53-1002828-03
03 December 2013

Brocade MLX Series and Brocade NetIron XMR

Diagnostic Guide

Supporting Release 05.5.00c

BROCADE

Brocade Communications Systems, Incorporated

Corporate and Latin American Headquarters
Brocade Communications Systems, Inc.
130 Holger Way
San Jose, CA 95134
Tel: 1-408-333-8000
Fax: 1-408-333-8101
E-mail: info@brocade.com

Asia-Pacific Headquarters
Brocade Communications Systems China HK, Ltd.
No. 1 Guanghua Road
Chao Yang District
Units 2718 and 2818
Beijing 100020, China
Tel: +8610 6588 8888
Fax: +8610 6588 9999
E-mail: china-info@brocade.com

European Headquarters
Brocade Communications Switzerland Sàrl
Centre Swissair
Tour B - 4ème étage
29, Route de l'Aéroport
Case Postale 105
CH-1215 Genève 15
Switzerland
Tel: +41 22 799 5640
Fax: +41 22 799 5641
E-mail: emea-info@brocade.com

Asia-Pacific Headquarters
Brocade Communications Systems Co., Ltd. (Shenzhen WFOE)
Citic Plaza
No. 233 Tian He Road North
Unit 1308 – 13th Floor
Guangzhou, China
Tel: +8620 3891 2000
Fax: +8620 3891 2111
E-mail: china-info@brocade.com

Document History

Title	Publication number	Summary of changes	Date
Brocade MLX Series and Brocade NetIron XMR Diagnostic Guide	*53-1002828-01*	Release 05.4.00a document updated with enhancements in Release 05.5.00.	April 2013
Brocade MLX Series and Brocade NetIron XMR Diagnostic Guide	*53-1002828-02*	New version of the document.	July 2013
Brocade MLX Series and Brocade NetIron XMR Diagnostic Guide	*53-1002828-03*	Release 05.5.00c document updated for MP OS mode commands.	December 2013

Contents

Chapter 7 ACL and QoS Diagnostics

Chapter 8 Multicast Diagnostics

Chapter 9 **Security Diagnostics**

About This Document

In this chapter

This manual describes troubleshooting and diagnostic commands available in the IronWare command line interface (CLI) for Brocade NetIron XMR series and Brocade MLX series devices.

NOTE
Some troubleshooting commands report information about internal hardware settings and registers that is relevant primarily to the Brocade engineering staff. Consequently, this information is not described in this document.

Audience

This document is designed for system administrators with a working knowledge of Layer 2 and Layer 3 switching and routing.

If you are using a Brocade device, you should be familiar with the following protocols if applicable to your network – IP, RIP, OSPF, BGP, ISIS, IGMP, PIM, MPLS, and VRRP.

Disclaimer

This manual is provided without any warranty of any kind, expressed or implied. When using this manual to troubleshoot Brocade products, you assume all risk as to the quality and performance of the diagnostic procedures. Brocade assumes no liability for any damages, including general, special, incidental, or consequential damages arising from the use of the diagnostic procedures in this manual (including, but not limited to any loss of profit or savings, loss of data, or failure to successfully troubleshoot network problems).

Diagnostic information may be changed or updated without notice. You are responsible for obtaining newer versions of this manual when they are made available. The procedures in this document are not intended as a substitute for the expertise of qualified technicians.

Enabling diagnostic commands can seriously degrade system performance. Diagnostic commands are generally intended for use when troubleshooting specific problems while working with qualified service technicians, or in conjunction with calls to Brocade Technical Support. Whenever possible, troubleshoot your system during periods of low network traffic and user activity to preserve system performance.

If you have any questions regarding this Disclaimer please contact us at support@brocade.com.

Supported hardware and software

The following hardware platforms are supported by this release of this guide:

TABLE 1 Supported devices

Brocade NetIron XMR Series	Brocade MLX Series	NetIron CES 2000 and NetIron CER 2000 Series
Brocade NetIron XMR 4000	Brocade MLX-4	Brocade NetIron CER-RT 2024C
Brocade NetIron XMR 8000	Brocade MLX-8	Brocade NetIron CER-RT 2024F
Brocade NetIron XMR 16000	Brocade MLX-16	Brocade NetIron CER-RT 2024F
Brocade NetIron XMR 32000	Brocade MLX-32	Brocade NetIron CES 2048C
	Brocade MLXe-4	Brocade NetIron CER-RT 2048C
	Brocade MLXe-8	Brocade NetIron CES 2048CX
	Brocade MLXe-16	Brocade NetIron CER-RT 2048CX
	Brocade MLXe-32	Brocade NetIron CES 2048F
		Brocade NetIron CES 2048FX
		Brocade NetIron CER 2024C
		Brocade NetIron CER 2024F
		Brocade NetIron CER 2048C
		Brocade NetIron CER 2048CX
		Brocade NetIron CER 2048F
		Brocade NetIron CER-RT 2048F
		Brocade NetIron CER 2048FX
		Brocade NetIron CER-RT 2048FX

Supported software

For the complete list of supported features and the summary of enhancements and configuration notes for this release, refer to the *Multi-Service IronWare R05.5.00c Release Notes*.

Document conventions

This section describes text formatting conventions and important notice formats used in this document.

Text formatting

The narrative-text formatting conventions that are used are as follows:

bold text	Identifies command names
	Identifies the names of user-manipulated GUI elements
	Identifies keywords
	Identifies text to enter at the GUI or CLI
italic text	Provides emphasis
	Identifies variables
	Identifies document titles
`code` text	Identifies CLI output

Notes and cautions

The following notices and statements are used in this manual. They are listed below in 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 potential damage to hardware or data.

CAUTION

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

Related publications

For the latest edition of these documents, which contain the most up-to-date information, see Documentation at *http://www.brocade.com/ethernetproducts*

- *Multi-Service IronWare Administration Guide*
- *Multi-Service IronWare Security Configuration Guide*

- *Multi-Service IronWare Switching Configuration Guide*
- *Multi-Service IronWare Routing Configuration Guide*
- *Multi-Service IronWare Traffic Management Configuration Guide*
- *Multi-Service IronWare Multicast Configuration Guide*
- *Multi-Service IronWare Multiprotocol Label Switch (MPLS) Configuration Guide*
- *Multi-Service IronWare Software Defined Networking (SDN) Guide*
- *Brocade MLX Series and NetIron Family YANG Guide*
- *Brocade MLX Series and NetIron XMR Series Diagnostic Reference*
- *Unified IP MIB Reference*
- *Multi-Service IronWare Software Upgrade Procedures for Brocade MLX Series and NetIron Family devices*
- *Brocade MLXe Series Installation Guide*
- *Brocade MLX Series and Brocade NetIron XMR Installation Guide*
- *Brocade NetIron CES 2000 Series and Brocade NetIron CER 2000 Series Hardware Installation Guide*

Getting technical help

To contact Technical Support, go to http://www.brocade.com/services-support/index.page for the latest e-mail and telephone contact information.

Document feedback

Quality is our first concern at Brocade and we have made every effort to ensure the accuracy and completeness of this document. However, if you find an error or an omission, or you think that a topic needs further development, we want to hear from you. Forward your feedback to:

documentation@brocade.com

Provide the title and version number of the document and as much detail as possible about your comment, including the topic heading and page number and your suggestions for improvement.

Chapter

Using Diagnostic Commands

1

In this chapter

This chapter describes how to use Brocade diagnostic debug commands to monitor and troubleshoot the Brocade NetIron XMR and Brocade MLX series device configurations. Debug commands are accessible from the Privileged EXEC mode in the Multi-Service IronWare command line interface (CLI). Most debug commands can be configured to send output to a destination that you specify.

When enabled, debug commands can noticeably affect system performance. Many debug commands are specifically designed to be used in conjunction with calls to Brocade Technical Support. If you report a problem, the support engineer may ask you to execute one or more of the debug commands described in this guide.

ATTENTION
Some debug commands report information about internal hardware settings and registers that are relevant primarily to the Brocade engineering staff. These commands are not described in this document.

Show commands

Show commands provide information that is extremely helpful for troubleshooting. For most of the environments discussed in this document, related show commands, show command output, and output descriptions are included.

Many debug commands work in conjunction with show commands to generate output for a specific configuration.

show log

Syntax: **show log**

The **show log** command allows you to view the system log or traps logged on an SNMP trap receiver. Command output similar to the following is displayed. This output indicates that one switchover from standby to active has occurred.

```
Brocade# show log
Syslog logging: enabled (0 messages dropped, 0 flushes, 0 overruns)
Buffer logging: level ACDMEINW, 24 messages logged
level code: A=alert C=critical D=debugging M=emergency E=error
I=informational N=notification W=warning
Static Log Buffer:
Sep 28 11:31:25:A:Power Supply 1, 1st left, not installed
Sep 28 11:31:25:A:Power Supply 3, middle left, not installed
```

```
Sep 28 11:31:25:A:Power Supply 4, middle right, failed
Sep 28 11:31:25:A:Power Supply 5, 2nd right, not installed
Dynamic Log Buffer (50 lines):
Sep 27 18:06:58:I:Interface ethernet6/2, state up
Sep 27 18:06:57:I:Interface ethernet3/2, state up
Sep 27 15:39:42:I:Interface ethernet3/2, state up
Sep 27 15:39:42:I:Interface ethernet6/2, state up
Sep 27 14:23:45:N:Module up in slot 6
Sep 27 14:23:45:N:Module up in slot 3
Sep 27 14:23:27:A:Management module at slot 9 state changed from standby to active
```

Generic debug commands

The following generic debug commands perform functions related to all debugging actions:

- **debug ?** - Generates a list of debug options.
- **[no] debug all** - Enables or disables all debug functions.
- **show debug** - Shows all enabled debug settings.
- **debug destination** - Allows you to select an output destination: Telnet, SSH, console, or logging (default).

debug ?

Syntax: debug ?

The **debug ?** command generates a list of available debug variables.

ATTENTION
Many first-level variables have their own variable subsets. When you enter a debug command, the system will indicate that there are additional variables by telling you that you have entered an incomplete command. Add a space and a question mark to your original command to view the additional variables.

```
Brocade# debug ip
Incomplete command.
Brocade# debug ip?
access-list          Enable ACL debugging
all                  Enable all debugging
bfd                  Enable BFD debugging
destination          Redirect debug message
dot1x                Debug 802.1X and Events
filters              Enable Filters debugging
gvrp                 Enable gvrp debugging
ip                   Debug trace IP
ipv6                 Debug trace IPv6
isis                 Debug isis
mac                  Enable MAC database debugging
.
.
```

show debug

Syntax: show debug

The **show debug** command displays all enabled debug functions. Command output resembles the following example, which shows that RSTP and IS-IS debugging are enabled, with the console as the output destination.

```
Brocade# show debug
RSTP
        RSTP:  debugging is on
Debug message destination:  Console
INTEGRATED IS-IS :
          IS-IS:  isis debugging is on INTEGRATED IS-IS :
          IS-IS:  isis debugging is on
```

debug all

Syntax: [no] debug all

This command enables all debug functions, and must *only* be used during troubleshooting. To cancel this setting, enter the **no debug all** command.

```
Brocade# debug all
Warning! This may severely impact network performance!
All possible debuggings have been turned on
```

CAUTION

This command generates extensive output and can significantly slow device operation. Use this command with caution. Never use this command during periods of peak network activity. Enter no debug all to stop the output.

NOTE
You may not be able to see the **no debug all** command as you type it. However, if you have typed the command correctly, output will stop as soon as you press the **Enter** key.

---

debug destination

Syntax: [no] debug destination [console | logging | telnet *num* **| ssh** *num***]**

This command allows you to specify a destination for debugging output. The default is the system console, but you can redirect output to a syslog buffer, or a Telnet or SSH session. The following parameters are available for this command:

- **console** - Directs output to the system console.
- **logging** - Directs output to the syslog buffer and to the syslog server (default).
- **telnet** *num* - Directs debugging output to a specified Telnet session (a number from 1 through 5).
- **ssh** *num* - Directs debugging output to a specified SSH session (a number from 1 through 5).

To send debug output to a Telnet session, first determine your session number using the **show who** command.

```
Brocade# show who
```

You must see output similar to the following example. For purpose of this example, the relevant Telnet session has been highlighted.

```
Console connections:
        established, monitor enabled
        1 minutes 57 seconds in idle
Telnet connections (inbound):
 1      closedn
 2      established, client ip address 10.55.1.128, user is <your login>
        you are connecting to this session
        15 seconds in idle
 3      closed
 4      closed
 5      closed
Telnet connection (outbound):
 6      closed
SSH connections:
 1      closed
 2      closed
 3      closed
 4      closed
```

This example indicates that you are connected through Telnet session 2. Redirect the debug output to your Telnet session by entering the following command.

```
Brocade# debug destination telnet 2
```

Brief and detail debug options

When enabled, many debug commands can significantly impact system performance. Many debug commands provide options for brief or detailed reporting. Generating detailed output places an additional burden on system performance, and in many cases the results may be more difficult to interpret than output generated using the **brief** option. To conserve performance and prevent system disruption, use the **brief** option whenever possible.

Disabling debug commands

When activated, most debug commands instruct the system to collect specific information about router configurations and activity. In all cases, adding **no** in front of the command disables the debug function.

Chapter

System and System Management Diagnostics 2

In this chapter

This chapter describes many of the common system and system management diagnostic processes for Brocade NetIron XMR series and Brocade MLX series devices.

Basic system information

Basic system troubleshooting includes the verification of software images and their locations, and monitoring hardware components such as fans and power supplies. The following sections describe how to display information, and what to look for when troubleshooting your hardware and system software.

