)#
```

IP subnet address on multiple port-based VLAN configuration

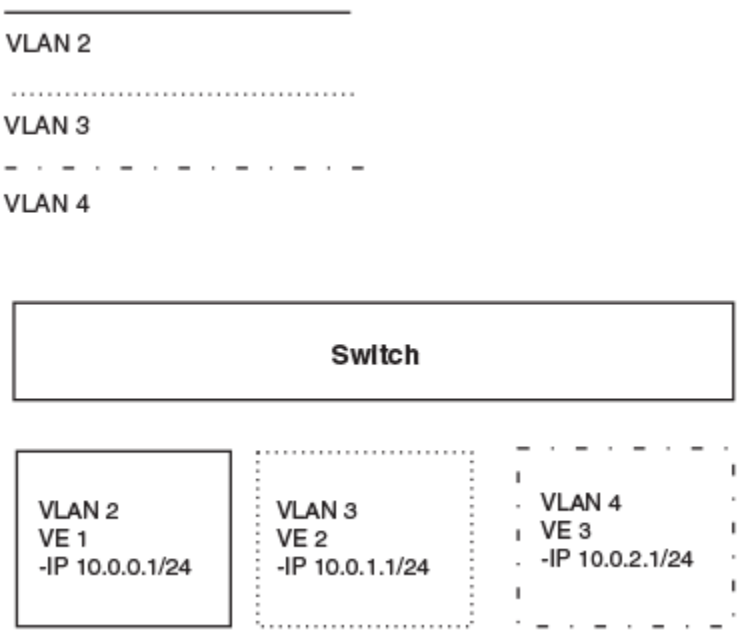
For a Ruckus device to route between port-based VLANs, you must add a virtual routing interface to each VLAN. Generally, you also configure a unique IP subnet address on each virtual routing interface. For example, if you have three port-based VLANs, you add a virtual routing interface to each VLAN, 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 285. You might be able to achieve the results you want using the methods in that section instead.

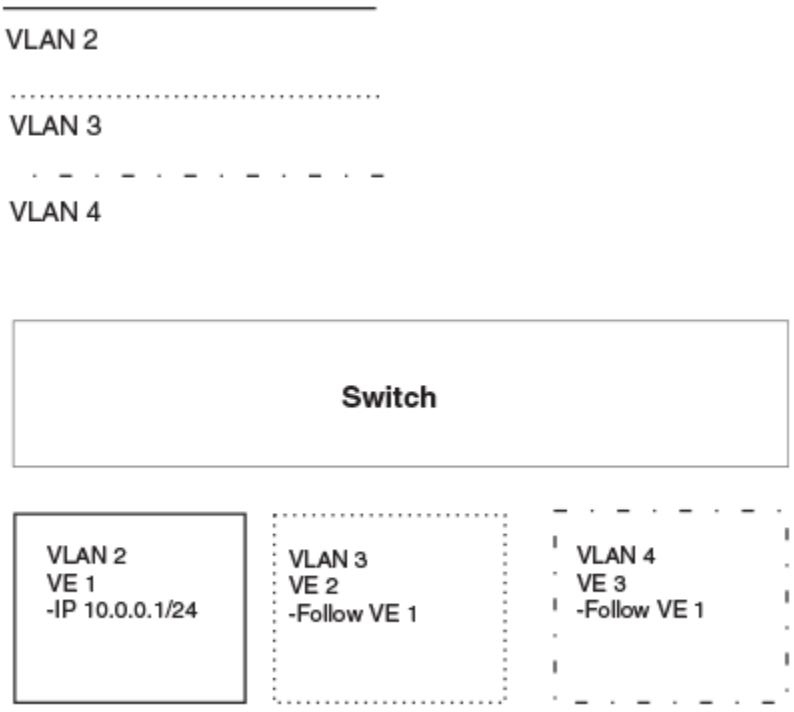
The following figure shows an example of this type of configuration.

FIGURE 88 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 89 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 89, 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)# router-interface ve 1
```

Syntax: router-interface ve *number*

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)# router-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)# router-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.

NOTE

Configuring a VLAN group

To configure a VLAN group, enter commands such as the following.

```
device(config)# vlan-group 1 vlan 2 to 257
device(config-vlan-group-1)# tagged 1/1/1 to 1/1/2
```

The first command in this example begins configuration for VLAN group 1, and assigns VLANs 2 through 257 to the group. The second command adds ports 1/1/1 and 1/1/2 as tagged ports. Because all the VLANs in the group share the ports, you must add the ports as tagged ports.

Syntax: vlan-group *num* **vlan** *vlan-id* **to** *vlan-id*

Syntax: tagged ethernet *unit*/*slotnum*/*portnum* [**to** *unit*/*slotnum*/*portnum* | **ethernet** *unit*/*slotnum*/*portnum*]

The **vlan-group** *num* parameter specifies the VLAN group ID and can be from 1 - 32. The **vlan** *vlan-id* **to** *vlan-id* parameters specify a contiguous range (a range with no gaps) of individual VLAN IDs. Specify the low VLAN ID first and the high VLAN ID second. The command adds all of the specified VLANs to the VLAN group.

You can add up to 256 VLANs with the command at one time. To add more than 256 VLANs, enter separate commands. For example, to configure VLAN group 1 and add 512 VLANs to the group, enter the following commands.

```
device(config)# vlan-group 1 vlan 2 to 257
device(config-vlan-group-1)# add-vlan 258 to 513
```

NOTE

The device memory must be configured to contain at least the number of VLANs you specify for the higher end of the range. For example, if you specify 2048 as the VLAN ID at the high end of the range, you first must increase the memory allocation for VLANs to 2048 or higher. Additionally, 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, before you configure the VLAN groups. This is true regardless of whether you use the virtual routing interface groups. The memory allocation is required because the VLAN groups and virtual routing interface groups have a one-to-one mapping. Refer to Allocating memory for more VLANs, more associated ports, or more virtual routing interfaces on page 288.

If a VLAN within the range you specify is already configured, or if the range contains more than 256 VLANs, the CLI does not add the group but instead displays an error message.

```
device(config)#vlan-group 1 vlan 2 to 1000
VLAN group 1 is too big. Only 256 vlans are allowed at a time
```

In this case, create the group by specifying a valid contiguous range. Then add more VLANs to the group after the CLI changes to the configuration level for the group. See the following example.

```
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
device(config-vlan-group-2)#add-vlan 1751 to 2000
```

You can add or remove individual VLANs or VLAN ranges from the VLAN group at configuration level. For example, if you want to add VLANs 1001 and 1002 to VLAN group 1 and remove VLANs 900 through 1000, enter the following commands.

```
device(config-vlan-group-1)# add-vlan 1001 to 1002
device(config-vlan-group-1)# remove-vlan 900 to 1000
```

Syntax: add-vlan *vlan-id* [**to** *vlan-id*]

Syntax: remove-vlan *vlan-id* [**to** *vlan-id*]

The *vlan-id* **to** *vlan-id* parameters specify a contiguous range (a range with no gaps) of individual VLAN IDs. Specify the low VLAN ID first and the high VLAN ID second. You can add or remove up to 256 VLANs at a time. To add or remove more than 256 VLANs, do so using separate commands. For example, to remove 512 VLANs from VLAN group 1, enter the following commands.

```
device(config-vlan-group-1)# remove-vlan 400 to 654
device(config-vlan-group-1)# remove-vlan 655 to 910
```

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
!
```

Syntax: show vlan-group [*group-id*]

The *group-id* specifies a VLAN group. If you do not use this parameter, the configuration information for all the configured VLAN groups is displayed.

Configuring a virtual routing interface group

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.

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.

CLI syntax for virtual routing interface group

To configure a virtual routing interface group, enter commands such as the following.

```
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
device(config-vif-group-1)# ip address 10.10.10.1/24
```

These commands enable VLAN group 1 to have a group virtual routing interface, then configure virtual routing interface group 1. The software always associates a virtual routing interface group only with the VLAN group that has the same ID. In this example, the VLAN group ID is 1, so the corresponding virtual routing interface group also must have ID 1.

Syntax: group-router-interface

Syntax: interface group-ve *num*

Syntax: [no] ip address *ip-addr ip-mask* [**secondary**]

or

Syntax: [no] ip address *ip-addr/mask-bits* [**secondary**]

The **router-interface-group** command enables a VLAN group to use a virtual routing interface group. Enter this command at the configuration level for the VLAN group. This command configures the VLAN group to use the virtual routing interface group that has the same ID as the VLAN group. You can enter this command when you configure the VLAN group for the first time or later, after you have added tagged ports to the VLAN and so on.

The *num* parameter in the **interface group-ve***num* command specifies the ID of the VLAN group with which you want to associate this virtual routing interface group. The VLAN group must already be configured and enabled to use a virtual routing interface group. The software automatically associates the virtual routing interface group with the VLAN group that has the same ID. You can associate a virtual routing interface group only with the VLAN group that has the same ID.

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 syntax and usage for the **ip address** command is the same as when you use the command at the interface level to add an IP interface.

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 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 and, for Chassis devices, the amount of DRAM on the management module. 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.

TABLE 22 VLAN, VPORT, and virtual routing interface support

VLANs		VPORTs		Virtual routing interfaces	
Default maximum	Configurable maximum	Default maximum	Configurable maximum	Default maximum	Configurable maximum
64	4094	8,192	524,032	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 285.

Increasing the number of VLANs you can configure

NOTE
Although you can specify up to 4095 VLANs, you can configure only 4094 VLANs. VLAN ID 4094 is reserved for use by the Single Spanning Tree feature.

To increase the maximum number of VLANs you can configure, enter commands such as the following at the global CONFIG level of the CLI.

```
device(config)# system-max vlan 2048
device(config)# write memory
device(config)# end
device# reload
```

Syntax: system-max vlan *num*

The num parameter indicates the maximum number of VLANs. The range of valid values depends on the device you are configuring. Refer to Table 22 on page 289.

Increasing the number of ports that can be assigned to VLANs

VPORT entries are created in the software database for each port association to a VLAN. The default VPORT limit is 64, which equals the default Max VLAN setting. By default, this limits port-to-VLAN associations to 8,192 (Max VLANs * MAX ports per VLAN, or 64 * 128).

To maximize the number of VLANs and hence the number of port-to-VLAN associations, enter the following commands:

```
device# configure terminal
device (config)# system-max vlan 4094
```

The example increases the available port-to-VLAN associations to more than 524,000 entries.

Increasing the number of virtual routing interfaces you can configure

To increase the maximum number of virtual routing interfaces you can configure, enter commands such as the following at the global CONFIG level of the CLI.

```
device(config)# system-max virtual-interface 512
device(config)# write memory
device(config)# end
device# reload
```

Syntax: system-max virtual-interface *num*

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 22 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 whose 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

To configure a topology group, enter commands such as the following.

```
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-vlan 4
device(config-topo-group-2)#member-vlan 5
device(config-topo-group-2)#member-group 2
```

These commands create topology group 2 and add the following:

- Master VLAN 2
- Member VLANs 2, 3, and 4
- Member VLAN group 2

Syntax: [no] topology-group *group-id*

The *group-id* parameter specifies the topology group ID and can be from 1 - 256.

Syntax: [no] master-vlan *vlan-id*

This command adds the master VLAN. The VLAN must already be configured. Make sure all the Layer 2 protocol settings in the VLAN are correct for your configuration before you add the VLAN to the topology group. A topology group can have only one master VLAN.

NOTE
When removing the master VLAN from the topology group, spanning tree is disabled on the master VLAN.

NOTE
If you remove the master VLAN (by entering **no master-vlan** *vlan-id*), the software selects the new master VLAN from member VLANs. For example, if you remove master VLAN 2 from the example above, the CLI converts member VLAN 3 into the new master VLAN. The new master VLAN inherits the Layer 2 protocol settings of the older master VLAN.

NOTE
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.

Syntax: [no] member-vlan *vlan-id*

The *vlan-id* parameter specifies a VLAN ID. The VLAN must already be configured.

Syntax: [no] member-group *num*

The *num* specifies a VLAN group ID. The VLAN group must already be configured.

NOTE
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 group is deleted. For example, if STP is configured on a VLAN and you add the VLAN to a topology group, the STP configuration is removed from the VLAN. Once you add the VLAN to a topology group, the VLAN uses the Layer 2 protocol settings of the master VLAN. 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.

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
```

Syntax: show topology-group [*group-id*]

This display shows the following information.

TABLE 23 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.
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.

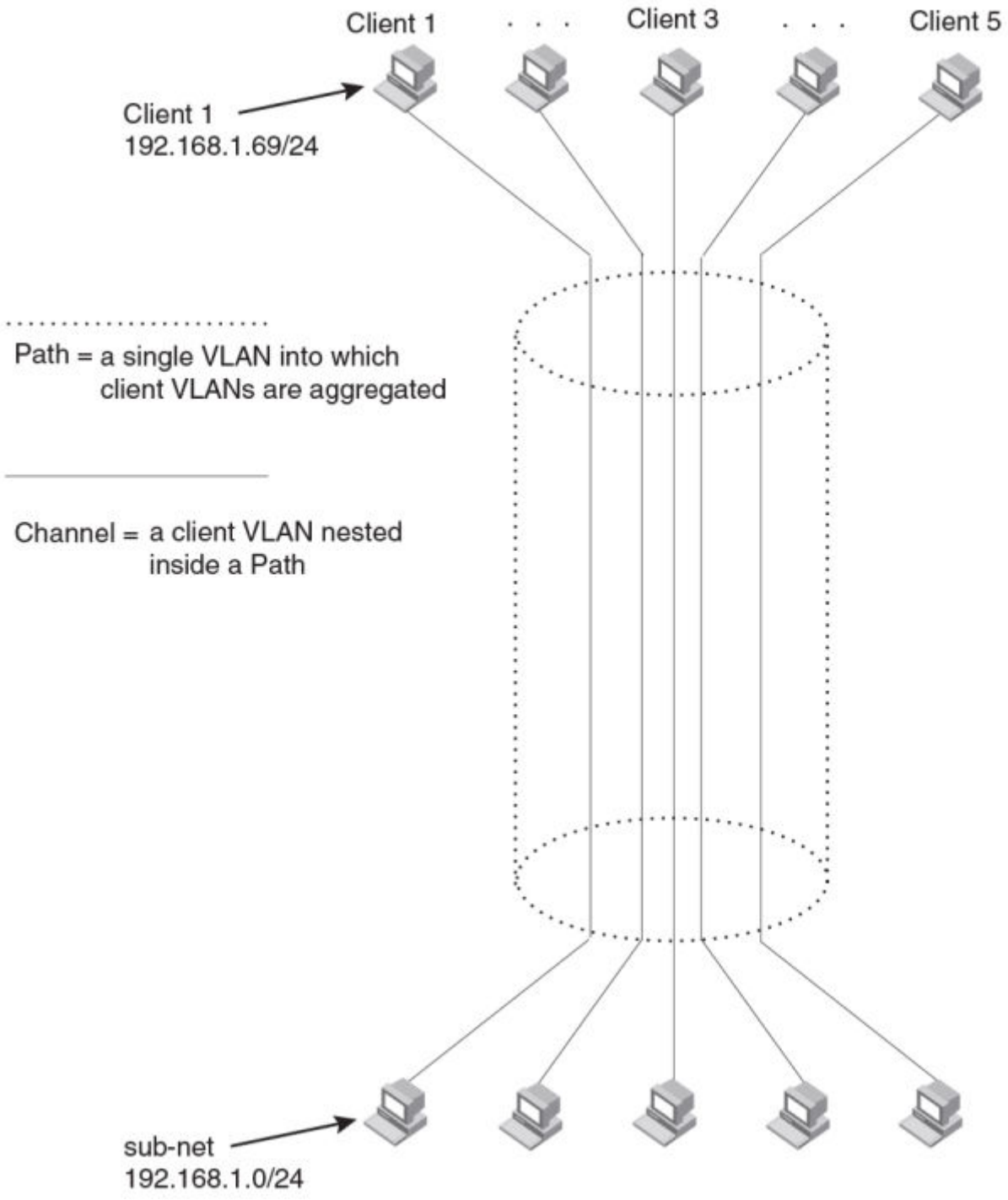
You can aggregate up to 4094 VLANs within another VLAN. This provides a total VLAN capacity on one Ruckus device of 16,760,836 channels (4094 * 4094).

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 90 Conceptual model of the super aggregated VLAN application