System hardware show commands

This section describes the show command that displays system hardware information.

show chassis

Syntax: show chassis

The **show chassis** command displays information about the Brocade NetIron XMR and Brocade MLX series chassis, including power supplies, fan status and operating speeds, and temperature readings for all installed modules (temperatures are, by default, polled every 60 seconds). The following example shows output for the **show chassis** command.

```
Brocade# show chassis
*** NetIron XMR 8000 CHASSIS ***

---POWERS ---
```

```
Power 1: Installed (Failed or Disconnected)
Power 2: Installed (Failed or Disconnected)
Power 3 (30351200 - AC 1200W): Installed (OK)
Power 4 (30351200 - AC 1200W): Installed (OK)
Total power budget for chassis = 2400W
Total power budget for LPs      = 2049W
Slot Power-On Priority and Power Usage:
Slot2 pri=1 module type=NI-X-OC48x4 4-port OC48/12 STM16/STM4 Module power usage
=132W
Slot3 pri=1 module type=NI-XMR-1Gx20-GC 20-port 10/100/1000 Copper Module power
usage=156W
Slot4 pri=1 module type=NI-XMR-10Gx2 2-port 10GbE Module power usage=165W

--- FANS ---
Right fan tray (fan 1): Status = OK, Speed = MED (75%)
Right fan tray (fan 2): Status = OK, Speed = MED (75%)
Right fan tray (fan 3): Status = OK, Speed = MED (75%)
Right fan tray (fan 4): Status = OK, Speed = MED (75%)

--- TEMPERATURE READINGS ---
Active Mgmt Module: 38.0C 52.375C
Standby Mgmt Module: 35.250C
SNM1: 30.0C
SNM2: 27.5C
SNM3: 30.0C
LP2 Sensor1: 38.0C
LP2 Sensor2: 53.0C
LP3 Sensor1: 33.5C
LP3 Sensor2: 40.750C
LP4 Sensor1: 38.5C
LP4 Sensor2: 46.500C
LP4 Sensor3: UNUSED
Temperature Monitoring Poll Period is 60 seconds
```

For more information about how to troubleshoot hardware issues, refer to “Power supplies, fans, and temperature” on page 48.

System software show commands

This section describes the show command that displays system software information.

show version

Syntax: show version

Most boot issues occur because incorrect or incompatible images have been downloaded. The **show version** command displays all versions that are currently loaded, as shown in the following example.

```
Brocade# show version
HW: NetIron XMR Router
Backplane (Serial #: Not Exist,  Part #: Not Exist)
NI-X-SF Switch Fabric Module 1 (Serial #: PR29050242,  Part #: 31523-100A)
FE 1: Type 00000,  Version 0
FE 3: Type 00000,  Version 0
NI-X-SF Switch Fabric Module 2 (Serial #: PR29050246,  Part #: 31523-100A)
FE 1: Type 00000,  Version 0
FE 3: Type 00000,  Version 0
NI-X-SF Switch Fabric Module 3 (Serial #: PR30050270,  Part #: 31523-100A)
```

```
FE 1: Type 00000,  Version 0
FE 3: Type 00000,  Version 0
==========================================================================
SL M1: NI-XMR-MR Management Module Active (Serial #: SA12061726, Part #:
31524-100A):
Boot     : Version 3.5.0T165 Copyright (c) 1996-2009 Brocade Communications, Inc.
Compiled on Jul 10 2009 at 19:13:56 labeled as xmprm03500
 (424484 bytes) from boot flash
Monitor  : Version 3.5.0aT165 Copyright (c) 1996-2009 Brocade Communications, Inc.
Compiled on Jul 30 2009 at 17:35:22 labeled as xmb03500a
 (424748 bytes) from code flash
IronWare : Version 3.5.0cT163 Copyright (c) 1996-2009 Brocade Communications, Inc.
Compiled on Sep 17 2009 at 01:00:12 labeled as xmr03500c
 (5840562 bytes) from Primary
Board ID : 00 MBRIDGE Revision : 18
916 MHz Power PC processor  (version 8003/0101) 166 MHz bus
512 KB Boot Flash (AM29LV040B), 32 MB Code Flash (MT28F128J3)
1024 MB DRAM
Active Management uptime is 21 days 10 hours 44 minutes 44 seconds
==========================================================================
SL M2: NI-XMR-MR Management Module Standby (Serial #: SA11060307, Part #:
31524-100A):
Boot     : Version 3.5.0T165 Copyright (c) 1996-2009 Brocade Communications, Inc.
Compiled on Jul 10 2009 at 19:13:56 labeled as xmprm03500
 (424484 bytes) from boot flash
Monitor  : Version 3.5.0aT165 Copyright (c) 1996-2009 Brocade Communications, Inc.
Compiled on Jul 30 2009 at 17:35:22 labeled as xmb03500a
 (424748 bytes) from code flash
IronWare : Version 3.5.0cT163 Copyright (c) 1996-2009 Brocade Communications, Inc.
Compiled on Sep 17 2009 at 01:00:12 labeled as xmr03500c
 (5840562 bytes) from Primary
Board ID : 00 MBRIDGE Revision : 18
916 MHz Power PC processor  (version 8003/0101) 166 MHz bus
512 KB Boot Flash (AM29LV040B), 32 MB Code Flash (MT28F128J3)
1024 MB DRAM
Standby Management uptime is 19 days 14 hours 4 minutes 45 seconds
SL 3: NI-XMR-1Gx20-SFP 20-port 1GbE/100FX Module (Serial #: SA23060375, Part #:
31570-103A)
Boot     : Version 3.3.0gT175 Copyright (c) 1996-2009 Brocade Communications, Inc.
Compiled on Aug 29 2009 at 12:12:02 labeled as xmlprm03300g
 (336122 bytes) from boot flash
Monitor  : Version 3.3.0gT175 Copyright (c) 1996-2009 Brocade Communications, Inc.
Compiled on Aug 29 2009 at 12:12:46 labeled as xmlb03300g
 (659473 bytes) from code flash
IronWare : Version 3.3.0gT177 Copyright (c) 1996-2009 Brocade Communications, Inc.
Compiled on Aug 29 2009 at 18:37:36 labeled as xmlp03300g
 (2410342 bytes) from Primary
FPGA versions:
Valid PBIF Version = 2.18, Build Time = 7/21/2009 12:21:0

Valid XPP Version = 2.25, Build Time = 8/2/2009 10:33:0

BCM5695GMAC 0
BCM5695GMAC 1
BCM5695GMAC 2
BCM5695GMAC 3
666 MHz MPC 8541 (version 8020/0020) 333 MHz bus
512 KB Boot Flash (AM29LV040B), 16 MB Code Flash (MT28F640J3)
1024 MB DRAM, 8 KB SRAM, 0 Bytes BRAM
PPCR0: 1024K entries CAM, 16384K PRAM, 2048K AGE RAM
```

```
LP Slot 3 uptime is 3 hours 11 minutes 50 seconds
SL 5: NI-XMR-10Gx4 4-port 10GbE Module (Serial #: pr32050022, Part #: 31546-100A)
Boot     : Version 3.3.0gT175 Copyright (c) 1996-2009 Brocade Communications, Inc.
Compiled on Aug 29 2009 at 12:12:02 labeled as xmlprm03300g
 (336122 bytes) from boot flash
Monitor  : Version 3.3.0gT175 Copyright (c) 1996-2009 Brocade Communications, Inc.
Compiled on Aug 29 2009 at 12:12:46 labeled as xmlb03300g
 (659473 bytes) from code flash
IronWare : Version 3.3.0gT177 Copyright (c) 1996-2009 Brocade Communications, Inc.
Compiled on Aug 29 2009 at 18:37:36 labeled as xmlp03300g
 (2410342 bytes) from Primary
FPGA versions:
Valid PBIF Version = 2.18, Build Time = 7/21/2009 12:21:0

Valid XPP Version = 2.25, Build Time = 8/2/2009 10:33:0

Valid XGMAC Version = 0.11, Build Time = 10/11/2009 12:45:0

BCM5673X10GMAC 0
BCM5673X10GMAC 1
BCM5673X10GMAC 2
BCM5673X10GMAC 3
666 MHz MPC 8541 (version 8020/0020) 333 MHz bus
512 KB Boot Flash (AM29LV040B), 16 MB Code Flash (MT28F640J3)
1024 MB DRAM, 8 KB SRAM, 0 Bytes BRAM
PPCR0: 1024K entries CAM, 16384K PRAM, 2048K AGE RAM
PPCR1: 1024K entries CAM, 16384K PRAM, 2048K AGE RAM
LP Slot 5 uptime is 3 hours 11 minutes 50 seconds
```

show flash

Syntax: show flash

The **show flash** command displays the images that have been copied onto flash memory.

```
Brocade# show flash
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Active Management Module (Right Slot)
Code Flash - Type MT28F128J3, Size 32 MB
  o IronWare Image (Primary)
    Version 3.5.0T163, Size 5819609 bytes, Check Sum ec17
    Compiled on Jun 18 2009 at 07:15:12 labeled as xmr03500b201
  o LP Kernel Image (Monitor for LP Image Type 0)
    Version 3.5.0T175, Size 386693 bytes, Check Sum 5ff6
    Compiled on May 31 2009 at 14:42:56 labeled as xmlb03500b155
  o LP IronWare Image (Primary for LP Image Type 0)
    Version 3.5.0T177, Size 3128223 bytes, Check Sum f07b
    Compiled on Jun 18 2009 at 07:49:48 labeled as xmlp03500b201
  o Monitor Image
    Version 3.5.0T165, Size 424045 bytes, Check Sum 66f0
    Compiled on May 31 2009 at 14:41:14 labeled as xmb03500b155
  o Startup Configuration
    Size 12466 bytes, Check Sum 1bb2
    Modified on 14:01:37 Pacific Mon Jun 18 2009
Boot Flash - Type AM29LV040B, Size 512 KB
  o Boot Image
    Version 3.5.0T165, Size 424038 bytes, Check Sum f1e9
    Compiled on May 31 2009 at 14:42:00 labeled as xmprm03500b155
Standby Management Module (Left Slot)
Code Flash: Type MT28F128J3, Size 32 MB
```

```
  o IronWare Image (Primary)
    Version 3.5.0T163, Size 5819609 bytes, Check Sum ec17
    Compiled on Jun 18 2009 at 07:15:12 labeled as xmr03500b201
  o LP Kernel Image (Monitor for LP Image Type 0)
    Version 3.5.0T175, Size 386693 bytes, Check Sum 5ff6
    Compiled on May 31 2009 at 14:42:56 labeled as xmlb03500b155
  o LP IronWare Image (Primary for LP Image Type 0)
    Version 3.5.0T177, Size 3128223 bytes, Check Sum f07b
    Compiled on Jun 18 2009 at 07:49:48 labeled as xmlp03500b201
  o Monitor Image
    Version 3.5.0T165, Size 424045 bytes, Check Sum 66f0
    Compiled on May 31 2009 at 14:41:14 labeled as xmb03500b155
  o Startup Configuration
    Size 12466 bytes, Check Sum 1bb2
    Modified on 14:01:38 Pacific Mon Jun 18 2009
Boot Flash: Type AM29LV040B, Size 512 KB
  o Boot Image
    Version 3.5.0T165, Size 424038 bytes, Check Sum f1e9

.
.
```

show who

Syntax: **show who**

The **show who** command displays information about users who are logged in to a Telnet connection, including privilege levels, as shown in the following example.

```
Brocade# show who
Console connections:
       established
       3 days 17 hours 31 minutes 27 seconds in idle
Telnet server status: Enabled
Telnet connections (inbound):
1      established, client ip address 10.53.1.65, privilege super-user
       you are connecting to this session
2      closed
3      closed
4      closed
5      closed
Telnet connections (outbound):
6      established, server ip address 10.47.2.200, from Telnet session 1
       4 seconds in idle
7      closed
8      closed
9      closed
10     closed
SSH server status: Enabled
SSH connections:
1      closed
```

show save

Syntax: **show save** [**active-mp** | **lp** | **standby-mp**]

- **active-mp** - Displays active Management Processor (MP) crash dump information.
- **lp** - Displays Line Processor (LP) crash dump information.
- **standby-mp** - Displays standby MP crash dump information.

The **show save** command displays saved crash information. Command output resembles the following example.

```
Brocade# show save
Boot    : 05.03.00T165 xmprm05300 built on Oct 24 2011 16:08:02 PDT
Monitor : 05.03.00T165 xmb05300b1 built on Nov 22 2011 22:58:34 PST
System  : 05.03.00T163 xmr05300b1 built on Nov 22 2011 05:01:48 PST

Task    : console

Created : 08:36:21 11-23-2011

System had been up for 5 minutes

EXCEPTION 0000, Soft Check - Timeout (30s)

Task    : console

GP Registers
r0      : 200535d4 26ed90f0 21cd62c0 00000000
r4      : 21d0e1fc 00331f40 26ed9198 6c2022f4
r8      : 6c000000 00000001 0000000c 00000020
r12     : 00000000 22063740 00000000 00000000
r16     : 00000015 00150001 00000000 26edb72c
r20     : 00000069 00000000 1000c100 00000000
```

System debug commands

This section describes the system-related debug commands.

debug system trace

Syntax: [**no**] **debug system trace**

This command performs a system debugging trace. Command output resembles the following example.

```
Brocade# debug system trace
SYSLOG: <13>Dec 10 20:48:43 Edge4 System: Module 5 powered off
Slot 5 is powered off.
WARN IPC: Slot for dest FID 53252 is not UP
WARN IPC: Slot for dest FID 53252 is not UP
WARN IPC: Slot for dest FID 53252 is not UP
WARN IPC: Slot for dest FID 53252 is not UP
WARN IPC: Slot for dest FID 53252 is not UP
WARN IPC: Slot for dest FID 53252 is not UP
WARN IPC: Slot for dest FID 53252 is not UP
WARN IPC: Slot for dest FID 53252 is not UP
WARN IPC: Slot for dest FID 53252 is not UP
WARN IPC: Slot for dest FID 53252 is not UP
WARN IPC: Slot for dest FID 53252 is not UP
WARN IPC: Slot for dest FID 53252 is not UP
WARN IPC: Slot for dest FID 53252 is not UP
WARN IPC: Slot for dest FID 53252 is not UP
WARN IPC: Slot for dest FID 53252 is not UP
WARN IPC: Slot for dest FID 53252 is not UP
WARN IPC: Slot for dest FID 53252 is not UP
WARN IPC: Slot for dest FID 53252 is not UP
WARN IPC: Slot for dest FID 53252 is not UP
```

```
WARN IPC: Slot for dest FID 53252 is not UP
WARN IPC: Slot for dest FID 53252 is not UP
WARN IPC: Slot for dest FID 53252 is not UP
WARN IPC: Slot for dest FID 53252 is not UP
WARN IPC: Slot for dest FID 53252 is not UP
```

debug system upgrade

Syntax: [no] debug system upgrade

This command enables and displays the debugging traces. Command output resembles the following example.

```
Brocade# debug system upgrade
*** Package upgrade CB parsed data ***
Manifest path :
Source        : 4
MP MON     : /Monitor/ManagementModule/xmb05300.bin
MP APP     : /Application/ManagementModule/xmr05300b270.bin
MP BOOT    : /Boot/ManagementModule/xmprm05200.bin
MBRIDGE    : /FPGA/ManagementModule/mbridge_05300b270.xsvf
MBRIDGE32  : /FPGA/ManagementModule/mbridge32_05300b270.xsvf
SBRIDGE    : /FPGA/ManagementModule/sbridge_05300b270.mcs
HSBRIDGE   : /FPGA/ManagementModule/hsbridge_05300b270.mcs
LP MON     : /Monitor/InterfaceModule/xmlb05300.bin
LP APP     : /Application/InterfaceModule/xmlp05300b270.bin
LP BOOT    : /Boot/InterfaceModule/xmlprm05200.bin
LP FPGA All: /Combined/FPGA/lpfpga05300b270.bin
PBIF SP2   : /FPGA/InterfaceModule/pbifsp2_05300b270.bin
PBIF MRJ   : /FPGA/InterfaceModule/pbifmrj_05300b270.bin
PBIF OC    : /FPGA/InterfaceModule/pbifoc_05300b270.bin
PBIF 8x10  : /FPGA/InterfaceModule/pbif8x10_05300b270.bin
XPP SP2    : /FPGA/InterfaceModule/xppsp2_05300b270.bin
XPP MRJ    : /FPGA/InterfaceModule/xppmrj_05300b270.bin
XPP OC     : /FPGA/InterfaceModule/xppoc_05300b270.bin
XPP 8x10   : /FPGA/InterfaceModule/xpp8x10_05300b270.bin
XPP 2x100  : /FPGA/InterfaceModule/xpp2x100_05300b270.bin
STATS MRJ  : /FPGA/InterfaceModule/statsmrj_05300b270.bin
STATS OC   : /FPGA/InterfaceModule/statsoc_05300b270.bin
XGMAC SP2  : /FPGA/InterfaceModule/xgmacsp2_05300b270.bin
CE         :
CEB        :
PBIF Metro :
num_downloads : 0
*** Package upgrade CB data ***
Manifest path  :
Source         : 4
num_downloads  : 7
num_download_recs : 7
error count       : 0
```

debug trace-l2 events

Syntax: [no] debug trace-l2 events

This command displays information about Layer 2 trace protocol events. Command output resembles the following example.

```
Brocade# debug trace-l2 events
Brocade# trace-l2 vlan 3
```

```
Dec 10 17:21:38 L2 Trace: trace_route_l2(): Entering
Dec 10 17:21:39 L2 Trace: trace_l2_timer(): Entering
Dec 10 17:21:39 L2 Trace: trace_l2_timer(): Stage 1
Dec 10 17:21:39 L2 Trace: trace_l2_append_payload(): manipulate_input_port = 0,
in_port = 65535
Dec 10 17:21:39 L2 Trace: trace_l2_append_payload(): Unmodified hop->input_port =
4095
Dec 10 17:21:39 L2 Trace: trace_l2_append_payload(): Exit 2
Dec 10 17:21:39 L2 Trace: trace_l2_timer(): Exit 2
Dec 10 17:21:39 L2 Trace: trace_l2_receive_reply(): Entering
Dec 10 17:21:39 L2 Trace: trace_l2_find_elapse_micro_seconds(): time1 = 425976272
Dec 10 17:21:39 L2 Trace: trace_l2_receive_reply(): Exit 4: Processed Hop
Dec 10 17:21:39 L2 Trace: trace_l2_receive_reply(): Entering
Dec 10 17:21:39 L2 Trace: trace_l2_find_elapse_micro_seconds(): time1 = 425976272
Dec 10 17:21:39 L2 Trace: trace_l2_receive_reply(): Exit 4: Processed Hop
Dec 10 17:21:39 L2 Trace: trace_l2_receive_reply(): Entering
Dec 10 17:21:39 L2 Trace: trace_l2_find_elapse_micro_seconds(): time1 = 425976272
Dec 10 17:21:39 L2 Trace: trace_l2_receive_reply(): Exit 4: Processed Hop
Dec 10 17:21:39 L2 Trace: trace_l2_receive_reply(): Entering
Dec 10 17:21:39 L2 Trace: trace_l2_find_elapse_micro_seconds(): time1 = 425976272
Dec 10 17:21:39 L2 Trace: trace_l2_receive_reply(): Exit 4: Processed Hop
Dec 10 17:21:39 L2 Trace: trace_l2_receive_reply(): Entering
Dec 10 17:21:39 L2 Trace: trace_l2_find_elapse_micro_seconds(): time1 = 425976272
Dec 10 17:21:39 L2 Trace: trace_l2_receive_reply(): Exit 4: Processed Hop
Dec 10 17:21:39 L2 Trace: trace_l2_receive_reply(): Entering
Dec 10 17:21:39 L2 Trace: trace_l2_find_elapse_micro_seconds(): time1 = 425976272
Dec 10 17:21:39 L2 Trace: trace_l2_receive_reply(): Exit 4: Processed Hop
Dec 10 17:21:39 L2 Trace: trace_l2_timer(): Entering
Dec 10 17:21:39 L2 Trace: trace_l2_timer(): Exit 3 (Stage 2)
Dec 10 17:21:40 L2 Trace: trace_l2_timer(): Entering
Dec 10 17:21:40 L2 Trace: trace_l2_timer(): Have no address
Vlan 3 L2 topology probed, use "trace-l2 show" to display
Dec 10 17:21:40 L2 Trace: trace_l2_timer(): Exit 4: End of function
```

Common diagnostic scenarios

System issues are rare. However, some problem sources can include:

- Software versions are not compatible.
- Line modules or switch fabric modules are not functioning properly.
- Environmental conditions, such as temperatures that are above or below operating thresholds, are affecting operation of hardware components.

If you are experiencing system issues, contact Brocade Technical Support for help in troubleshooting your system.

TCAM partitioning and usage

Ternary Content Addressable Memory (TCAM) is a component of Brocade devices that facilitates hardware-forwarding. As packets flow through the Brocade device from a given source to a given destination, the management processor records forwarding information about the flow in TCAM entries. A TCAM entry generally contains next-hop information, such as the outgoing port, the MAC address of the next-hop router, a VLAN tag, and so on. Once the Brocade device has this information in TCAM, packets with the same source and destination can be forwarded by hardware, bypassing the management processor, and speeding up forwarding time.

TCAM entries can contain Layer 2, Layer 3, or Layer 4 information. Each type of TCAM entry has its own format:

- Layer 2 TCAM entries contain destination MAC information and deal with 802.1p (priority) and VLAN information.
- Layer 3 TCAM entries contain destination IP information.
- Layer 4 TCAM entries contain destination IP, destination TCP/UDP port, source IP, and source TCP/UDP port information.

When a Brocade device is initialized, the software partitions the available TCAM into segments for Layer 2, Layer 3, or Layer 4 information. The percentage of TCAM devoted to each type of TCAM entry is determined by the profile.

Table 2 shows the TCAM sizes and functions for Brocade NetIron XMR and Brocade MLX series devices.

TABLE 2 TCAM sizes and support

TCAM	Brocade NetIron XMR	Brocade MLX series	Supports
TCAM 0/TCAM 1	Cascaded 2x18 Mb	18 Mb	IPv4, IPv6, MAC DA, MAC SA, Layer 3 VPN routes (uplink and endpoints), IPv4 for RPF
TCAM 2	18 Mb	9 Mb	Inbound IPv4 ACL, inbound IPv6 ACL, inbound Layer 2 ACL, MAC SA, MAC DA, Layer 3 VPN routes (uplinks and endpoints), VPLS DA, VPLS SA (uplinks and endpoints), multicast
TCAM 3	9 Mb	9 Mb	Outbound IPv4 ACL, outbound IPv6 ACL, outbound Layer 2 ACL only

TCAM show commands

This section describes the show commands that display TCAM information.

show cam ifl

Syntax: **show cam ifl** *slotnum/portnum*

This command displays Content Addressable Memory (CAM) Internal Forwarding Lookup (IFL) information for a specified slot and port.

```
Brocade# show cam ifl 7/7
Slot Index  Port  Outer VLAN Inner VLAN PRAM  IFL ID IPV4/V6| (Hex)
(Hex)          Routing
7    0081fe9 7/4   4000       0          181fe9 131071 1/1
7    0081fea 7/3   4000       0          181fea 131071 1/1
7    0081feb 7/2   4000       0          181feb 131071 1/1
7    0081fec 7/1   4000       0          181fec 131071 1/1
7    0081fed 7/8   607        0          181fed 131071 1/1
7    0081fee 7/7   607        0          181fee 131071 1/1
7    0081fef 7/8   606        0          181fef 131071 1/1
7    0081ff0 7/7   606        0          181ff0 131071 1/1
7    0081ff1 7/8   605        0          181ff1 131071 1/1
7    0081ff2 7/7   605        0          181ff2 131071 1/1
7    0081ff3 7/8   604        0          181ff3 131071 1/1
7    0081ff4 7/7   604        0          181ff4 131071 1/1
7    0081ff5 7/8   603        0          181ff5 131071 1/1
7    0081ff6 7/7   603        0          181ff6 131071 1/1
```

```
7    0081ff7 7/8   602        0          181ff7 131070 1/1
7    0081ff8 7/7   602        0          181ff8 131070 1/1
7    0081ff9 7/8   601        0          181ff9 131071 1/1
7    0081ffa 7/7   601        0          181ffa 131071 1/1
Slot Index   Port  Outer VLAN Inner VLAN PRAM   IFL ID IPV4/V6
(Hex)                               (Hex)         Routing
7    0081ffb 7/8   6          0          181ffb 131071 1/1
7    0081ffc 7/7   6          0          181ffc 131071 1/1
7    0081ffd 7/19  4000       0          181ffd 131071 1/1
7    0081ffe 7/14  100        0          181ffe 131071 1/1
7    0081fff 7/18  4000       0          181fff 131071 1/1
```

show cam l4

Syntax: show cam l4 *slot/port*

This command displays TCAM partition information on a specific layer 4 interface, as shown in the following example.

```
Brocade# show cam l4 1/2
LP Index Src  IP         SPort Pro Age IFL/ Out IF Group PRAM
   (Hex)(Dest IP         DPort)        VLAN Action       (Hex)
1  a4000 0.0.0.0         0     17  Dis 0    Protoc 31    00084
        (127.0.0.0       3784 )
1  a4800 10.1.1.2        0     0   Dis 0    Pass   16    00097
        (10.1.1.255      0    )
1  a4802 10.1.1.2        0     0   Dis 0    Pass   16    00098
        (10.1.1.255      0    )
1  a4804 0.0.0.0         0     0   Dis 0    Drop   16    00099
        (10.1.1.255      0    )
1  a4806 0.0.0.0         0     0   Dis 0    Drop   16    0009a
        (10.1.1.255      0    )
```

show cam-partition

Syntax: show cam-partition [**brief** | **slot** *slotnum* | **usage**]

The following parameters are available for this command:

- **brief** - Displays a brief summary of partition information.
- **slot** *slotnum* - Displays partition information for a specific slot.
- **usage** - Displays brief partition usage information.

The following examples show output for this command using these parameters.

The **show cam-partition brief** command displays TCAM information per partition and sub-partition in three formats: raw size, user size, and reserved size, as shown in the following example.

```
Brocade# show cam-partition brief
CAM partitioning profile: default
Slot 1 XPP20SP 0:
# of CAM device               = 4
Total CAM Size                = 917504 entries (63Mbits)
IP: Raw Size 524288, User Size 524288(0 reserved)
  Subpartition  0: Raw Size 12288, User Size 12288, (0 reserved)
  Subpartition  1: Raw Size 468107, User Size 468107, (0 reserved)
  Subpartition  2: Raw Size 37335, User Size 37335, (0 reserved)
  Subpartition  3: Raw Size 5140, User Size 5140, (0 reserved)
  Subpartition  4: Raw Size 778, User Size 778, (0 reserved)
IPv6: Raw Size 131072, User Size 65536(0 reserved)
```

```
  Subpartition  0: Raw Size 12288, User Size 6144, (0 reserved)
  Subpartition  1: Raw Size 107496, User Size 53748, (0 reserved)
  Subpartition  2: Raw Size 9332, User Size 4666, (0 reserved)
  Subpartition  3: Raw Size 1284, User Size 642, (0 reserved)
  Subpartition  4: Raw Size 384, User Size 192, (0 reserved)
IP VPN Raw Size 131072, User Size 131072(0 reserved)
  Subpartition  0: Raw Size 2048, User Size 2048, (0 reserved)
  Subpartition  1: Raw Size 116886, User Size 116886, (0 reserved)
  Subpartition  2: Raw Size 9333, User Size 9333, (0 reserved)
  Subpartition  3: Raw Size 1285, User Size 1285, (0 reserved)
  Subpartition  4: Raw Size 384, User Size 384, (0 reserved)
MAC: Raw Size 131072, User Size 131072(0 reserved)
  Subpartition  0: Raw Size 10, User Size 10, (0 reserved)
  Subpartition  1: Raw Size 32, User Size 32, (0 reserved)
  Subpartition  2: Raw Size 131030, User Size 131030, (0 reserved)
Session: Raw Size 98304, User Size 49152(0 reserved)
  Subpartition  0: Raw Size 79872, User Size 39936, (0 reserved)
  Subpartition  1: Raw Size 2048, User Size 1024, (0 reserved)
  Subpartition  2: Raw Size 16384, User Size 8192, (0 reserved)
IPv6 Session: Raw Size 32768, User Size 4096(0 reserved)
  Subpartition  0: Raw Size 15872, User Size 1984, (0 reserved)
  Subpartition  1: Raw Size 512, User Size 64, (0 reserved)
  Subpartition  2: Raw Size 16384, User Size 2048, (0 reserved)
Out Session: Raw Size 196608, User Size 98304(49152 reserved)
Out IPv6 Session: Raw Size 65536, User Size 8192(4096 reserved)

Slot 1 XPP20SP 0:
Slot 3 XPP20SP 0:
# of CAM device                = 4
Total CAM Size                 = 917504 entries (63Mbits)
```

The **show cam-partition usage** command displays the amount of TCAM being used and how much is available, as shown in the following example.