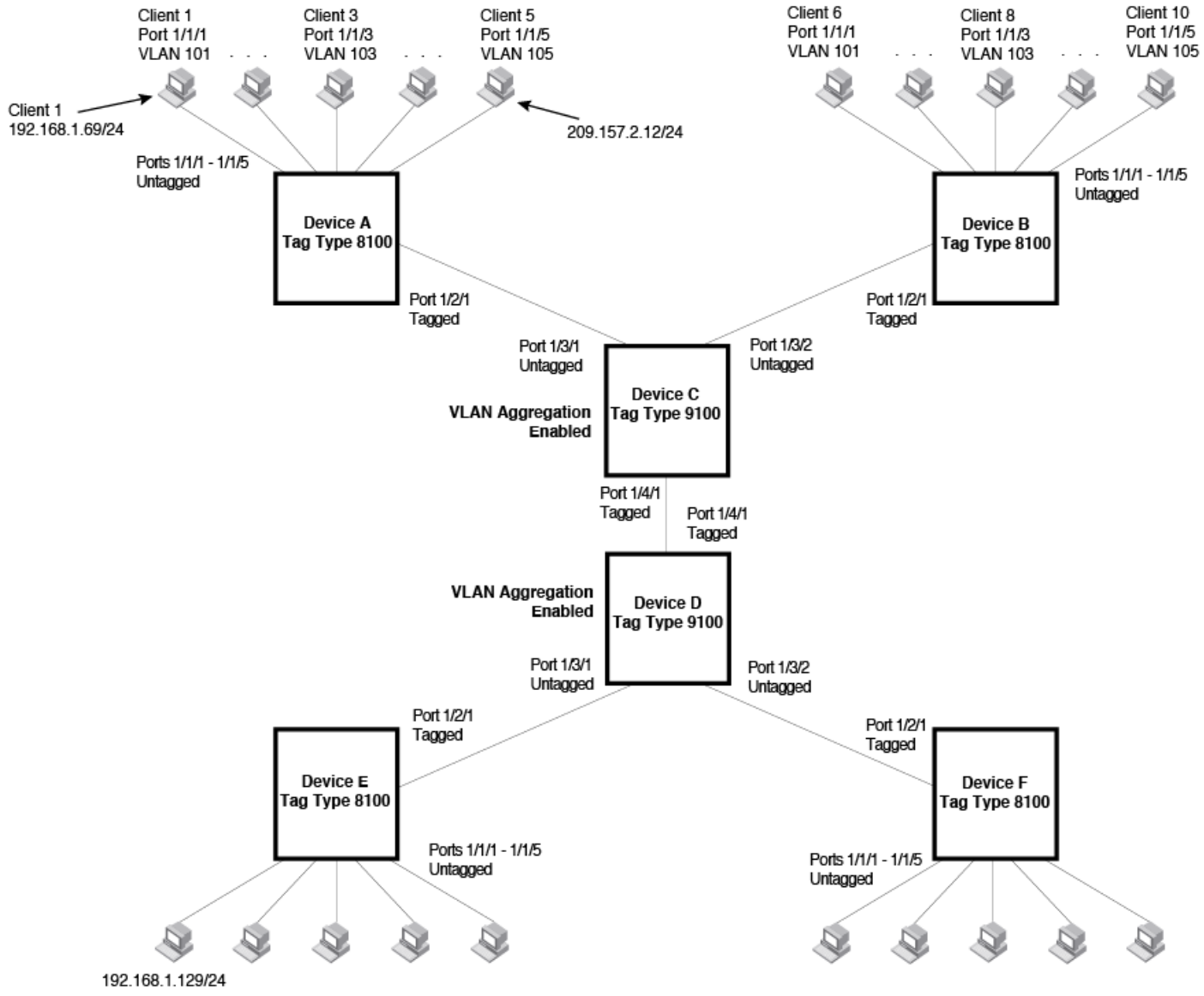


Each client connected to the edge device is in its own port-based VLAN, which is like an ATM channel. All the clients' VLANs are aggregated by the edge device into a single VLAN for connection to the core. The single VLAN that aggregates the clients' VLANs is like an ATM path.

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 90.

FIGURE 91 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 type (tag ID) that is different than the tag type used on the edge devices. If you use the default tag type (8100) on the edge devices, set the tag type on the core devices to another value, such as 9100. The tag type must be the same on all the core devices. The edge devices also must have the same tag type but the type must be different from the tag type 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 91 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
```

Syntax: [no] vlan *vlan-id* [**by port**]

Syntax: [no] tagged ethernet *unit*/*slotnum*/*portnum* [**to** *unit*/*slotnum*/*portnum* | **ethernet** *unit*/*slotnum*/*portnum*]

Syntax: [no] untagged ethernet *unit*/*slotnum*/*portnum* [**to** *unit*/*slotnum*/*portnum* | **ethernet** *unit*/*slotnum*/*portnum*]

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 91 on page 295, enter the following commands.

```
device(config)# tag-type 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
```

Syntax: [no] tag-type *num*

Syntax: [no] aggregated-vlan

The num parameter specifies the tag type can be a hexadecimal value from 0 - ffff. The default is 8100.

NOTE
tag-type is not applicable to ICX 7xxx devices. However, the same functionality can be achieved using the tag-profile CLI.

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 91 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 91 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
vA(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 type and enable VLAN aggregation.

```
deviceC(config)# tag-type 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 type must be the same as tag type configured on the other core device (Device C). In addition, VLAN aggregation also must be enabled.

```
deviceD(config)# tag-type 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 91 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 91 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
```

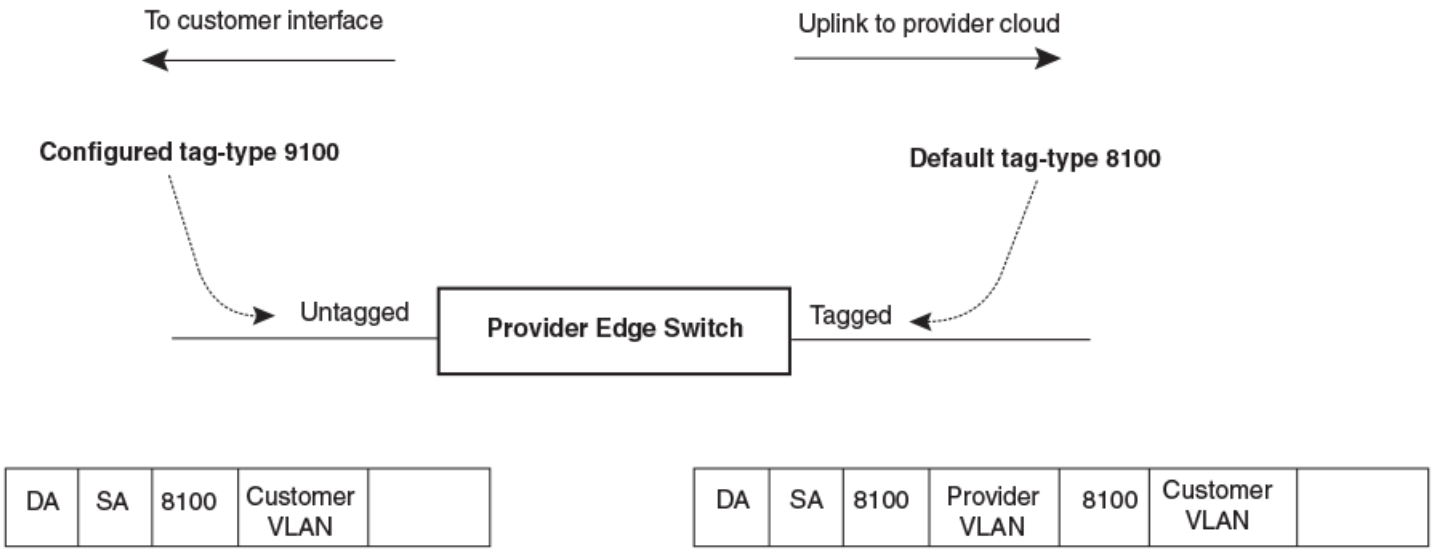
802.1ad tagging configuration

802.1ad tagging provides finer granularity for configuring 802.1Q tagging, enabling you to configure 802.1Q tag-types on a group of ports. 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-types, but are not very flexible with the tag-types 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 92 802.1ad 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-type 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-type on the uplink port is 8100, so the Ruckus device will switch the frames to the uplink device with an additional 8100 tag, thereby supporting devices that only support this method of VLAN tagging.

Configuration rules for 802.1ad tagging

- There are two ways to achieve q-in-q in FastIron:
  - Using the tag-type option in the CLI: When you enable tag-type on one port, it is applied to complete port-region (in most cases port-region can be seen as a single device). Likewise, if you remove the 802.1Q tag-type from a port, the Ruckus device automatically removes the 802.1Q tag-type from all ports within the same port region. To achieve q-in-q using tag-type, ingress port and egress port have to be in different port-regions. So FI units, which have only one port regions per device, cannot achieve q-in-q using tag-type CLI.

    NOTE
    This option is not applicable to the ICX 7xxx series devices. Only the tag-profile option is supported.

  - Using the tag-profile option in the CLI: 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 in ingress and egress ports are in a single device. The tag-profile provides more functionality compared to tagtype and should be the preferred method to enable q-in-q.
- The ICX 7xxx devices support tag-profile.
- **NOTE**
  802.1ad tagging and L2 protocols are not supported on tag profile enabled ports or port regions.
- In addition to **tag-type** Ruckus ICX devices support **tag-profile**. For more information, refer to Configuring 802.1ad tag profiles on page 303.

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 93 on page 303, the 802.1Q tag on the untagged edge links (ports 11 and 12) is 9100, whereas, the 802.1Q tag for incoming traffic is 8100.

To configure 802.1 ad tagging as shown in Figure 93 on page 303, enter commands such as the following on the untagged edge links of devices C and D.

```
device(config)# tag-type 9100 ethernet 11 to 12
device(config)# aggregated-vlan
```

Note that because ports 11 and 12 belong to the port region 1 - 12, the 802.1Q tag actually applies to ports 1 - 12.