NOTE

The display has been shortened for brevity.

```
Brocade# show cam-partition usage
CAM partitioning profile: default
Slot 1 XPP20SP 0:
Slot 1 XPP20SP 0:
            [IP]524288(size), 518257(free), 01.15%(used)
        :SNet 0: 12288(size),  12269(free), 00.15%(used)
        :SNet 1:468107(size), 462099(free), 01.28%(used)
        :SNet 2: 37335(size),  37332(free), 00.00%(used)
        :SNet 3:  5140(size),   5140(free), 00.00%(used)
        :SNet 4:   778(size),    778(free), 00.00%(used)
        .
          [IPV6] 65536(size),  65534(free), 00.00%(used)
        :SNet 0:  6144(size),   6144(free), 00.00%(used)
        :SNet 1: 53748(size),  53748(free), 00.00%(used)
        :SNet 2:  4666(size),   4666(free), 00.00%(used)
        :SNet 3:   642(size),    642(free), 00.00%(used)
        :SNet 4:   192(size),    192(free), 00.00%(used)
        .
        [IP VPN]131072(size), 131072(free), 00.00%(used)
        :SNet 0:  2048(size),   2048(free), 00.00%(used)
        :SNet 1:116886(size), 116886(free), 00.00%(used)
        :SNet 2:  9333(size),   9333(free), 00.00%(used)
        :SNet 3:  1285(size),   1285(free), 00.00%(used)
```

```
        :SNet 4:   384(size),    384(free), 00.00%(used)
        .
           [MAC]131072(size), 131067(free), 00.00%(used)
    :Forwarding:131030(size), 131025(free), 00.00%(used)
      :Protocol:    32(size),     32(free), 00.00%(used)
      :Flooding:    10(size),     10(free), 00.00%(used)
       [Session] 49152(size),  49152(free), 00.00%(used)
  :IP Multicast:  8192(size),   8192(free), 00.00%(used)
   :Receive ACL:  1024(size),   1024(free), 00.00%(used)
      :Rule ACL: 39936(size),  39936(free), 00.00%(used)
  [IPV6 Session]  4096(size),   4096(free), 00.00%(used)
  :IP Multicast:  2048(size),   2048(free), 00.00%(used)
   :Receive ACL:    64(size),     64(free), 00.00%(used)
      :Rule ACL:  1984(size),   1984(free), 00.00%(used)
   [Out Session] 49152(size),  49152(free), 00.00%(used)
[Out V6 Session]  4096(size),   4096(free), 00.00%(used)
```

The **show cam-partition slot** *slotnum* command displays the TCAM information for a specific slot, as shown in the following example.

NOTE
The display has been shortened for brevity.

```
Brocade# show cam-partition slot 1
Session Section : 655360 (0a0000) - 753663 (0b7fff)
  IP Source Guard Denial:      0 (000000) -     -1 (ffffff)
  IP Source Guard Permit:      0 (000000) -     -1 (ffffff)
  Rule-based ACL        : 675840 (0a5000) - 753663 (0b7fff)
  Broadcast ACL         : 673792 (0a4800) - 675839 (0a4fff)
  Receive ACL           : 671744 (0a4000) - 673791 (0a47ff)
  IP Multicast          : 655360 (0a0000) - 671743 (0a3fff)
  IP Multicast 1G       : 655360 (0a0000) - 655359 (09ffff)
  IP Multicast 2GM      : 655360 (0a0000) - 655359 (09ffff)
```

Configuration notes

Keep the following information in mind when you are resetting TCAM partitioning:

- Partition TCAMs to best fit the applications that are running on your device.
- If you do not select a non-default profile, the default profile will be in effect.
- The system must be rebooted for TCAM changes to take effect. Always remember to write to memory before you reboot your system.
- Choose a TCAM profile based on all of the application requirements, not on the maximum available TCAM entries for any specific application. The maximum number of entries will vary for different applications.

Maximum TCAM address dependencies

The Brocade NetIron XMR and Brocade MLX series router can have up to 16,000 static and dynamic MAC address entries stored in the TCAM. The ability of the TCAM to store large numbers of addresses depends on the following factors:

- The number of source MAC addresses being learned by the TCAM.
- The number of destination MAC addresses being forwarded by the TCAM.

- The distribution of the MAC address entries across ports. For example, if one port is learning all the source MAC addresses, the available TCAM for that port will be used up. In addition, a large number of MAC address entries in the MAC table could increase CPU use.

Supernet TCAM partition sharing

In Multi-Service IronWare software versions prior to 03.2.00, TCAM resources could not be shared between the 32 levels of the IP Forwarding Information Base (FIB). Beginning with 03.2.00, TCAM allocation is optimized for dynamic allocation of resources to each level. If one level runs out of TCAM resources, it can obtain resources that have been allocated to another level but are unused. This feature applies to IPv4 and Layer 3 VPN routes.

Configuring adequate TCAM resources for VPLS CPU protection

There must be adequate TCAM resources available to use Virtual Private LAN Service (VPLS) CPU protection. Each endpoint and each uplink port requires a single TCAM entry. In addition, if an endpoint is a trunk port, one entry is required for each port in the trunk. To determine the number of entries required for your system, add the number of VPLS endpoints, ports within a trunk port used as an endpoint, and uplink ports. Use this number with the **system-max hw-flooding** command to configure adequate TCAM resources.

Common diagnostic scenarios

When troubleshooting TCAM issues, it is helpful to understand how to determine the most appropriate TCAM settings for your system and to know when a device is running out of TCAM. The following sections describe how to work with TCAM settings.

Determining appropriate TCAM settings

When a Brocade device boots, the system automatically sets default TCAM partitions. You can customize TCAM settings to best fit the specific tasks your devices are performing. The default TCAM settings are the same as the default partition percentage settings.

Changing TCAM partition profiles

TCAM is partitioned on the Brocade NetIron XMR and Brocade MLX series routers using a variety of profiles that you can select depending on your application. To implement TCAM partition profiles, enter the **cam-partition profile** command.

cam-partition profile

Syntax: **cam-partition profile** [**ipv4** | **ipv4-ipv6** | **ipv4-vpls** | **ipv4-vpn** | **ipv6** | **l2-metro** | **l2-metro-2** | **mpls-l3vpn** | **mpls-l3vpn-2** | **mpls-vpls** | **mpls-vpls-2** | **mpls-vpn-vpls** | **multi-service**]

You can change the default settings based on your specific needs. Brocade provides the following TCAM partitioning profiles for the Brocade NetIron XMR and Brocade MLX series routers:

- **ipv4** - Optimized for IPv4 applications.
- **ipv4-ipv6** - Optimized for IPv4 and IPv6 dual-stack applications.
- **ipv4-vpls** - Optimized for IPv4 and MPLS VPLS applications.
- **ipv4-vpn** - Optimized for IPv4 and MPLS Layer 3 VPN applications.

- **ipv6** - Optimized for IPv6 applications.
- **l2-metro** and **l2-metro-2** - Optimized for Layer 2 Metro applications.
- **mpls-l3vpn** and **mpls-l3vpn-2** - Optimized for MPLS Layer 3 VPN applications.
- **mpls-vpls** and **mpls-vpls-2** - Optimized for MPLS VPLS applications.
- **mpls-vpn-vpls** - Optimized for MPLS Layer 3 and Layer 2 VPN applications.
- **multi-service** - Optimized for Multi-Service applications.

To display the TCAM settings on your router, use the **show cam-partition** command, as described in "show cam-partition" on page 14.

Determining if a device is running out of TCAM

The **show cam-partition usage** command will tell you if the Brocade NetIron XMR and Brocade MLX series device is running out of TCAM.

Managing memory and CPU usage

To achieve maximum performance, it is important to understand CPU usage and memory issues in the Brocade NetIron XMR and Brocade MLX series router. The following sections discuss how to manage memory and CPU usage.

NOTE
The following commands are available only on MP OS mode. Brocade recommends TAC guidance to execute MP OS mode commands.

- **set sample-task** *task name*
- **set sample-rate** *value*
- **show sample**
- **show bm-dump-mode**
- **show bm-dump-mode hold**
- **show bm-overflow**

Memory and CPU usage show commands

The first step in determining how your device is using memory and CPU is to get a view of the activity. Several show commands display information about CPU usage and CPU task activity. This section lists these commands and provides output examples.

show tasks

Syntax: **show tasks**

This command displays CPU usage statistics for tasks, as shown in the following example.

```
Brocade# show tasks
  Task Name    Pri  State  PC        Stack       Size  CPU Usage(%)  task vid
  -----------  ---  -----  --------  --------  ------  ------------  --------
  idle           0  ready  0000448c  0404dfa0    4096           100         0
  monitor       20  wait   0001493c  0404be10   16384             0         0
```

```
  wd            31  wait   0001493c  0452df48    8192             0         0
  flash         17  wait   0001493c  04535f48    8192             0         0
  dbg           30  wait   0001493c  04532ef0   16384             0         0
  boot          17  wait   0001493c  0462ee08   65536             0         0
  main           3  wait   0001493c  20819f38  131072             0         1
  itc            6  wait   0001493c  2081eb30   16384             0         1
  tmr            5  wait   0001493c  20854670   16384             0         1
  ip_rx          5  wait   0001493c  20859f78   16384             0         1
  scp            5  wait   0001493c  20882670   16384             0         1
  console        5  wait   0001493c  2088d660   32768             0         1
  vlan           5  wait   0001493c  20895660   16384             0         1
  mac_mgr        5  wait   0001493c  2089c670   16384             0         1
  mrp            5  wait   0001493c  20ca3670   16384             0         1
  vsrp           5  wait   0001493c  20caa668   16384             0         1
  snms           5  wait   0001493c  20caf670   16384             0         1
  rtm            5  wait   0001493c  20cb8670   16384             0         1
  ip_tx          5  ready  0001493c  20f33670   16384             0         1
  rip            5  wait   0001493c  27629668   16384             0         1
```

show cpu

Syntax: **show cpu** [**average** | **detail** | **lp**]

- **average** - Displays average CPU utilization information.
- **detail** - Displays detailed information about the CPU utilization.
- **lp** - Displays CPU utilization information for the line card.

This command displays detailed information about the CPU utilization. Command output resembles the following example.

```
Brocade# show cpu detail
Name           State   Act Wait     Hold     Time         CPU
$(idle)        -       A   0        0        257961164    99.2
$con           wait    A   0        139      12372        0.0
$mon           wait    A   0        0        13461        0.0
$flash         wait    A   123      0        11668        0.0
$dbg           wait    A   0        0        5980         0.0
$boot          wait    A   0        4        11986        0.0
main           wait    A   1        2292     4417         0.0
itc            wait    A   0        2        17           0.0
tmr            wait    A   357      0        432901       0.1
ip_rx          wait    A   44       5        144816       0.0
scp            wait    A   33       102      86138        0.0
lpagent        wait    A   2        0        4511         0.0
console        ready   A   55       60       11181        0.0
vlan           wait    A   4        324      19180        0.0
mac_mgr        wait    A   53       135      8971         0.0
mrp            wait    A   139      0        25373        0.0
vsrp           wait    A   140      0        0            0.0
erp            wait    A   140      0        25594        0.0
snms           wait    A   140      31       57475        0.0
rtm            wait    A   141      1382     104725       0.0
rtm6           wait    A   1524     564      77327        0.0
ip_tx          wait    A   2058     592      529010       0.2
rip            wait    A   2651     1        2            0.0
l2vpn          wait    A   2653     71       18055        0.0
mpls           wait    A   2654     135      105867       0.0
nht            wait    A   2657     3        5            0.0
mpls_glue      wait    A   2658     0        1985         0.0
```

```
bgp           wait    A   2659     14       86948       0.0
bgp_io        wait    A   2662     0        486         0.0
ospf          wait    A   2663     49       205871      0.0
ospf_r_calc   wait    A   2667     0        29          0.0
isis          wait    A   2667     42       7640        0.0
isis_spf      wait    A   2668     0        0           0.0
mcast         wait    A   2668     21       21896       0.0
msdp          wait    A   2674     0        2621        0.0
vrrp          wait    A   2674     16       17          0.0
ripng         wait    A   2674     0        0           0.0
ospf6         wait    A   2675     39       33244       0.0
ospf6_rt      wait    A   2675     0        13          0.0
mcast6        wait    A   2675     4        21356       0.0
vrrp6         wait    A   2680     16       16          0.0
bfd           wait    A   2680     6        9           0.0
ipsec         wait    A   2686     0        1           0.0
l4            wait    A   2687     53       16137       0.0
stp           wait    A   2708     1        2           0.0
gvrp_mgr      wait    A   2710     0        0           0.0
snmp          wait    A   2710     0        71          0.0
rmon          wait    A   2711     1        2677        0.0
web           wait    A   2711     3        22285       0.0
lacp          wait    A   2712     1        11803       0.0
dot1x         wait    A   2712     0        0           0.0
dot1ag        wait    A   2713     0        1322        0.0
loop_detect   wait    A   2713     0        7695        0.0
ccp           wait    A   2713     13       1335        0.0
cluster_mgr   wait    A   2713     0        7582        0.0
statistcs     wait    A   2713     0        0           0.0
hw_access     wait    A   2713     3        834977      0.3
sfm_mon       wait    A   32       1        646545      0.2
ntp           wait    A   2714     0        4913        0.0
ospf_msg_task wait    A   1        0        2019        0.0
```

Sampling CPU usage

There are three commands that will show you how much CPU is being used for each task. The first two commands, issued from the management module monitor, identify a task to be sampled, and a rate at which to sample the task:

- **set sample-task** *task name*
- **set sample-rate** *value*
- **show sample**

When **set sample-task** and **set sample-rate** are configured, the **show sample** command samples the CPU for a period of time, and displays stack traces. The resulting information shows you what the CPU is doing, which can be especially helpful during periods of high CPU usage. The maximum number of traces that can be stored is 100. To display the stack traces during the sampling period, enter the **show sample** command.

To stop the sampling, enter the **set sample-rate 0** command.