Syntax: [no] tag-type *num* [**ethernet** *port* [**to** *port*]]

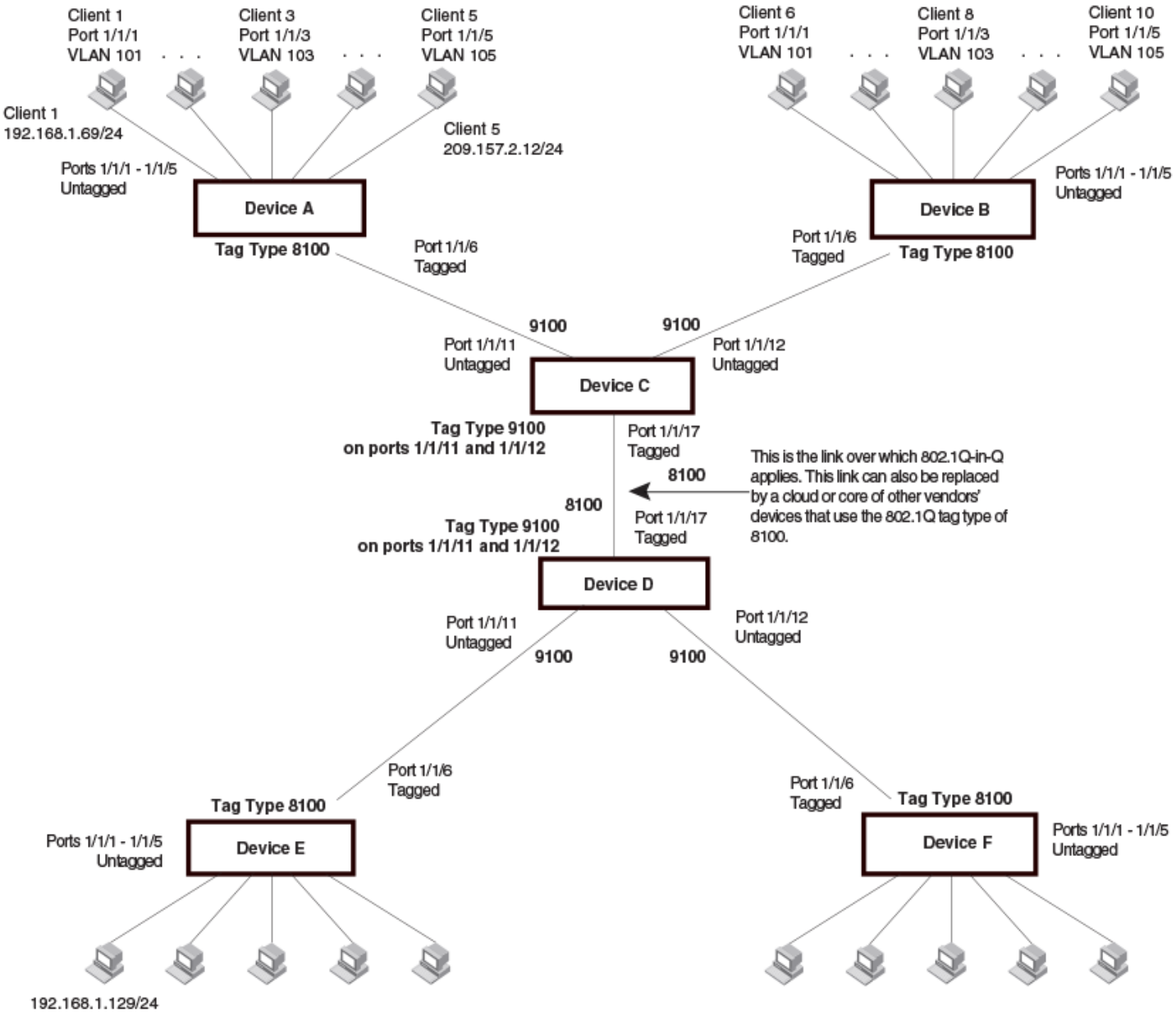
The **ethernet** *port* to *port* parameter specifies the ports that will use the defined 802.1Q tag. This parameter operates with the following rules:

- If you specify a single port number, the 802.1Q tag applies to all ports within the port region. For example, if you enter the command **tag-type 9100 ethernet 1** , the Ruckus device automatically applies the 802.1Q tag to ports 1 - 12 because all of these ports are in the same port region. You can use the **show running-config** command to view how the command has been applied.
- If you do not specify a port or range of ports, the 802.1Q tag applies to all Ethernet ports on the device.

Example 802.1ad configuration

The following figure shows an example 802.1ad configuration.

FIGURE 93 Example 802.1ad configuration



Configuring 802.1ad tag profiles

The 802.1ad tagging feature supports a **tag-profile** command that allows you to add a tag profile with a value of 0 to 0xffff in addition to the default tag-type 0x8100. This enhancement also allows you to add a tag profile for a single port, or to direct a group of ports to a globally-configured tag profile.

Configuration notes for 802.1ad tagging

- One global tag profile with a number between 0 and 0xffff can be configured on stackable devices.

- On individual ports, if **tag-profile** is enabled, it points to the global tag profile.
- **Tag-profile** can also be enabled for provisional ports.
- **Tag-type** and **tag-profile** cannot be configured at the same time. You will see the message "un-configure the tag-type to set the tag-profile." It **tag-type** is already configured, you will need to unconfigure it and then add the **tag-profile** .
- Do not use the **tag-type** command in conjunction with the **tag-profile** command. If a **tag-type** has already been configured and you try to use the **tag-profile** command, you will see an error message telling you to remove the tag-type before you add the tag-profile.
- For devices operating in an IronStack topology, when a tag-type for a port is changed, the tag-type for all of the ports on a stack unit also changes. Because of this limitation, SAV and Q-in-Q cannot be used at the same time on stacking devices.

CLI Syntax for 802.1ad tagging

To add a global tag-profile enter the following command.

```
device(config)# tag-profile 9500
```

This command adds a profile in addition to the default profile of 0x8100.

Syntax: [no] tag-profile *tag-no*

where tag-no can be 0x8100 (the default) or 0xffff.

To enable the new profile on individual ports, enter commands similar to the following.

```
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
```

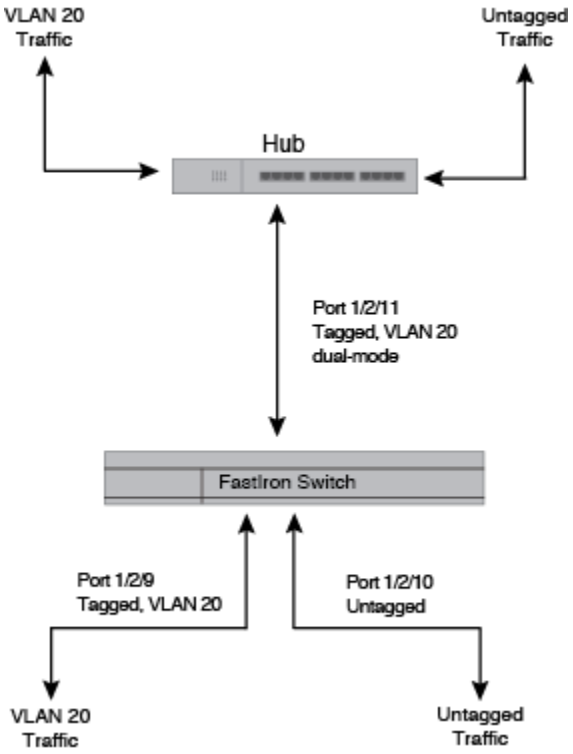
Syntax: [no] tag-profile enable

Dual-mode VLAN ports

Configuring a tagged port as a dual-mode port allows it to accept and transmit both tagged traffic and untagged traffic at the same time. A dual-mode port accepts and transmits frames belonging to VLANs configured for the port, as well as frames belonging to the default VLAN (that is, untagged traffic).

For example, in the following figure, port 1/2/11 is a dual-mode port belonging to VLAN 20. Traffic for VLAN 20, as well as traffic for the default VLAN, flows from a hub to this port. The dual-mode feature allows traffic for VLAN 20 and untagged traffic to go through the port at the same time.

FIGURE 94 Dual-mode VLAN port example



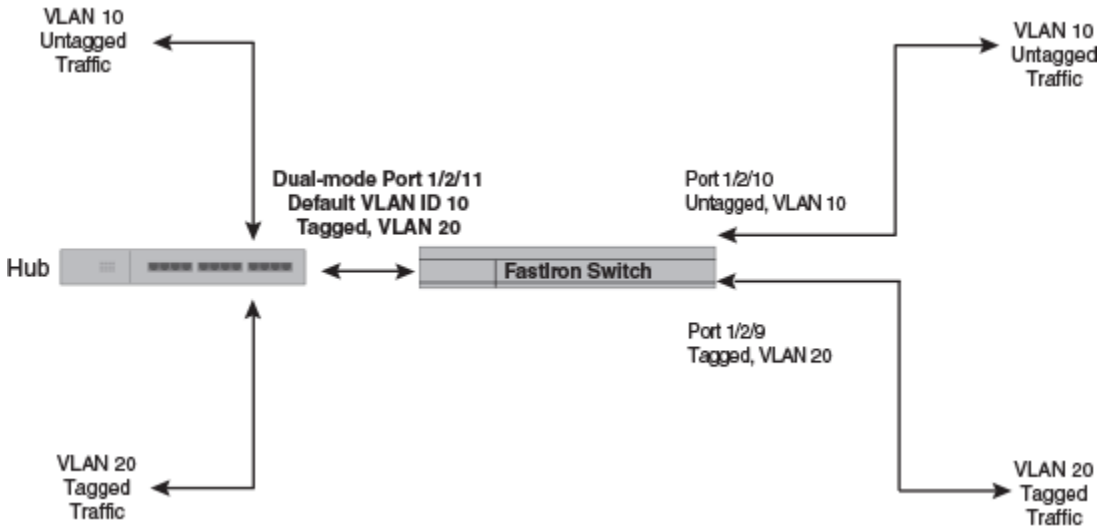
To enable the dual-mode feature on port 1/2/11 in the above figure, enter the following commands.

```
device(config)# vlan 20
device(config-vlan-20)# tagged ethernet 1/2/11
device(config-vlan-20)# tagged ethernet 1/2/9
device(config-vlan-20)# interface ethernet 1/2/11
device(config-if-e1000-1/2/11)# dual-mode
device(config-if-e1000-1/2/11)# exit
```

Syntax: [no] dual-mode

You can configure a dual-mode port to transmit traffic for a specified VLAN (other than the DEFAULT-VLAN) as untagged, while transmitting traffic for other VLANs as tagged. The following figure illustrates this enhancement.

FIGURE 95 Specifying a default VLAN ID for a dual-mode port



In the above figure, tagged port 1/2/11 is a dual-mode port belonging to VLANs 10 and 20. The default VLAN assigned to this dual-mode port is 10. This means that the port transmits tagged traffic on VLAN 20 (and all other VLANs to which the port belongs) and transmits untagged traffic on VLAN 10.

The dual-mode feature allows tagged traffic for VLAN 20 and untagged traffic for VLAN 10 to go through port 1/2/11 at the same time. A dual-mode port transmits only untagged traffic on its default VLAN (that is, either VLAN 1, or a user-specified VLAN ID), and only tagged traffic on all other VLANs.

The following commands configure VLANs 10 and 20 in Figure 95. Tagged port 1/2/11 is added to VLANs 10 and 20, then designated a dual-mode port whose specified default VLAN is 10. In this configuration, port 1/2/11 transmits only untagged traffic on VLAN 10 and only tagged traffic on VLAN 20.

```
device(config)# vlan 10 by port
device(config-vlan-10)# untagged ethernet 1/2/10
device(config-vlan-10)# tagged ethernet 1/2/11
device(config-vlan-10)# exit
device(config)# vlan 20 by port
device(config-vlan-20)# tagged ethernet 1/2/9
device(config-vlan-20)# tagged ethernet 1/2/11
device(config-vlan-20)# exit
device(config)# interface ethernet 1/2/11
device(config-if-e1000-1/2/11)# dual-mode 10
device(config-if-e1000-1/2/11)# exit
```

You can configure multiple ports using commads such as the following.

```
device# interface ethernet 1/1/6 to 1/1/9
device(config-mif-1/1/6-1/1/9)# dual-mode
```

NOTE
An error message is displayed while attempting to configure an existing dual-mode on a port range.

Example:

```
Port 1/1/6 has already been configured as dual mode on VLAN 20
Port 1/1/7 has already been configured as dual mode on VLAN 20
Port 1/1/8 has already been configured as dual mode on VLAN 20
Port 1/1/9 has already been configured as dual mode on VLAN 1
```

Notes:

- If you do not specify a *vlan-id* in the **dual mode** command, the port default VLAN is set to 1. The port transmits untagged traffic on the DEFAULT-VLAN.
- The dual-mode feature is disabled by default. Only tagged ports can be configured as dual-mode ports.
- In trunk group, either all of the ports must be dual-mode, or none of them can be.

The **show vlan** command displays a separate row for dual-mode ports on each VLAN.

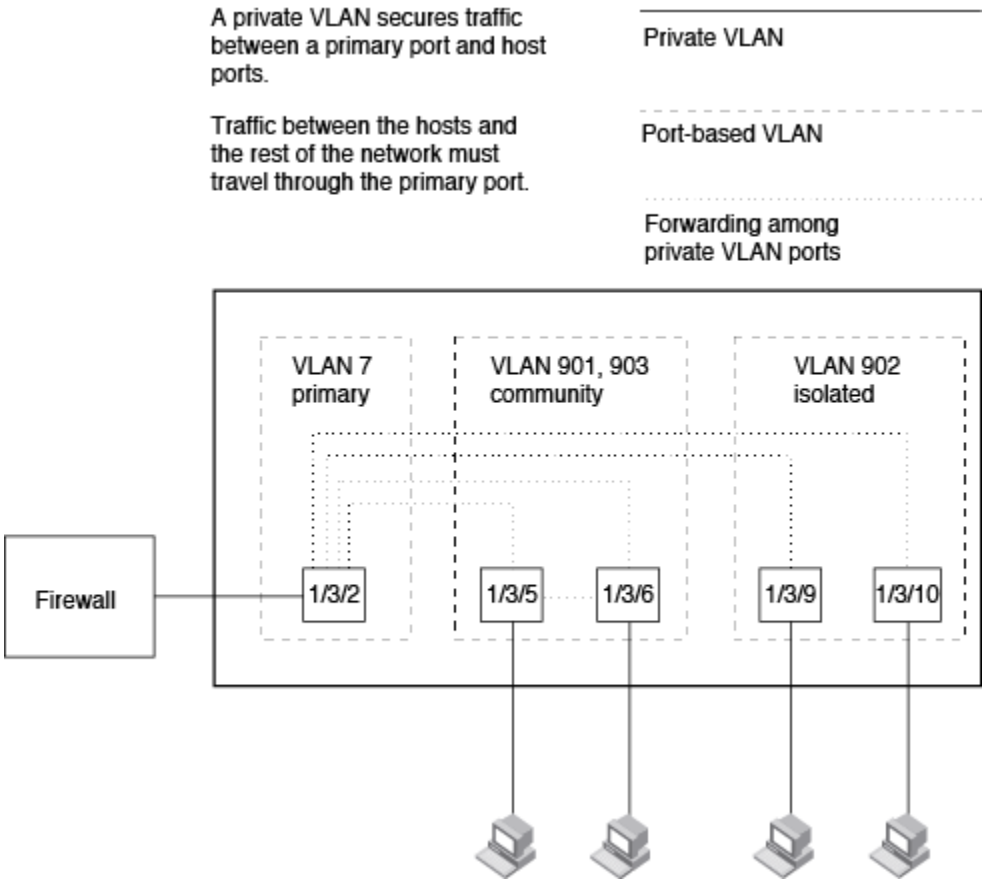
```
device# show vlan
Total PORT-VLAN entries: 3
Maximum PORT-VLAN entries: 16
legend: [S=Slot]
PORT-VLAN 1, Name DEFAULT-VLAN, Priority level0, Spanning tree Off
 Untagged Ports: (S1)  1  2  3  4  5  6  7  8
 Untagged Ports: (S2)  1  2  3  4  5  6  7  8 12 13 14 15 16 17 18 19
 Untagged Ports: (S2) 20 21 22 23 24
   Tagged Ports: None
   Uplink Ports: None
 DualMode Ports: None
PORT-VLAN 10, Name [None], Priority level0, Spanning tree Off
 Untagged Ports: (S2) 10
   Tagged Ports: None
   Uplink Ports: None
 DualMode Ports: (S2) 11
PORT-VLAN 20, Name [None], Priority level0, Spanning tree Off
 Untagged Ports: None
   Tagged Ports: (S2)  9
   Uplink Ports: None
 DualMode Ports: (S2) 11
```

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 96 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.