CPU memory show commands

The CPU uses memory buffers to handle interprocess communication (IPC) and external packets sent and received by the management processor. Buffer pools can consist of 256 bytes, 512 bytes, 1024 bytes, 1040 bytes, and 2048 bytes. All buffers are allocated from these pools on a best-fit basis. The pBuf table maintains start and end addresses, size, stack trace, and number of references for each allocated buffer.

show bm

Syntax: show bm

The **show bm** command determines if a task is suffering from a buffer leak and displays a general overview of the CPU buffer health, as shown in the following example.

```
Brocade# show bm
  SIZE   TOTAL  FREE   IN-USE  OUT-OF-BUF BAD-FREE BAD-REF BAD-SIG
-------------------------------------------------------------------
  256    1023   1022   1            0         0         0        0
  512    1024   1024   0            0         0         0        0
  1024   1024   1024   0            0         0         0        0
  2048   6144   128     6016      3446      0         0        0
  10240  512    512 0           0         0         0        0
 -------------------------------------------------------------------
Pool id = 0 Application Buffer Usage:
Total buf used by appl = 1
loop_detect  (0x7   ) owns 1        buffers
Pool id = 1 Application Buffer Usage:
Total buf used by appl = 0
Pool id = 2 Application Buffer Usage:
Total buf used by appl = 0
Pool id = 3 Application Buffer Usage:
Total buf used by appl = 5632
mac_mgr      (0x3   ) owns 3        buffers
ip_tx        (0xb   ) owns 2        buffers
rtm          (0xc   ) owns 4        buffers
mcast        (0x11  ) owns 11       buffers
console      (0x15  ) owns 1        buffers
ip_rx        (0x1e  ) owns 7        buffers
rtm6         (0x22  ) owns 2        buffers
mcast6       (0x25  ) owns 5        buffers
mpls         (0x29  ) owns 1        buffers
nht          (0x2a  ) owns 2        buffers
ntp          (0x37  ) owns 5593     buffers
isis_spf     (0x5d  ) owns 1        buffers
BM Free App Id Invalid      = 4974 BM Free App Id Not Owner = 15374
BM Get App Ref Error        = 4974 BM Inc App Ref Error     = 0
BM Transfer App Ref Error   = 1102
```

An overview of system activity can be helpful in troubleshooting issues. Too many IN-USE buffers must be justified or there may be memory leaks. BAD-SIG readings may indicate memory corruptions. BAD-REF readings may indicate improper freeing when buffers are shared.

show bm appid

Syntax: show bm appid *num*

The *num* variable specifies the application ID.

This command displays buffer usage information for the specified application ID. Command output resembles the following example.

```
Brocade# show bm appid 6
Pool id = 0 Application Buffer Usage:
Total buf used by appl = 1
vsrp         (0x6   ) owns 0        buffers
Pool id = 1 Application Buffer Usage:
Total buf used by appl = 0
vsrp         (0x6   ) owns 0        buffers
Pool id = 2 Application Buffer Usage:
Total buf used by appl = 0
vsrp         (0x6   ) owns 0        buffers
Pool id = 3 Application Buffer Usage:
Total buf used by appl = 39
vsrp         (0x6   ) owns 0        buffers
```

show bm-dump-mode

Syntax: **show bm-dump-mode**

Use the **show bm-dump-mode** command to pinpoint offending code that may be responsible for double fees and memory leaks. Command output resembles the following example.

```
MP-1 OS> show bm-dump-mode
Buffer dump mode is enabled
```

NOTE
A track state of 0 means that the buffer was allocated before the **show bm-dump-mode** command was executed.

---

show bm-dump-mode hold

Syntax: **show bm-dump-mode hold**

If a buffer leak is suspected, use the **show bm-dump-mode hold** command to help locate the source of the leak, as shown in the following example.

```
MP-1 OS> show bm-dump-mode hold
 Buffer-ID     Second  Dir  Hold  Application
43636ac        60602   rx   1     o
4362ecc        60585   tx   1     1
436298c        60585   tx   1     1
435fcec        60585   tx   1     1
436f10c        60585   tx   1     1
436c18c        60585   tx   1     1
436a4ec        60394   tx   1     1
436bc8c        60387   tx   1     1
43769cc        60369   tx   1     1
43747ac        110     rx   1     0
```

show bm hold

Syntax: **show bm hold**

This command displays hold buffer usage information. Command output resembles the following example.

```
Brocade# show bm hold
Buffer-ID Second  Dir  Hold  Application
  43636ac 60602   rx     1         0
  4362ecc 60585   tx     1         1
  436298c 60585   tx     1         1
  435fcec 60585   tx     1         1
  436f10c 60585   tx     1         1
  436d18c 60585   tx     1         1
  436a4ec 60394   tx     1         1
  436bc8c 60387   tx     1         1
  43769cc 60369   tx     1         1
  43747ac 110     rx     1         0
```

show bm-overflow

Syntax: **show bm-overflow** [**start** | **stop**]

The command stops or starts buffer overflow monitoring.

This command displays buffer overruns, and is enabled by default. Output from this command resembles the following example, where *non-cpu overflow* indicates that the corruption was due to the monitor code and not the application code.

```
MP-1 OS> show bm-overflow
DABR Watch disabled, DABR disabled because no overflow detected yet.
Info for first overflow
bufptr = 0x29558800 payload = 0x2955810 sig_ptr = 0x2955ffc sig = 0x1234eeee
pid = 3  len = 1500  ref = 1  appl_ref = 1  pri = 0  flags = 0x0
Buf Alloc Stack:

Call Stack:
<-372cc<-371d4<-36dfc<-357d0<-358d4<-4074,-80e9040<-8024d58<-80257cc<-80251f0<-8
025a94<-80241d4<-80200e8<-801ff7c<-8473d70<-84086ac<-84088b4<-8474474<-43f0
Buf Data:
00 00 00 00 00 04 00 00-00 00 00 11 81 00 00 0a
0a 00 6e 82 ff ff ff ff-ff ff 00 2a ff ff ff ff
ff ff ff ff ff ff ff ff-ff ff ff ff ff ff ff ff
b3 fa 00 00 00 00 00 04-00 00 00 00 00 11 81 00
. . .
Prev Buf Data
prev_bufptr = 0x2955000 prev_payload = 0x2955030 prev_sig_ptr = 0x29557fc prev_sig
= 0x1234eeee prev_pid = 3 len = 1968
Prev Buf Alloc Stack:
Prev Buf Data:
00 00 00 00 00 04 00 00-00 00 00 11 81 00 00 0a
0a 00 73 6e ff ff ff ff-ff ff 00 29 ff ff ff ff
ff ff ff ff ff ff ff ff-ff ff ff ff ff ff ff ff
ad 4c 0f 16 00 00 4b 16-00 00 00 00 00 04 81 00
00 0a 0a 00 36 ba 00 00-00 00 00 00 00 00 00 00
. . .
next_bufptr = 0x29560000  next_payload = -x2956032  next_sig_ptr = -x29567fc
next_sig = 0x1234eeee  next_pid = 3  len = 88
Next Buf Alloc Stack:
Next Buf Data:
00 00 00 00 00 04 00 00-00 00 00 11 81 00 00 0a
0a 00 d9 d6 ff ff ff ff-ff ff 00 33 ff ff ff ff
ff ff ff ff ff ff ff ff-ff ff ff ff ff ff ff ff
35 95 00 00 00 00 00 04-00 00 00 00 00 11 81 00
. . .
```

```
Info for non-cpu overflow:
Bufptr = NULL
pre_bufptr = NULL
next_bufptr = NULL

Info for DABR hit:
bufptr = 0x2955800  payload = 0x2955810  sig_ptr = 0x2955ffc  sig = 0x123  pid = 3
len = 1500  ref = 0  appl_ref = 0  pri = 0  flags = 0x0
```

Configuration notes

Several things can affect the memory in the Brocade NetIron XMR and Brocade MLX series device:

- When you change the table size for a parameter, device memory is reconfigured. When memory is reconfigured, you must save the change to the startup configuration file, and then reload the software for the change to take effect.
- Because Border Gateway Protocol version 4 (BGP4) can handle a very large number of routes, it requires a great deal of memory. In a typical configuration with a single BGP4 neighbor, a BGP4 router may need to hold up to 150,000 routes. Many configurations, especially those involving more than one neighbor, can require the router to hold even more routes. Brocade NetIron XMR and Brocade MLX series devices provide dynamic memory allocation for BGP4 data by automatically allocating memory when needed to support BGP4 neighbors, routes, and route attribute entries. Dynamic memory allocation is performed automatically by the software and does not require a reload.
- As a guideline, the Brocade NetIron XMR and Brocade MLX series devices with a 2-GB management module can accommodate 150 to 200 neighbors, with the assumption that the device receives a total of about one million routes from all neighbors and sends a total of about eight million routes to neighbors. For each additional one million incoming routes, the capacity for outgoing routes decreases by about two million.
- You can allocate memory for more VLANs or virtual routing interfaces. By default, you can configure up to 512 VLANs and virtual routing interfaces on the router. Although this is the default maximum, the Brocade NetIron XMR and Brocade MLX series devices can support up to 4094 VLANs and 4095 virtual routing interfaces. (VLAN IDs 0 and 4095 are reserved.) If many of your VLANs will have an identical configuration, you might want to configure VLAN groups.

Management module diagnostics

The management modules control Brocade NetIron XMR and Brocade MLX series hardware components, run networking protocols, and provide the Real Time Operating System (RTOS).

Each chassis requires one management module, and can accept a second module for redundancy that works in conjunction with the active management module. If the active management module becomes unavailable, the redundant management module automatically takes over the system operation, minimizing system downtime.

Running management module diagnostics

You can run diagnostics on the management modules to check if the devices needed for proper operation are accessible and in working order. The diagnostics for the Line Processor (LP) modules begin after the completion of diagnostics for the management processor (MP) modules, if the LP modules are present in the chassis.

MP module is considered to have passed the diagnostics if the result of all the checks is "Passed". If an MP or an LP does not pass the **diag burn-in** command, contact Brocade Technical Support for further assistance.

NOTE
Remove the standby management module from the chassis before running the diagnostics. If the standby management module is present, running the diagnostics on the interface module fails.

To run diagnostics on management modules, perform the following steps.

1. Reload the system and immediately press the **B** key repeatedly until the system boots into monitor mode.
2. Type **boot os flash primary** to enter the OS.

   The prompt will change from MP Monitor> to MP OS>.
3. From the MP OS> prompt, enter **diag burn-in**, as shown in the following example.

```
MP-1 OS>diag burn-in
PCI access                - Passed
88E1145 PHY               - Passed
Storage Card              - Passed
M41T11 RTC                - Passed
  FE (slot 0; FE 0; 0x11fe6000) access passed;
  FE (slot 0; FE 1; 0x11fe6000) access passed;
  FE (slot 0; FE 2; 0x11fe6000) access passed;

  FE (slot 1; FE 0; 0x11fe6000) access passed;
  FE (slot 1; FE 1; 0x11fe6000) access passed;
  FE (slot 1; FE 2; 0x11fe6000) access passed;

SAND access               - Passed
Valere power Supply 0 Passed
Valere power Supply 1 Passed
Power Supply access       - Passed
  Port 0 passed
  Port 1 passed
  Port 2 passed
  Port 3 passed
  Port 4 passed
  Port 5 passed
  Port 6 passed
  Port 7 passed
  Port 8 passed
 Port 9 passed
  Port 10 passed
  Port 11 passed
  Port 12 passed
  Port 13 passed
  Port 14 passed
  Port 15 passed
  Port 16 passed
```

```
  Port 17 passed
  Port 18 passed
  Port 19 passed
  Port 23 passed
Dx246 Switch Port Loopback - Passed

###- PASS -###
MP-1 OS>
LP (6) [MLX-X 1Gx24 Copper] burn-in started
LP (6) PING test passed
LP (7) [MLX-X 1Gx24 Copper] burn-in started
LP (7) PING test passed

LP (6) (MLX-X 1Gx24 Copper) diagnostic Passed
LP (7) (MLX-X 1Gx24 Copper) diagnostic Passed

###- PASS -###
```

NOTE
After the completion of diagnostics for the MP modules, the system displays the MP-1 OS> prompt and then starts the diagnostics for the LP modules.

NOTE
Brocade requires that you remove physical connections to all ports on the module, and all optics to all ports on the module, so the module does not receive traffic while the diagnostics are running.

---

4. Enter the **reset** command to return the system to normal operation (system reboot).

```
MP-1 OS>reset
```

```
REBOOT S1: NI-XMR-1Gx20-SFP 20-port 1GbE/100FX Module CARD_STATE_REBOOT 20   0000.003d.8500
BOOT  S1: NI-XMR-1Gx20-SFP 20-port 1GbE/100FX Module CARD_STATE_BOOT  20   0000.003d.8500
CARD_STATE_UP S1: NI-XMR-1Gx20-SFP 20-port 1GbE/100FX Module CARD_STATE_SW_LOADED 20
0000.003d.8500
UP    S1: NI-XMR-1Gx20-SFP 20-port 1GbE/100FX Module  CARD_STATE_UP    20   0000.003d.8500
```

After the system reboots, you can display the status of the module using the **show module** command, as shown in the following example.

```
Brocade# show module
        Module                                  Status                 Ports  Starting MAC
M1 (upper): NI-XMR-MR Management Module         Active
M2 (lower):
F1: NI-X-SF Switch Fabric Module                Active
F2: NI-X-SF Switch Fabric Module                Active
F3: NI-X-SF Switch Fabric Module                Active
F4: NI-X-SF Switch Fabric Module                Active
S1: NI-XMR-1Gx20-SFP 20-port 1GbE/100FX Module  CARD_STATE_SW_LOADED    20    0000.003d.8500
S2:
S3: NI-XMR-1Gx20-SFP 20-port 1GbE/100FX Module  CARD_STATE_UP           20    0000.003d.8550
```

Management modules

Table 3 lists the management modules that are available for the Brocade NetIron XMR and Brocade MLX series routers.

TABLE 3 Management modules

Part number	Description
NI-MLX-MR	Brocade MLXe and NetIron MLX management module, 1 GB SDRAM, dual PCMCIA slots, EIA or TIA-232 and 10/100/1000 Ethernet ports for out-of-band management.
NI-MLX-32-MR	Brocade MLXe-32 and NetIron MLX-32 management module, 1 GB SDRAM, dual PCMCIA slots, EIA or TIA-232 and 10/100/1000 Ethernet ports for out-of-band management.
NI-XMR-MR	NetIron XMR management module, 2 GB SDRAM, dual PCMCIA slots, EIA or TIA-232 and 10/100/1000 Ethernet ports for out-of-band management.
NI-XMR-32-MR	NetIron XMR 32000 management module, 2 GB SDRAM, dual PCMCIA slots, EIA or TIA-232 and 10/100/1000 Ethernet ports for out-of-band management.