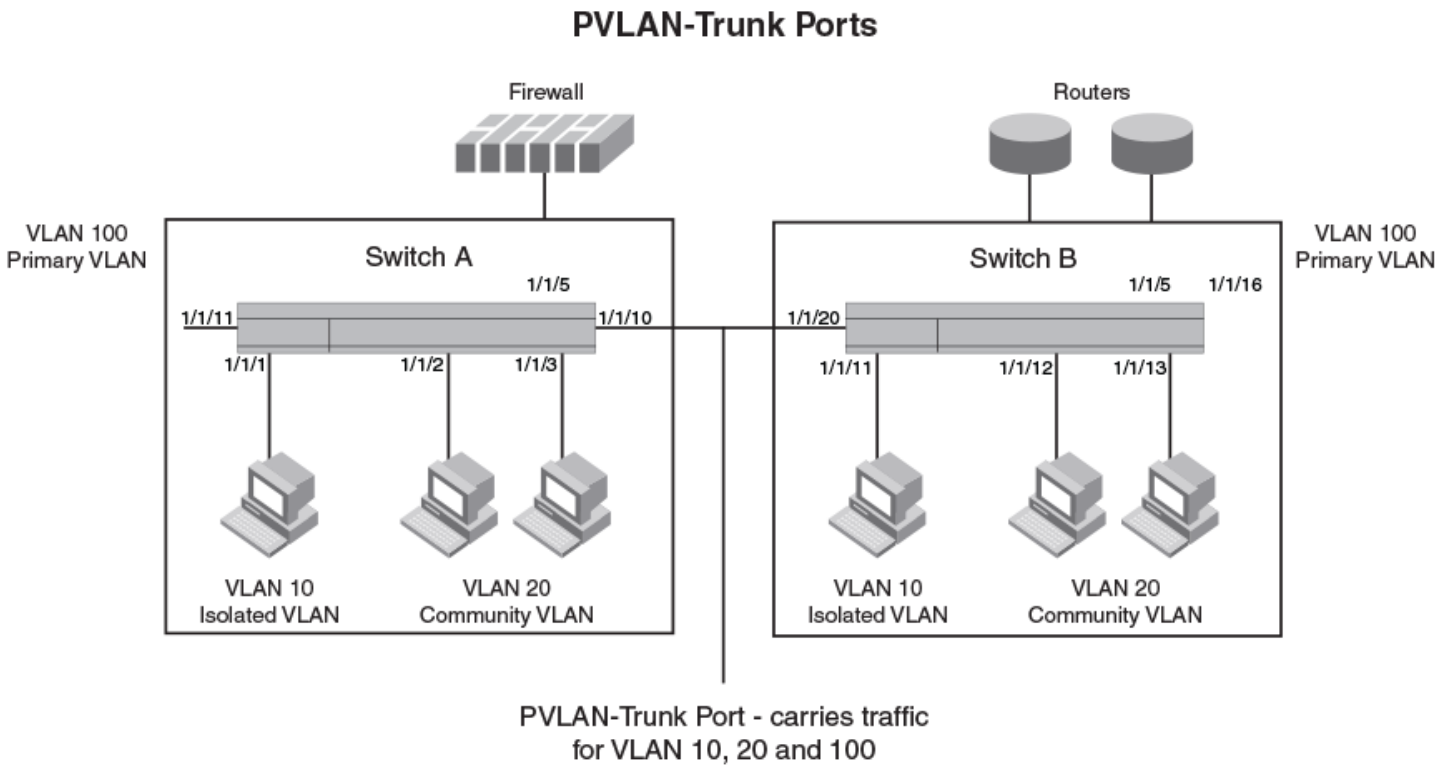
As with regular VLANs, PVLANs can span multiple switches. The PVLAN is treated like any other VLAN by the PVLAN-trunk ports.

NOTE
ISL(Inter-Switch Link) is an alias for PVLAN-trunk ports.

Figure 97 shows an example of a PVLAN network across switches:

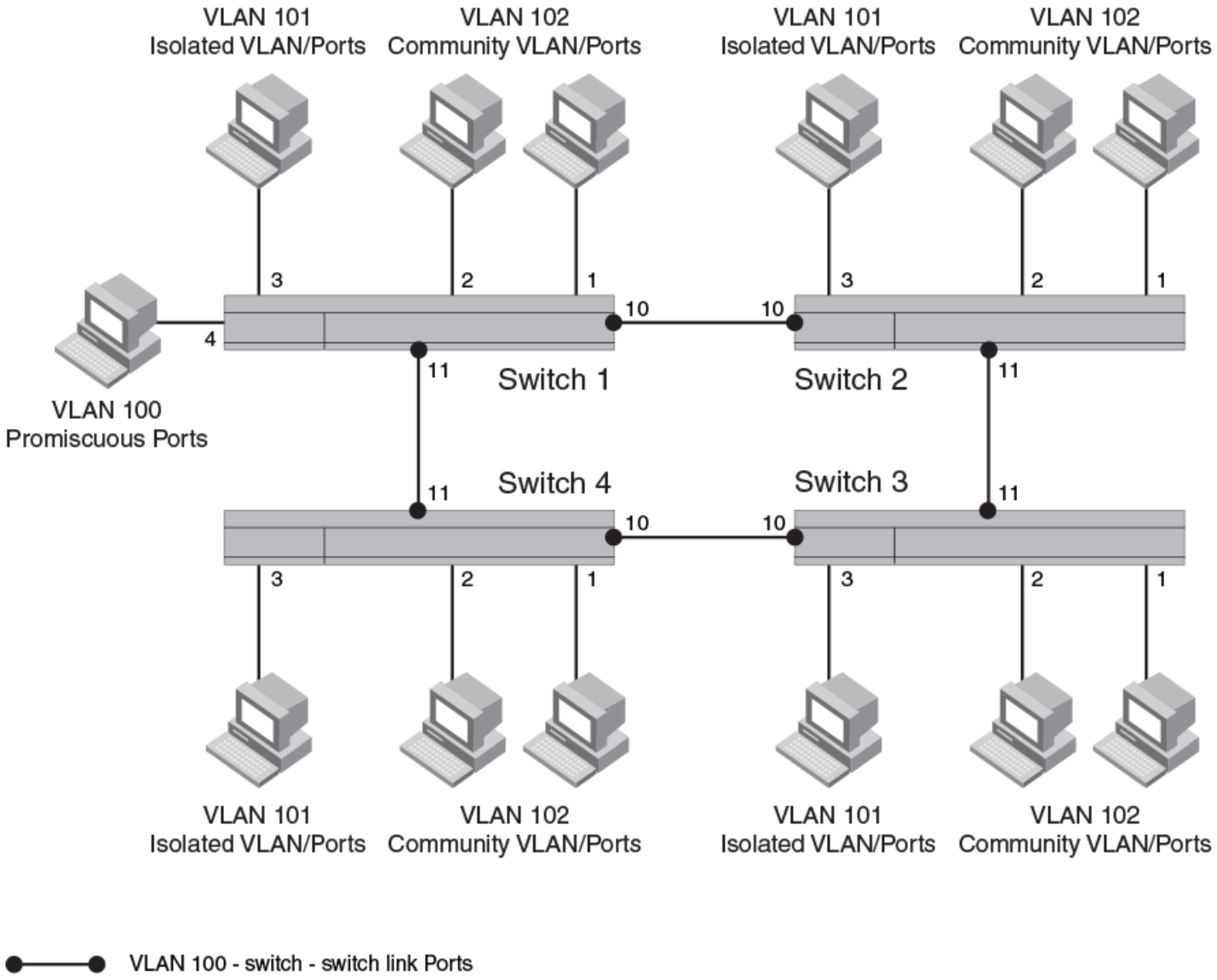
- Broadcast, unknown unicast or unregistered multicast traffic from the primary VLAN port is forwarded to all ports in isolated and community VLANs in both the switches.
- Broadcast, unknown unicast or unregistered multicast traffic from an isolated port in switch A is not forwarded to an isolated port 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 97 PVLAN across switches



The following figure shows an example PVLAN network with tagged switch-switch link ports.

FIGURE 98 Example PVLAN network with tagged ports



The following table lists the differences between PVLANs and standard VLANs.

TABLE 24 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 24 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. Configuring a tagged port as a dual-mode port allows it to accept and transmit both tagged and untagged traffic at the same time. A dual-mode port accepts and transmits frames belonging to VLANs configured for the port as well as frames belonging to the port default VLAN (untagged traffic).

NOTE
After adding secondary VLAN ports to multiple secondary VLAN ports of the same type, if the Ruckus device is downgraded to FastIron 08.0.40 or older versions, errors are generated during parsing as this is not supported prior to FastIron 08.0.50 releases.

Private VLANs allow Layer 2 segregation and also minimizes usage of system VLANs. Using 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 either not be allowed or will generate a warning message.

TABLE 25 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	Dual-mode	Promiscuous ports	ISL ports				
Promiscuous ports	Yes	Yes	Yes	Yes	No	No	No	No	No
Inter Switch Link (ISL) ports	Yes	No	No	No	Yes	No	No	No	No
Isolated VLAN ports	Yes	Yes	Yes	NA	NA	Yes	No	No	No
Community VLAN ports	Yes	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 public)
  - 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 dual-mode VLAN ports on the same device. However, the dual-mode VLAN ports, other than those which are dual-mode in system default VLAN, can be member ports in a PVLAN domain.
- 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.
- Dual mode ports are supported in a private VLAN domain. However, since ISL ports can only be tagged ports, they cannot be enabled on 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 either not be allowed or will generate a warning message.
- PVST, when needed in PVLANs, should be enabled on all (primary and secondary) private VLANs across switches.
- LAG is not supported in any PVLAN.
- Port MAC security is not supported on ports in a private VLAN domain.