Management module show commands

This section describes the show commands that display information about management modules.

show version

Syntax: **show version**

This command displays information about your management modules.

```
Brocade# show version
HW: NetIron XMR Router
Backplane (Serial #: Not Exist,  Part #: Not Exist)
NI-X-SF Switch Fabric Module 1 (Serial #: PR29050242,  Part #: 31523-100A)
FE 1: Type 00000,  Version 0
FE 3: Type 00000,  Version 0
NI-X-SF Switch Fabric Module 2 (Serial #: PR29050246,  Part #: 31523-100A)
FE 1: Type 00000,  Version 0
FE 3: Type 00000,  Version 0
NI-X-SF Switch Fabric Module 3 (Serial #: PR30050270,  Part #: 31523-100A)
FE 1: Type 00000,  Version 0
FE 3: Type 00000,  Version 0
==============================================================================
SL M1: NI-XMR-MR Management Module Active (Serial #: PR30050511, Part #: 31524-0
00A):
Boot     : Version 3.5.0T165 Copyright (c) 1996-2009 Brocade Communications, Inc.
Compiled on May 31 2009 at 14:42:00 labeled as xmprm03500b155
 (424038 bytes) from boot flash
Monitor  : Version 3.5.0T165 Copyright (c) 1996-2009 Brocade Communications, Inc.
Compiled on May 31 2009 at 14:41:14 labeled as xmb03500b155
 (424045 bytes) from code flash
IronWare : Version 3.5.0T163 Copyright (c) 1996-2009 Brocade Communications, Inc.
Compiled on Jun 11 2009 at 07:15:58 labeled as xmr03500b181
 (5816681 bytes) from Primary
Board ID : 00 MBRIDGE Revision : 18
--More--, next page: Space, next line: Return key, quit: Control-c^C
```

show module

Syntax: show module

This command displays the status of the management modules. Enter this command at any CLI level. Command output resembles the following example.

```
Brocade# show module
Module Status Ports Starting MAC
M1 (left): NI-MLX-MR Management Module Active
M2 (right): NI-MLX-MR Management Module Standby (Ready)
```

Management module status is either active or standby. Standby modules can be in one of the following modes:

- **Init** – The module is currently initializing as the standby module.
- **Ready** – The module is ready to take over as the active module, if necessary.
- **Wait** – The module is awaiting boot information from the active management module.
- **Sync** – The active module is currently synchronizing files with the standby module.

show redundancy

Syntax: show redundancy

This command displays module switchover activity, as shown in the following example.

```
Brocade# show redundancy
=== MP Redundancy Settings ===
Default Active Slot = 17
Running-Config Sync Period = 7 seconds
=== MP Redundancy Statistics ===

Current Active Session:
Active Slot = 9, Standby Slot = 10 (Ready State), Switchover Cause = No Switchover
Start Time = 0-0-17 19:47:39 (Wednesday)

Previous Active Session #1:
Active Slot = 10, Standby Slot = 9, Switchover Cause = Active Rebooted
Start Time = 0-0-17 19:46:9 (Wednesday), End Time = 0-0-17 19:47:39 (Wednesday)

Previous Active Session #2:
Active Slot = 9, Standby Slot = 10, Switchover Cause = Active Rebooted
Start Time = 0-0-17 19:44:14 (Wednesday), End Time = 0-0-17 19:46:9 (Wednesday)
```

show log

Syntax: show log

This command allows you to view the system log or the traps logged on an SNMP trap receiver, as shown in the following example, which indicates that one switchover occurred on the management module in slot 9.

```
Brocade# show log
Syslog logging: enabled (0 messages dropped, 0 flushes, 0 overruns)
Buffer logging: level ACDMEINW, 24 messages logged
level code: A=alert C=critical D=debugging M=emergency E=error
I=informational N=notification W=warning
Static Log Buffer:
Sep 28 11:31:25:A:Power Supply 1, 1st left, not installed
Sep 28 11:31:25:A:Power Supply 3, middle left, not installed
Sep 28 11:31:25:A:Power Supply 4, middle right, failed
Sep 28 11:31:25:A:Power Supply 5, 2nd right, not installed
Dynamic Log Buffer (50 lines):
Sep 27 18:06:58:I:Interface ethernet6/2, state up
Sep 27 18:06:57:I:Interface ethernet3/2, state up
Sep 27 15:39:42:I:Interface ethernet3/2, state up
Sep 27 15:39:42:I:Interface ethernet6/2, state up
...
Sep 27 14:23:45:N:Module up in slot 6
Sep 27 14:23:45:N:Module up in slot 3
Sep 27 14:23:27:A:Management module at slot 9 state changed from standby to
active
```

Management module debug commands

There are no debug commands that are specific to management modules.

Configuration notes

Management sessions can be established through the management port on the active management module. If a switchover occurs, the management port on the original active module shuts down and all open CLI, Web management interface, and Brocade Network Advisor (BNA) sessions close. You can open new sessions with the new active module after the switchover, if the new module has a management port.

For example, if you were accessing the Web management interface using a PC connected to the original active management port, you can open a new session if a PC is connected to the management port on the new active module. Open a new session using the same IP address you used before the switchover. (If a switchover occurs, the IP address you configured on the original active module is automatically assumed by the new active module.)

Common diagnostic scenarios

Management module switchover events

In a *redundant* management module configuration, whenever the standby management module assumes the role of active module a *switchover* event has occurred. This happens when the active module becomes unavailable (for example, power is lost or a component fails), you perform a manual switchover, or you remove or replace the active management module. When a switchover occurs between the active and standby modules, management sessions, syslog, and SNMP traps may be affected.

When an active module becomes unavailable

The following events will cause an active module to become unavailable and trigger a switchover:

- An active module experiences a problem significant enough to cause a reset of the module. If this is a software problem, capture the output from the **show tech** command. If the output contains a crash dump, contact Brocade Technical Support.
- The active module loses power. This may happen if a power supply fails, or because of a general disruption in power.

Before a switchover occurs, the active module resets itself and sends an interrupt signal to the standby module. The standby module then becomes the active module, allowing the modules to continue forwarding traffic.

The new active module also begins to manage the system. When the original active module again becomes available or is replaced, the new active module resumes in the role of standby.

Management module error messages

The following messages are displayed in the event of a switchover event, if there is a problem with the standby management module in a redundant configuration.

```
Warning: Active MP running image is not in its flash or PCMCIA card, image
synchronization is not possible:

Warning: Standby Module is not allowed - put standby MP in reset.
```

Switchover syslog and SNMP traps

When a switchover occurs, the system sends a syslog message to the local syslog buffer and to the syslog server, if one is configured. The system also sends an SNMP trap to the receiver, if one is configured.

If the system is reset as the result of a switchover to the standby management module, the system sends a warm start message and trap.

Monitoring management module redundancy

You can monitor the following aspects of management module redundancy:

- The status of the management modules (if a module is the active or standby module).
- The switchover history for the management modules.

Management module LEDs

Management modules contain six LEDs that indicate operational status. Once you have installed a management module and powered on the system, you can read the LED indicators on the module faceplate. Table 4 lists these LEDs and describes what to do if an LED shows that the module is not operating properly.

TABLE 4 Management module LED indicators

LED	Position	State	Meaning
Slot 1 and Slot 2	Adjacent to the PCMCIA slot it represents	On or blinking	The software is currently accessing the flash card.
		Off	The software is not accessing a PCMCIA flash card inserted in a slot. If this occurs, you must: • Make sure the flash card is fully seated in the connector. • From a command prompt, type **dir/slot1** or **dir/slot 2** to see if either slot is readable.
Active	Lower left	On	The module is functioning as the active management module.
		Off	The module is functioning as the standby management module.
Pwr	Upper left	On	The module is receiving power.
		Off	The module is not receiving power. If this occurs, you must: • Make sure the module is seated properly. • Make sure the power supply is operating properly. The Power LED must be green. • Try the module in another slot or chassis.
10/100/1000 Ethernet port	Above and to right of RJ-45 connector	On (green)	A link is established with the remote port.
		Off	No link is established with the remote port. If this occurs, you must use a straight-through cable to the switch to verify that interface management is enabled and the link is good.
10/100/1000 Ethernet Port	Above and to left of RJ-45 connector	On or blinking (yellow)	The port is transmitting and receiving packets.
		Off for extended period	The port is not transmitting or receiving packets. If this occurs, you must do the following tasks: • Verify that the port is enabled and configured properly for auto-negotiation. • Try another port to identify a hardware failure.

Interface module diagnostics

NOTE
Brocade requires that you remove physical connections to all ports on the module, and all optics to all ports on the module, so the module does not receive traffic while the diagnostics are running.

NOTE
Brocade recommends that you run diagnostics on the module running with factory default configuration, so that the module does not work any protocol process while the diagnostics are running. If you need to configure some commands such as **snmp-server max-ifindex-per-module** *num* to recognize the modules, the minimum configuration to recognize would be approved.

To run diagnostics on an interface module, perform the following steps.

1. Boot the module into interactive mode by entering the following commands.

```
Brocade# lp boot sys in 1
Brocade# Reset slot 1
Slot 1: booted to Interactive Mode.
```

2. With the module in interactive mode, remote console to the module by entering the following command.

```
Brocade# rcon 1
Remote connection to LP slot 1 established
Press CTRL-X or type 'exit' to disconnect it
LP-1 Monitor>
```

3. Boot the module in the OS mode.

```
LP-1 Monitor>boot os flash primary
LP-1 OS>
```

4. Run the diagnostic using the **diag burn** command. You must see output similar to the following example.

```
LP-1 OS>diag burn
PRAM 0 -- TM DDRII support disabled
XPP PLL Status Register at 0 micro-sec = 0x0000ff0f
XPP PLL Status Register at 2 micro-sec = 0x0000ff0f
XPP PLL Status Register at 4 micro-sec = 0x0000ff0f
XPP PLL Status Register at 6 micro-sec = 0x0000ff0f
XPP PLL Status Register at 8 micro-sec = 0x0000ff0f
XPP PLL Status Register at 10 micro-sec = 0x0000ff0f
XPP PLL Status Register at 12 micro-sec = 0x0000ff0f
XPP PLL Status Register at 14 micro-sec = 0x0000ff0f
XPP PLL Status Register at 16 micro-sec = 0x0000ff0f
XPP PLL Status Register at 18 micro-sec = 0x0000ff0f
XPP PLL Status Register at 20 micro-sec = 0x0000ff0f
XPP PLL Status Register at 22 micro-sec = 0x0000ff0f
XPP PLL Status Register at 24 micro-sec = 0x0000ff0f
XPP PLL Status Register at 26 micro-sec = 0x0000ff0f
XPP PLL Status Register at 28 micro-sec = 0x0000ff0f
XPP Reset Sequence Done -  PLL Status Register = 0x0000ff0f
XPP Reset Sequence Done -  PLL Status Register = 0x0000ff0f
XPP Reset Sequence Done -  PLL Status Register = 0x00c0ffff
XPP: P1 board or higher detected
PASSED
Dev 0 PRAM passed
STATSRAM 0 -- PASSED
Dev 0 STATSRAM passed
TXVLAN Table 0 -- PASSED
Dev 0 TXVLANRAM passed
CAM2PRAM 0 -- PASSED
Dev 0 CAMTOPRAMRAM passed
AGERAM 0 -- Pass 1
Pass 2

AGERAM memory tested for 2097152 entries
PASSED
Dev 0 AGERAM passed
ServTypeTable 0 -- PASSED
Dev 0 SERVTYPETABLERAM passed
TXNEXTHOP TABLE 0 -- PASSED
```

```
Dev 0 TXNEXTHOPTABLERAM passed
XPP 0 TOS TABLE -- PASSED
Dev 0 TOSTABLERAM passed
XPP 0 MULTICAST START OFFSET TABLE -- PASSED
Dev 0 MCASTSTARTTABLERAM passed
XPP 0 MULTICAST REPLACEMENT TABLE -- PASSED
Dev 0 MCASTREPLTABLERAM passed
PRAM 1 -- TM DDRII support disabled
XPP PLL Status Register at 0 micro-sec = 0x0000ff0f
XPP PLL Status Register at 2 micro-sec = 0x0000ff0f
XPP PLL Status Register at 4 micro-sec = 0x0000ff0f
XPP PLL Status Register at 6 micro-sec = 0x0000ff0f
XPP PLL Status Register at 8 micro-sec = 0x0000ff0f
XPP PLL Status Register at 10 micro-sec = 0x0000ff0f
XPP PLL Status Register at 12 micro-sec = 0x0000ff0f
XPP PLL Status Register at 14 micro-sec = 0x0000ff0f
XPP PLL Status Register at 16 micro-sec = 0x0000ff0f
XPP PLL Status Register at 18 micro-sec = 0x0000ff0f
XPP PLL Status Register at 20 micro-sec = 0x0000ff0f
XPP PLL Status Register at 22 micro-sec = 0x0000ff0f
XPP PLL Status Register at 24 micro-sec = 0x0000ff0f
XPP PLL Status Register at 26 micro-sec = 0x0000ff0f
XPP PLL Status Register at 28 micro-sec = 0x0000ff0f
XPP Reset Sequence Done -  PLL Status Register = 0x0000ff0f
XPP Reset Sequence Done -  PLL Status Register = 0x0000ff0f
XPP Reset Sequence Done -  PLL Status Register = 0x00c0ffff
XPP: P1 board or higher detected
```

5. Once the diagnostics are complete, return the interface module to operational status by entering the following commands.

```
Brocade# lp boot sys flash pri 1
Brocade# reset slot 1
```

6. Reconnect the ports you disconnected prior to running the tests.

Interface modules

Table 5 lists the interface modules that are available for the Brocade NetIron XMR and Brocade MLX series routers.

TABLE 5 Interface modules

Part number	Description
NI-MLX-10Gx2	2-port 10 GbE module with IPv4, IPv6, and MPLS hardware support--Requires XFP optics.
NI-MLX-1Gx20-SFP	20-port FE or GE (100/1000) module with IPv4, IPv6, and MPLS hardware support--Requires SFP optics.
NI-MLX-1Gx20-GC	20-port 10/100/1000 copper module with IPv4, IPv6, and MPLS hardware support.
NI-X-OC192x2	2-port Packet over SONET (SDH) OC-192 (STM-64) interface module.
NI-X-OC192x1	1-port Packet over SONET (SDH) OC-192 (STM-64) interface module.
NI-X-OC48x8	8-port Packet over SONET (SDH) OC-12/48 (STM-4/16) interface module.
NI-X-OC48x4	4-port Packet over SONET (SDH) OC-12/48 (STM-4/16) interface module.
NI-X-OC48x2	2-port Packet over SONET (SDH) OC-12/48 (STM-4/16) interface module.