TABLE 26 PVLAN support matrix

Platform	Forwarding Type	Tagged Port	Untagged Port	ISL Port	Multiple Promiscuous Port
ICX 7250	Hardware	Yes	Yes	Yes	Yes
ICX 7450	Hardware	Yes	Yes	Yes	Yes
ICX 7750	Hardware	Yes	Yes	Yes	Yes

Configuring an isolated or community PVLAN

You can use the **pvlan type** command to configure the PVLAN as an isolated or community PVLAN. The following are some configuration considerations to be noted for configuring isolated and community PVLANs.

Isolated VLANs

- Every isolated VLAN should be in a unique primary VLAN domain.
- A port being added to the isolated VLAN can be either a tagged port or an untagged port or a dual-mode port.
- 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.

To configure an isolated PVLAN, enter commands such as the following.

```
device(config)# vlan 901
device(config-vlan-901)# untagged ethernet 1/3/5 to 1/3/6
device(config-vlan-901)# pvlan type isolated
```

Community VLANs

- Every community VLAN should be in a unique primary VLAN domain.
- A port being added to the community VLAN can be either a tagged port or an untagged port or a 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

To configure a community PVLAN, enter commands such as the following.

```
device(config)# vlan 901
device(config-vlan-901)# untagged ethernet 1/3/5 to 1/3/6
device(config-vlan-901)# pvlan type community
```

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.

Syntax: untagged ethernet *unit**/**slotnum**/**portnum* [**to** *unit**/**slotnum**/**portnum* | **ethernet** *unit**/**slotnum**/**portnum*]

or

Syntax: tagged ethernet *unit**/**slotnum**/**portnum* [**to** *unit**/**slotnum**/**portnum* | **ethernet** *unit**/**slotnum**/**portnum*]

Syntax: [no] pvlan type community | **isolated** | **primary**

The **untagged ethernet** or **taggd 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 the primary VLAN

To configure a primary VLAN, enter commands such as the following.

```
device(config)# vlan 7
device(config-vlan-7)# pvlan type primary
device(config-vlan-7)# untagged ethernet 1/3/2
device(config-vlan-7)# pvlan mapping 901 ethernet 1/3/2
```

These commands create port-based VLAN 7, add port 1/3/2 as an untagged port, identify the VLAN as the primary VLAN in a PVLAN, and map the other secondary VLANs to the ports in this VLAN.

To map the secondary VLANs to the primary VLAN and to configure the tagged switch link port, enter commands such as the following.

```
device(config)# vlan 100
device(config-vlan-100)# tagged ethernet 1/1/10 to 1/1/11
device(config-vlan-100)# pvlan type primary
device(config-vlan-100)# untagged ethernet 1/1/4
device(config-vlan-100)# pvlan mapping 101 ethernet 1/1/4
device(config-vlan-100)# pvlan mapping 102 ethernet 1/1/4
device(config-vlan-100)# pvlan pvlan-trunk 101 ethernet 1/1/10 to 1/1/11
```

These commands create port-based VLAN 100, add port 1/1/10 to 1/1/11 as a tagged port, identify the VLAN as the primary VLAN in a PVLAN, map the other secondary VLANs to the ports in this VLAN, and configure the tagged switch link port.

Syntax: untagged ethernet [*stack-unit***/***slotnum***/**]*portnum* [**to** [*stack-unit***/***slotnum***/**]*portnum* | **ethernet** [*stack-unit***/***slotnum***/**]*portnum*]

or

Syntax: tagged ethernet [*stack-unit***/***slotnum***/**]*portnum* [**to** [*stack-unit***/***slotnum***/**]*portnum* | **ethernet**[*stack-unit***/***slotnum***/**]*portnum*]

Syntax: [**no**] **pvlan type community** | **isolated** | **primary**

Syntax: [no] pvlan mapping *vlan-id* ethernet [*stack-unit***/***slotnum***/**]*portnum*

Syntax: [no] pvlan pvlan-trunk *vlan-id* ethernet [*stack-unit***/***slotnum***/**]portnum [**to** [*stack-unit***/***slotnum***/**]*portnum*]

The **untagged** or **tagged** command adds the ports to the VLAN.

The **pvlan type** command specifies that this port-based VLAN is a PVLAN. Specify **primary** as the type.

The **pvlan mapping** command identifies the other PVLANs for which this VLAN is the primary. The command also specifies the primary VLAN ports to which you are mapping the other secondary VLANs. The mapping command is not allowed on the secondary VLANs. The parameters of the **pvlan mapping** command are as follows:

- The *vlan-id* parameter specifies another PVLAN. The other PVLAN you want to specify must already be configured.
- The **ethernet** *portnum* parameter specifies the primary VLAN port to which you are mapping all the ports in the other PVLAN (the one specified by vlan-id).

The **pvlan pvlan-trunk** command identifies the switch-switch link for the PVLAN. There can be more than one switch-switch link for a single community VLAN.

NOTE
The **pvlan pvlan-trunk** command is not allowed on the secondary VLANs.

CLI example for a general PVLAN network

To configure the PVLANs shown in Figure 96 on page 308, enter the following commands.

```
device(config)# vlan 901
device(config-vlan-901)# untagged ethernet 1/3/5 to 1/3/6
device(config-vlan-901)# pvlan type community
device(config-vlan-901)# exit
device(config)# vlan 902
device(config-vlan-902)# untagged ethernet 1/3/9 to 1/3/10
device(config-vlan-902)# pvlan type isolated
device(config-vlan-902)# exit
device(config)# vlan 903
device(config-vlan-903)# untagged ethernet 1/3/7 to 1/3/8
device(config-vlan-903)# pvlan type community
device(config-vlan-903)# exit
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 dual-mode PVLAN network

To configure the dual-mode PVLAN network, enter the following commands.

```
device(config)# vlan 101 by port
device(config-vlan-101)# pvlan type isolated
device(config-vlan-101)# tagged ethernet 1/1/25 ethernet 1/1/35
Added tagged port(s) ethernet 1/1/25 ethernet 1/1/35 to port-vlan 101.
device(config-vlan-101)# spanning-tree
device(config-vlan-101)#!
device(config-vlan-101)# vlan 102 by port
device(config-vlan-102)# pvlan type community
device(config-vlan-102)# spanning-tree
device(config-vlan-102)#!
device(config-vlan-102)# vlan 103 by port
device(config-vlan-103)# pvlan type community
device(config-vlan-103)# spanning-tree
device(config-vlan-103)#!
device(config-vlan-103)# vlan 100 by port
device(config-vlan-100)# pvlan type primary
device(config-vlan-100)# tagged ethernet 1/1/20 to 1/1/21 ethernet 1/1/31
Added tagged port(s) ethernet 1/1/20 to 1/1/21 ethernet 1/1/31 to port-vlan 100.
device(config-vlan-100)# pvlan mapping 102 ethernet 1/1/20
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)#!
device(config-vlan-100)# vlan 201 by port
device(config-vlan-201)# pvlan type isolated
device(config-vlan-201)# tagged ethernet 1/1/25 ethernet 1/1/35
Added tagged port(s) ethernet 1/1/25 ethernet 1/1/35 to port-vlan 201.
device(config-vlan-201)# spanning-tree
device(config-vlan-201)#!
device(config-vlan-201)# vlan 202 by port
device(config-vlan-202)# pvlan type community
device(config-vlan-202)# spanning-tree
device(config-vlan-202)#!
device(config-vlan-202)# vlan 203 by port
device(config-vlan-203)# pvlan type community
device(config-vlan-203)# spanning-tree
device(config-vlan-203)#!
device(config-vlan-203)# vlan 200 by port
```

```
device(config-vlan-200)# pvlan type primary
device(config-vlan-200)# tagged ethernet 1/1/20 to 1/1/21 ethernet 1/1/31
Added tagged port(s) ethernet 1/1/20 to 1/1/21 ethernet 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)#!
device(config-vlan-200)# interface ethernet 1/1/25 ethernet 1/1/35
device(config-mif-1/1/25,1/1/35)# dual-mode 201
device(config-mif-1/1/25,1/1/35)# interface ethernet 1/1/20
device(config-if-e10000-1/1/20)# dual 200
device(config-if-e10000-1/1/20)#
```

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
```

Displaying VLAN information

After you configure the VLANs, you can verify the configuration using the **show** commands described in this section.

NOTE
If a VLAN name begins with "GVRP_VLAN_", the VLAN was created by the GARP VLAN Registration Protocol (GVRP). If a VLAN name begins with "STATIC_VLAN_", the VLAN was created by GVRP and then was converted into a statically configured VLAN.

Displaying VLANs in alphanumeric order

By default, VLANs are displayed in alphanumeric order, as shown in the following example.

```
device# show run
...
vlan 2 by port
...
vlan 10 by port
...
vlan 100 by port
...
```

Displaying system-wide VLAN information

Use the **show vlans** command to display VLAN information for all the VLANs configured on the device.