TABLE 5 Interface modules (Continued)

Part number	Description
NI-MLX-1Gx48-T-A	48-port 10/100/1000BASE-T mini-RJ-21 interface module with IPv4, IPv6, and MPLS hardware support.
NI-MLX-10Gx8-M	8-port 10 Gbps Ethernet (M) module with IPv4, IPv6, and MPLS hardware support - Requires SFP+ optics, high-speed fabric modules, and high-speed fans (NIBI-16-FAN-EXH-A) on 16-slot routers. Supported for Brocade MLX series devices only.
NI-MLX-10Gx8-D	8-port 10 Gbps Ethernet (D) module with IPv4 and IPv6 hardware support - Requires SFP+ optics, high-speed fabric modules, and high-speed fans (NIBI-16-FAN-EXH-A) on 16-slot routers. Supported for Brocade MLX series devices only.
NI-MLX-10Gx4	4-port 10 Gbps Ethernet module with XFP optical interfaces for wire-speed performance.
NI-MLX-1Gx24-GC	24-port 1 Gbps Ethernet copper module with RJ-45 interfaces for wire-speed performance.
NI-MLX-1Gx24-GF	24-port 1 Gbps fiber module for wire-speed performance.
24x1G-SFP	24-port 1 Gbps Ethernet module with SFP optical interfaces.
NI-XMR-10Gx4	4-port 10 Gbps Ethernet module with IPv4, IPv6, and MPLS hardware support--Requires XFP optics.
NI-XMR-10Gx2	2-port 10 Gbps Ethernet module with IPv4, IPv6, and MPLS hardware support--Requires XFP optics.
NI-XMR-1Gx20-SFP	20-port FE/GE (100/1000) module with IPv4, IPv6, and MPLS hardware support--Requires SFP optics.
NI-XMR-1Gx20-GC	20-port 10/100/1000 copper module with IPv4, IPv6, and MPLS hardware support.

Interface module show commands

This section describes the show commands that display information about interface modules.

show media

Syntax: show media

This command displays information about the media installed in ports, as shown in the following example, which provides the type, vendor, part number, version, and serial number of the SFP or XFP device installed in a port. If no SFP or XFP device is installed in a port, the Type field displays “N/A”, the Vendor field is empty, and the other fields display “Unknown”.

```
Brocade# show media
Port 1/1:
    Type  : OC-48 SR-1 (12km) - 1310nm
    Vendor:     OCP             , Version:                 0000
    Part# :     TRPA48S1XBAM    , Serial#:     4274425
Port 1/2:
    Type  : OC-48 SR-1 (2km) - 1310nm
    Vendor:     WaveSplitter    , Version:
    Part# :     WST-S3CDCS      , Serial#:     SC0704150067
Port 1/3:
    Type  : N/A
    Vendor:                     , Version:              Unknown
    Part# :              Unknown, Serial#:              Unknown
Port 1/4:
    Type  : N/A
    Vendor:                     , Version:              Unknown
    Part# :              Unknown, Serial#:              Unknown
Port 3/1:
    Type  : OC-48 SR-1 (2km) - 1310nm
    Vendor:     JDS UNIPHASE    , Version:                 D25
    Part# :     CT2-MS1LBTD33C5 , Serial#:     6331003584
Port 3/2:
    Type  : OC-48 IR-1 (15km) - 1310nm
    Vendor:     Brocade Communications, Version:                 A
```

show optics

Syntax: show optics *slot number*

This command displays optics information for XFP and SFP ports, including temperature, transmit power, receive power, transmit bias, and current.

```
Brocade# show optics 4
Port Temperature Tx Power Rx Power Tx Bias Current
+----+-----------+---------+-----------+-------------------+
4/1 30.8242 C -001.8822 dBm -002.5908 dBm 41.790 mA
Normal Normal Normal Normal
4/2 31.7070 C -001.4116 dBm -006.4092 dBm 41.976 mA
Normal Normal Normal Normal
4/3 30.1835 C -000.5794 dBm 0.000 mA
Normal Low-Alarm Normal Low-Alarm
4/4 0.0000 C 0.000 mA
Normal Normal Normal Normal
```

show tm-voq-stat

Syntax: show tm-voq-stat [**src_port** [**pos** *slotnum/portnum* | **ethernet** *slotnum/portnum*] [**dst_port** [**pos** *slotnum/portnum* | **ethernet** *slotnum/portnum*]] *priority*]

This command displays traffic management statistics for interface modules, as shown in the following example.

```
Brocade# show tm-voq-stat src_port pos 1/1 dst_port pos 3/1 0
   EnQue Pkt Count                            0
   EnQue Bytes Count                          0
   DeQue Pkt Count                            0
   DeQue Bytes Count                          0
   Total Discard Pkt Count                    0
   Total Discard Bytes Count                  0
   Oldest Discard Pkt Count                   0
   Oldest Discard Bytes Count                 0
   WRED Dropped Pkt Count                     0
   WRED Dropped Bytes Count                   0
   Current Queue Depth                        0
   Maximum Queue Depth since Last read        0
```

show tm statistics

Syntax: show tm statistics slot *slot number*

You can monitor traffic manager statistics to ensure that traffic load balancing is working properly. The following example shows traffic statistics for an interface module identified by its slot number.

```
Brocade# show tm statistics slot 4
--------- Ports 4/1 - 4/20 ---------
Ingress Counters:
Total Ingress Pkt Count: 22
EnQue Pkt Count: 0
EnQue Byte Count: 0
DeQue Pkt Count: 0
DeQue Byte Count: 0
TotalQue Discard Pkt Count: 0
TotalQue Discard Byte Count: 0
Oldest Discard Pkt Count: 0
Oldest Discard Byte Count: 0
Egress Counters:
EnQue Pkt Count: 0
EnQue Byte Count: 0
Discard Pkt Count: 0
Discard Byte Count: 0
```

show version

Syntax: show version

This command displays information about your system, including all installed interface modules, as shown in the following example.

```
Brocade# show version
Monitor  : Version 3.5.0B6T165 Copyright (c) 1996-2009 Brocade Communications,
Inc.
Compiled on Jul  9 2009 at 18:46:18 labeled as xmb03500B6
 (424489 bytes) from code flash
IronWare : Version 3.5.0T163 Copyright (c) 1996-2009 Brocade Communications, Inc.
Compiled on Jul 19 2009 at 07:17:58 labeled as xmr03500b279
 (5836796 bytes) from Primary
Board ID : 00 MBRIDGE Revision : 18
916 MHz Power PC processor  (version 8003/0101) 166 MHz bus
512 KB Boot Flash (AM29LV040B), 32 MB Code Flash (MT28F128J3)
2048 MB DRAM
Standby Management uptime is 1 days 52 minutes 24 seconds
==========================================================================
```

```
SL 2: NI-X-OC48x4 4-port OC48/12 STM16/STM4 Module (Serial #:PR50060005,Part #:
35650-00BA)
Boot : Version 3.5.0B6T175 Copyright (c) 1996-2009 Brocade Communications, Inc.
Compiled on Jul  9 2009 at 18:47:52 labeled as xmlprm03500B6
 (387074 bytes) from boot flash

Monitor  : Version 3.5.0B6T175 Copyright (c) 1996-2009 Brocade Communications,
Inc.
Compiled on Jul  9 2009 at 18:48:28 labeled as xmlb03500B6
 (387059 bytes) from code flash
Compiled on Jul 19 2009 at 07:50:18 labeled as xmlp03500b279
 (3163145 bytes) from Primary
IronWare : Version 3.5.0T177 Copyright (c) 1996-2009 Brocade Communications, Inc.
Compiled on Jul 19 2009 at 07:50:18 labeled as xmlp03500b279
 (3163145 bytes) from Primary
FPGA versions:
Valid PBIF Version = 2.11, Build Time = 4/3/2009 13:0:00
Valid XPP Version = 3.11, Build Time = 5/14/2009 14:22:00
Valid STATS Version = 2.04, Build Time = 4/26/2009 12:12:00
666 MHz MPC 8541 (version 8020/0020) 333 MHz bus
512 KB Boot Flash (AM29LV040B), 16 MB Code Flash (MT28F640J3)
1024 MB DRAM, 8 KB SRAM, 0 Bytes BRAM
PPCR0: 1024K entries CAM, 16384K PRAM, 2048K AGE RAM
LP Slot 2 uptime is 1 days 52 minutes 24 seconds
(3163145 bytes) from Primary
==========================================================================
SL 3: NI-XMR-1Gx20-GC 20-port 10/100/1000 Copper Module (Serial #: SA33061518,
Part #: 35555-200A)
Boot     : Version 3.5.0B6T175 Copyright (c) 1996-2009 Brocade Communications,
Inc.
Compiled on Jul  9 2009 at 18:48:28 labeled as xmlb03500B6
 (387059 bytes) from code flash
Compiled on Jul  9 2009 at 18:47:52 labeled as xmlprm03500B6
 (387074 bytes) from boot flash
Monitor  : Version 3.5.0B6T175 Copyright (c) 1996-2009 Brocade Communications,
Inc.
Compiled on Jul  9 2009 at 18:48:28 labeled as xmlb03500B6
 (387059 bytes) from code flash
IronWare : Version 3.5.0T177 Copyright (c) 1996-2009 Brocade Communications, Inc.
Compiled on Jul 19 2009 at 07:50:18 labeled as xmlp03500b279
FPGA versions:
Valid PBIF Version = 2.13, Build Time = 2/14/2009 11:55:00
Valid XPP Version = 3.05, Build Time = 5/14/2009 11:12:00
BCM5695GMAC 0
BCM5695GMAC 1
BCM5695GMAC 2
BCM5695GMAC 3
666 MHz MPC 8541 (version 8020/0020) 333 MHz bus
512 KB Boot Flash (AM29LV040B), 16 MB Code Flash (MT28F640J3)
1024 MB DRAM, 8 KB SRAM, 0 Bytes BRAM
PPCR0: 1024K entries CAM, 16384K PRAM, 2048K AGE RAM
LP Slot 3 uptime is 1 days 52 minutes 23 seconds
==========================================================================
SL 4: NI-XMR-10Gx2 2-port 10GbE Module (Serial #: PR34050032, Part #: 31546-100A
)
Boot     : Version 3.5.0B6T175 Copyright (c) 1996-2009 Brocade Communications,
Inc.
Compiled on Jul  9 2009 at 18:47:52 labeled as xmlprm03500B6
 (387074 bytes) from boot flash
Monitor  : Version 3.5.0B6T175 Copyright (c) 1996-2009 Brocade Communications,
```

```
Inc.
Compiled on Jul  9 2009 at 18:48:28 labeled as xmlb03500B6
 (387059 bytes) from code flash
IronWare : Version 3.5.0T177 Copyright (c) 1996-2009 Brocade Communications, Inc.
Compiled on Jul 19 2009 at 07:50:18 labeled as xmlp03500b279
 (3163145 bytes) from Primary
FPGA versions:
Valid PBIF Version = 2.13, Build Time = 2/14/2009 11:55:00

Valid XPP Version = 3.05, Build Time = 5/14/2009 11:12:00

Valid XGMAC Version = 0.11, Build Time = 10/11/2009 12:45:00
```

Clearing traffic management statistics

To clear traffic management statistics, enter the **clear tm-voq-stat src_port dst_port** command.

The following commands can be useful for displaying and clearing information about interface modules:

- **clear tm-voq-stat src_port dst_port [priority]**
- **show tm-voq-stat src_port multicast**
- **clear tm-voq-stat src_port multicast**
- **show tm-voq-stat src_port cpu-queue**
- **clear tm-voq-stat src_port cpu-queue**

Interface module debug commands

There are no debug commands specific to interface modules.

Common diagnostic scenarios

Common diagnostic situations involving interface modules can include the following situations:

- The module is not receiving power.
- A link has not been successfully established.
- A port is not transmitting or receiving packets.
- Packets fail to enter an ingress queue on the traffic manager. This can happen because the queue has reached its maximum length, or Weighted Random Early Detection (WRED) is enabled to prevent an output queue from ever filling to capacity.

Interface modules continue to send traffic when a switchover occurs between management modules. After a switchover event, interface modules send updates to the new active management module, which verifies that the interface modules are synchronized.

If the new active management module becomes out of sync with an interface module, information on the interface module may be overwritten, which can cause an interruption of traffic forwarding. This must only occur if there is a Layer 3 topology change elsewhere in the network during the management module switchover. Brocade NetIron XMR and Brocade MLX series routers support

Layer 3 hitless failover with graceful restart for high availability routing in protocols such as BGP and OSPF. With these high availability features enabled, when a router experiences a failover or restart, forwarding disruptions are minimized, and route flapping is diminished to provide continuous service during a switchover or restart event.

IPC diagnostics

Interprocess communication (IPC) is a set of techniques that manage the exchange of data between two or more processors running on one or more computers connected by a network. IPC methods include message passing, synchronization, shared memory, and remote procedure calls (RPCs).

IPC show commands

This section describes the show commands that display IPC information.

ipc show dy-sync

Syntax: ipc show dy-sync

This command displays dynamic IPC sync statistics. Command output resembles the following example.

```
Brocade# ipc show dy-sync
ARP table sync_type=2, enabled=1
b_cast: serial # 000004DA, packets 1242, msg 1242, hello=160454
b_cast buf=02962862, index=20, msg_i=0
u_cast  1, msg # 1, dropped (alloc failure)=0
u_specific reply 0, u_specific miss 0
10/1:1
The above is slot/cpu u_cast pkt#
DAI table sync_type=30, enabled=1
b_cast: serial # 00000001, packets 1, msg 2, hello=160454
b_cast buf=0298B862, index=20, msg_i=0
u_cast  1, msg # 2, dropped (alloc failure)=0
u_specific reply 0, u_specific miss 0
10/1:1
The above is slot/cpu u_cast pkt#
Label to VRF ta sync_type=25, enabled=1
b_cast: serial # 00000000, packets 0, msg 0, hello=160454
b_cast buf=0296A862, index=20, msg_i=0
u_cast  1, msg # 0, dropped (alloc failure)=0
u_specific reply 0, u_specific miss 0
10/1:1
The above is slot/cpu u_cast pkt#
ND6 neighbor ta sync_type=12, enabled=1
b_cast: serial # 000005D5, packets 1493, msg 1493, hello=160454
b_cast buf=0299E862, index=20, msg_i=0
u_cast  2, msg # 1, dropped (alloc failure)=0
u_specific reply 0, u_specific miss 0
10/1:2
The above is slot/cpu u_cast pkt#
NHT sync_type=23, enabled=1
b_cast: serial # 00000000, packets 0, msg 0, hello=160454
b_cast buf=02969862, index=20, msg_i=0
```

```
u_cast  118, msg # 4096, dropped (alloc failure)=0
u_specific reply 0, u_specific miss 0

10/1:118
The above is slot/cpu u_cast pkt#
...
```

ipc show heartbeat history-log

Syntax: ipc show heartbeat history-log

This command displays the IPC heartbeat error log from standby MP and LP based on the Color-coded Heartbeat (CCHB) received on the active MP.