The following example shows the display for the configured IP subnet VLANs.

```
device# show vlans
Total PORT-VLAN entries: 2
Maximum PORT-VLAN entries: 8
legend: [S=Slot]
PORT-VLAN 1, Name DEFAULT-VLAN, Priority level0, Spanning tree Off
 Untagged Ports: (S2)  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16
 Untagged Ports: (S2) 17 18 19 20 21 22 23 24
 Untagged Ports: (S4)  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16
 Untagged Ports: (S4) 17 18 19 20 21 22 23 24
   Tagged Ports: None
PORT-VLAN 10, Name IP_VLAN, Priority level0, Spanning tree Off
 Untagged Ports: (S1)  1  2  3  4  5  6
   Tagged Ports: None
 IP-subnet VLAN 10.1.1.0 255.255.255.0, Dynamic port enabled
          Name: Mktg-LAN
  Static ports: None
 Exclude ports: None
 Dynamic ports: (S1)  1  2  3  4  5  6
```

In the **show vlans** output, ports that are tagged but are not dual-mode ports are listed as tagged ports. In the following example display output, ports 7 and 8 are dual-mode ports in port-based VLAN 4. Ports 7 and 8 also belong to port-based VLAN 3, but they are tagged ports only in VLAN 3 and are not configured as dual-mode ports.

```
device# show vlan 4
Total PORT-VLAN entries: 5
Maximum PORT-VLAN entries: 3210
PORT-VLAN 4, Name [None], Priority level0, Spanning tree Off
 Untagged Ports: None
   Tagged Ports:   6   9  10  11
   Uplink Ports: None
 DualMode Ports:   7   8
ESX624FE+2XG Router# show vlan 3
Total PORT-VLAN entries: 5
Maximum PORT-VLAN entries: 3210
PORT-VLAN 3, Name [None], Priority level0, Spanning tree Off
 Untagged Ports: None
   Tagged Ports:   6   7   8   9  10
   Uplink Ports: None
 DualMode Ports: None
```

Syntax: show vlans [*vlan-id* | **ethernet** *unit**/**slotnum**/**portnum*]

The *vlan-id* parameter specifies a VLAN for which you want to display the configuration information.

The *slotnum* parameter is required on chassis devices.

The *portnum* parameter specifies a port. If you use this parameter, the command lists all the VLAN memberships for the port.

Displaying global VLAN information

The **show vlan brief** command displays the following information:

- The system-max VLAN values (maximum, default, and current)
- The default VLAN ID number
- The total number of VLANs configured on the device
- The VLAN ID numbers of the VLANs configured on the device

The following shows example output.

```
device#  show vlan brief
System-max vlan Params: Max(4095) Default(64) Current(3210)
Default vlan Id :1
Total Number of Vlan Configured :5
VLANs Configured :1 to 4 10
```

Syntax: show vlan brief

Displaying VLAN information for specific ports

Use one of the following methods to display VLAN information for specific ports.

To display VLAN information for all the VLANs of which port 1/7/1 is a member, enter the following command.

```
device# show vlans 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
```

Syntax: show vlans [*vlan-id* | **ethernet** *unit**/**slotnum**/**portnum*

The *vlan-id* parameter specifies a VLAN for which you want to display the configuration information.

The *slotnum* parameter is required on chassis devices.

The *portnum* parameter specifies a port. If you use this parameter, the command lists all the VLAN memberships for the port.

Displaying a port VLAN membership

To display VLAN membership for a specific port on the device, enter a command such as the following.

```
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.

Syntax: show vlan brief ethernet *unit*/*slotnum*/*portnum*

The *slotnum* parameter is required on chassis devices.

Displaying a port dual-mode VLAN membership

The output of the **show interfaces** command lists dual-mode configuration and corresponding VLAN numbers. The following shows an example output.

```
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
  Member of 3 L2 VLANs, port is dual mode in Vlan 4
, port state is BLOCKING
```

Syntax: show interfaces ethernet *unit*/*slotnum*/*portnum* [**to** *unit*/*slotnum*/*portnum* [**ethernet** *unit*/*slotnum*/*portnum*...]]

The *slotnum* parameter is required on chassis devices.

NOTE
The port up/down time is required only for physical ports and not for loopback/ ve/ tunnel ports.

Displaying port default VLAN IDs (PVIDs)

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 dual-mode ports, the PVID is the dual-mode VLAN ID number.
- For tagged ports without dual-mode, the PVID is always Not Applicable (NA).

```
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
```

Syntax: show interfaces brief [**ethernet** *unit*/*slotnum*/*portnum* [**to** *unit*/*slotnum*/*portnum* [**ethernet** *unit*/*slotnum*/*portnum*...]]]

The *slotnum* parameter is required on chassis devices.

Displaying PVLAN information

To display the PVLAN configuration with respect to the primary VLAN and its associated secondary VLANs and to display the member ports, promiscous ports, and the switch-switch link ports of a PVLAN, enter a command such as the following.

```
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
```

Syntax: show pvlan *vid*

The *vid* variable specifies the VLAN ID of the PVLAN. If the VLAN ID is not specified, the command displays the default VLAN ID.

NOTE
The show pvlan command is not supported on software forwarding platforms.

Protected Port

- Protected port.....321
- Configuring protected port.....323

Protected port

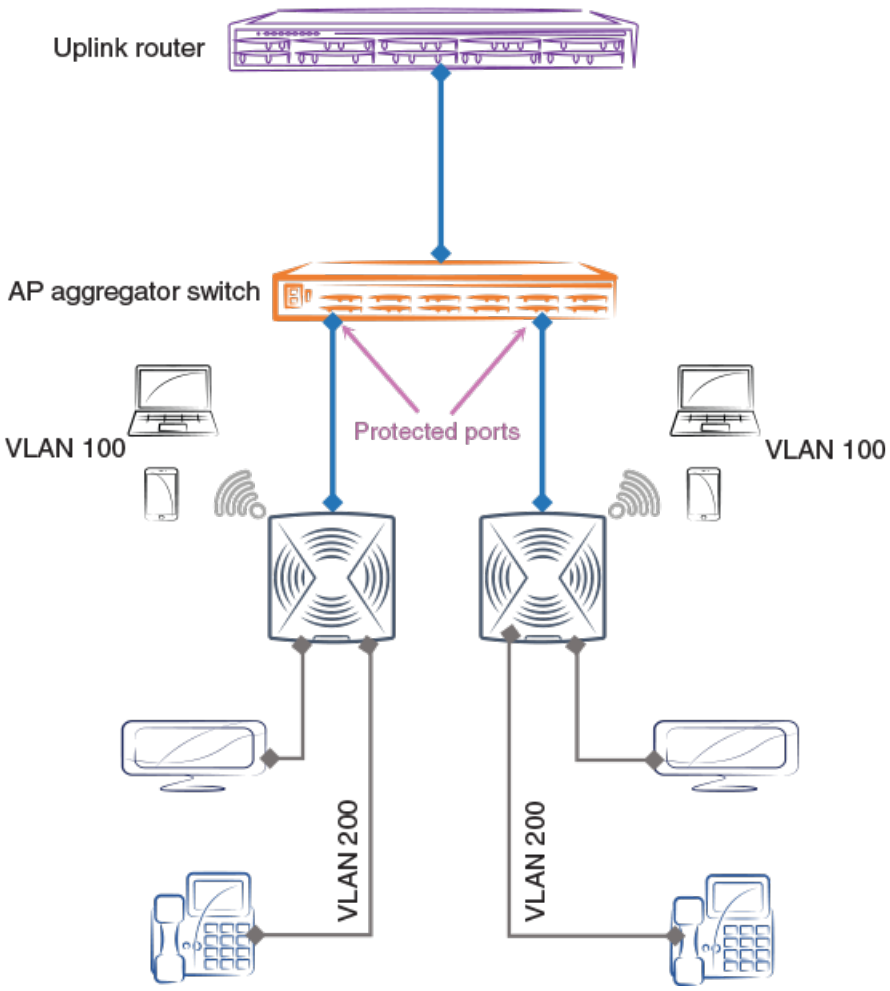
Protected ports restrict all but CPU–bound or –originated traffic, providing isolation to end hosts.

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 99 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

In addition, it is recommended that multiple ports (MIF) mode be configured.

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
- SPX provider edge (PE) ports
- SPX ZTP–enabled ports
- Multi-Chassis Trunk (MCT) ICL and CCEP ports

Configuring protected port

This task configures protected port on a single and multiple interfaces in MIF mode.

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. Do the following to enable 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
```

   You can also use ranging, as in the following example.

```
device(config)# interface ethernet 2/1/1 to ethernet 2/1/48
```

   You can also enable this feature on LAG virtual interface.

```
device(config)# interface lag 1
device(config-lag-if-lg1)# protected-port
```

6. 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
```

7. Use the **no** form of this command to disable the feature.

```
device(config-if-e1000-2/1/1,3/1/1)# no protected-port

device(config-lag-if-lg1)# no protected-port
```



© 2019 ARRIS Enterprises LLC. All rights reserved.
Ruckus Wireless, Inc., a wholly owned subsidiary of ARRIS International plc.
350 West Java Dr., Sunnyvale, CA 94089 USA
www.ruckuswireless.com