The following is the sample output from the **ipc show heartbeat history-log** command.

```
Brocade# ipc show heartbeat history-log
May  8 06:19:44: Stand By MP : CPU Rate    (above 80%), Low Buffer (Y), IPC Tx
QFull (N), IPC Rx QFull (Y)
           IPC Unrel Tx Error (N), IPC Rel Tx Error (Y), IPC Send Buffer Error (N)
           IPC Send Buffer-retransmit Error (Y), IPC QFull Delay (Y), IPC Rel Tx
Delay Error (N)

May  8 06:19:42: Slot 8 : CPU Rate    (above 80%), Low Buffer (Y), IPC Tx QFull
(N), IPC Rx QFull (Y)
           IPC Unrel Tx Error (N), IPC Rel Tx Error (Y), IPC Send Buffer Error (N)
           IPC Send Buffer-retransmit Error (Y), IPC QFull Delay (Y), IPC Rel Tx
Delay Error (N)
```

ipc show names

Syntax: ipc show names

This command displays IPC message type names, with numbers that correspond to the **ipc show statistics** output.

```
Brocade# ipc show names
Names of registered IPC message types:
2 IPC reset message
3 IPC startup message
5 IPC Test message
8 File Tx End
12 File Tx Request
13 LP Card Boot
14 LP Card SW Loaded
15 LP Card Oper Info
16 LP Ports Oper Info
17 LP Port Oper Info
18 LP Stripe Sync Send Done
19 LP Stripe Sync Status
31 LP Stripe Sync Loss
32 Slot Info
33 Module Reboot
40 MP Red Standby Boot Info
41 MP Red Standby SW Loaded
45 MP Red Cmd
49 MP Red Standby Info
50 MP SW Upgrade Info
72 HAL Set Port Value
74 HAL Set Port Values
```

```
76 HAL Set Ports Value
78 HAL Set Port Active Mac Address
80 HAL Set Port Active Mac Addresses
82 HAL Set Port Data
84 HAL Set Module Value
86 SET LP PROM
93 HAL Add Fid Portmask
95 HAL Delete Card
97 HAL Add Card
104 ACL Mirror Status from LP
106 Trunk query response
110 LP MAC table sync
113 LP MAC free
115 LP MAC SA learn
116 LP MAC DA learn
120 Port security
143 VLAN LP ACK
161 VSRP Update Session
180 AS_PATH REQ
182 IPV6 NEIGHBOR REQUEST
190 LP Mcast Main
191 L2 Mcast msg
192 L2 Mcast6 msg
202 ARP LP REQ
208 AuthReq from LP
209 LP dot1ag Main210 L4 STATUS FROM LP
211 POS Status Update
214 LP CLI show stat Value
215 MP LP show CLI
216 SYNC RTC req
217 SysLog LP message
219 LP Temperature Res
232 VPLS MAC SYNC
233 IPC Text Msg
235 Port Stat Req
236 Port Stat Sampling
237 IPC_MSGTYPE_MP_SYNC_REQ_0
238 IPC_MSGTYPE_MP_SYNC_REQ_1
242 IPC_MSGTYPE_MP_SYNC_REQ_5
243 Snmp mp sync message
246 IPC_MSGTYPE_MP_SYNC_REQ_9
278 IPC_MSGTYPE_MP_SYNC_ACK
279 BGP STATUS FROM MP
280 OSPF STATE FROM MP
284 10g wan phy alarm stat
287 BFD
288 BFD
289 TM logs
```

ipc show statistics

Syntax: **ipc show statistics**

This command displays IPC statistics, as shown in the following example.

```
Brocade# ipc show statistics
----------- CPU and Reliable IPC Status (Slot/Cpu) ----------
 9/0 ALIVE   Expected rx seq:  914  Next tx seq:  507  In tx seq:    0
----------------------- Message count -----------------------
 UnrelRxPkt=494902   RelRxPkt=20371    UnrelTxPkt=394404   RelTxPkt=1531
```

```
 UnrelRxMsg=767517   RelRxMsg=20371    UnrelTxMsg=394405   RelTxMsg=3205

----MsgType:Count (* instead of : means unregistered type)---
 Reliable RX   :   16:1         17:17       182:2871     201:1
                  202:1251     217:1       236:16227    287:1
 Unreliable RX :    1:81261      2:1          3:1         13:1
                   14:1         15:2        16:1         17:1
                   33:1         72:1        76:2         78:2
                   80:2        113:3       115:3741     116:271709
                  198:813      199:405828  215:87       219:4058
                  289:1
  Reliable TX   :   24:1         28:9         29:1         98:1
                  100:1        103:2       107:1        108:4
                  109:8        110:3027    112:3        114:4
                  117:1        118:1       122:1        124:1
                  125:2        126:1       129:5        134:1
                  144:1        147:5       155:1        162:1
                  163:1        164:2       165:1        167:1
                  170:1        171:17      173:1        177:1
                  178:13       179:1       183:1        185:1
                  186:30       187:8       189:1        194:11
                  201:7        203:1       205:1        227:2
                  229:1        286:18      290:1

Unreliable TX :    1:90757      2:1        21:1         22:10
                   30:2         34:1        71:1         75:2
                   77:2         79:2       110:2        195:2773
                  196:133952   198:162291  199:132      215:63
                  216:46       218:4058    243:309

-------------------- IPC Debug counters ---------------------
 BufAllocFail  =0        BufFreeFail   =0        BadChecksum   =0
 TxTooBigPkt   =0        TxBadMsgType  =0        ItcSendFail   =0
 RxTooBigPkt   =0        RxBadMsgType  =0        CardDownEvent =0
 TxBadContent  =0        TxBadFid      =0        TxEmptyFidMask=0
 TxRelSemLock  =0        TxUnrelSemLock=0        TxRelQFullErr =0
 DelayTxRelErr =0        DelayTimerErr =0        DelayFlushErr =0
 DelyTxRelQFul =0        RxBadFid      =0        RxNoCallback  =0
 RxBadAckAddr  =0        TxUnrelErr    =0        TxRelErr      =0
 SendBufError  =0        BufAlreadyFree=0        Retransmits   =0
 YieldSendBuf   =0        YieldGetBuf   =0        RxBadContent  =0

 QCountMinusErr=0        RxQIdxErr     =0        LastBadRxQIdx =0
 LastRelRxMsgFr=9        LastRelRxQidx =913      LastRelRxBlkAd=15990396
 RelRxQFull    =0        LastRelRxQFul =0        RelRxOutOfWin =1
 RelRxAlrdyVald=0        RelRxFrags    =0        LastAckTxSlot =9
 LastAckTxSeq  =914      LastAckRxSeq  =507      LastAckRxSlot =9
 RxAckNotInWin =0        RxAckAlrdyAckd=0        RxAckInvdBlk  =0
 RxAckNotInRang=0        RetransBadQIdx=0        LastBadTxQIdx =0
 LastBadTxQIdxQ=0        LastBadTxQidxW=0        LastBadTxQIdxA=0
 CurrentRetranQ=0        FirstRetranWin=0        FirstRetranBlk=0
 FirstRetranBuf=0        RxAckForNActQ =0
```

IPC debug commands

This section describes the debug command that generates information about IPC.

debug ip ipc

Syntax: [no] debug ip ipc

This command generates information about interprocess communication (IPC) activity in IP modules. The output generally includes a record of IPC messages sent and received, and any errors or warnings related to IP IPC. Several examples follow.

The following example indicates that the system is unable to obtain enough buffer space to send IPC messages to line cards.

```
Brocade# debug ip ipc
error - ip_ipc_icmp_config_info. system out of buffer
error - ip_ipc_ve_mac_addr system out of buffer
error - ip_ipc_vport_mask_info system out of buffer
error - ip_ipc_clear_cache_msg system out of buffer
error - ip_ipc_config_info. system out of buffer
error - ip_ipc_mcast_info. system out of buffer
```

The following message indicates that the routing table manager (RTM) is sending initialize data to line card 1, CPU 1.

```
RTM: ipc sync init data to (1,1), max routes 100
```

In the following example, Layer 3 port configuration information (state, MTU, redirect, encap, and so on) is being set for the specified port.

```
IP/IPC: set port data, port 1/10, type:2 value 1
```

In the following example, an IP address for a given port (1/10) is being sent to the line cards.

```
IP/IPC: set port address, port 1/10, add1, addr 10.10.10.1
```

In the following example, forwarding information is being sent to the line cards.

```
IP/IPC: send port table, FID_ID 10
```

In the following example, tunnel forwarding information is being sent to the line cards.

```
IP/IPC: send tunnel table, FID_ID 10
IP/IPC: send tunnel config, size 56
```

In the following example, DHCP configuration information is being sent to the line cards.

```
IP/IPC: send port dhcp index, FID_ID 10
```

In the following example, a DHCP list with 200 entries is being sent to the line cards.

```
IP/IPC: send dhcp list, size 200
```

Common diagnostic scenarios

If the error counts are continuously incrementing at a fast pace, or if the “In tx seq:” value continues to increase, contact Brocade Technical Support.

ITC diagnostics

An inter-task communication (ITC) is a data transfer between processes in two different tasks. ITC methods include message passing, synchronization, shared memory, and RPCs.

ITC show commands

This section describes the show commands that display ITC information.

itc show error list

Syntax: **itc show error list**

This command displays error information about the ITC messages. Command output resembles the following example.

```
Brocade# itc show error list
------------- first 32 itc errors ----------------------------
dest app id = 0x00000021 : src app id = 0x0000000b : msg type = 0x00130011 : error
= ITC_ERR_DEST_QUEUE_FULL
dest app id = 0x00000021 : src app id = 0x0000000b : msg type = 0x00130011 : error
= ITC_ERR_DEST_QUEUE_FULL
------------- itc errors per msgtype -----------------------------
  msgtype                 req failed       resp failed
  0x0130011             -         2              -         0
```

show emac

Syntax: **show emac** [*IF num* | **detail** | **name** | **phy_reg**]

- *IF num* - Displays information about the Ethernet MAC Management Processor (MP) packet by interface number. *IF num* variable can take values from 1 through 3.
- **detail** - Displays detailed information about the Ethernet MAC MP packet.
- **name** - Displays information about the Ethernet MAC MP packet by name.
- **phy_reg** - Displays dump information about the ten physical registers.

Command output resembles the following example.

```
Brocade# show emac detail
 intf : drx[0]    drx[1]    dtx       mrx       mtx       mbad
    1 : 477416    0         13928     1364593   13928     0
    2 : 1101925   35540     5592842   1137465   5592842   0
    3 : 0         0         0         0         0         0
  TX        P0/Pri0    P0/Pri1    P1/Pri0    P1/Pri1   :
   Depth   : 0          0          0          0
   Hi Wm   : 1          0          85         1
   QDrops  : 0          0          0          0
   QPkts   : 13928      0          1037316    4557264
  TX QPkts:MpPktType :
   Port0/Pri0:   MP_PKT_IP:1182       MP_PKT_ARP:8703
   Port1/Pri0:   1MP_PKT_IP:1038560
   Port1/Pri1:   MP_PKT_IP:4557269

  RX       P0        P1        P2         RX      P0          P1       P2
   Depth : 0         0         0          Hi Wm : 19         84        20
   QDrops: 0         0         0          QPkts : 809485    191978    613629
  RX QPkts:MpPktType :
   Port0/Pri0:   MP_PKT_IP:808741     MP_PKT_ARP:649       MP_PKT_RARP:21
MP_PKT_CDP:16         MP_PKT_FDP:11
              MP_PKT_IPC:10       MP_PKT_ISIS_HELLO:9        MP_PKT_LA:8
MP_PKT_STP:5          MP_PKT_GARP:3
               MP_PKT_SU_STP:3        MP_PKT_PVST:3         MP_PKT_MRP:2
MP_PKT_VSRP:2          MP_PKT_L2MCAST:2
                 MP_PKT_L2MCAST6:1
```

```
   Port0/Pri1:   MP_PKT_IP:190284     MP_PKT_ARP:1658       MP_PKT_RARP:37
   Port0/Pri2:   MP_PKT_IP:613291     MP_PKT_ARP:407        MP_PKT_RARP:42
MP_PKT_CDP:13         MP_PKT_FDP:4    MP_PKT_IPC:1
```

show mq

Syntax: **show mq**

This command displays information such as maximum depth, maximum messages and the number of failed messages.

NOTE
This command is available only on MP OS mode. Brocade recommends TAC guidance to execute OS mode commands.

The **show mq** command output resembles the following example.

```
MP-1 OS> show mq
Name    PRI  Length  Depth   RPtr    WPtr    MDepth  sMDepth  NMsgs   MMsgs   sMMsgs  Failed  sFailed
<…>
itc      0    4096    0       000ab0  000ab0  112     112      0       1       1       0       0
         1    4096    0       000000  000000  0       0        0       0       0       0       0
<…>
scp      0    102400  0       004430  004430  3488    3488     0       5       5       0       0
         1    102400  0       015078  015078  17440   17440    0       482     482     0       0
console 0    16384   0       003f44  003f44  372     372      0       13      13      0       0
         1    16384   0       001810  001810  36      36       0       1       1       0       0
vlan     0    102400  0       014954  014954  2640    2640     0       66      66      0       0
         1    102400  0       0090c0  0090c0  1840    1840     0       2       2       0       0
mac_mgr 0    204800  0       0045a8  0045a8  40      191332   0       1       1506    0       0
         1    204800  0       004bb0  004bb0  36      735996   0       1       466     0       0
```

Switch fabric modules

Switch fabric modules switch user packets from one interface module in a chassis to another. Switch fabric modules are hot-swappable.

Switch fabric fault monitoring

The Switch Fabric Fault Monitoring feature allows to display information about the current status of links between the switch fabric modules (SFMs) and interface modules and sends log messages to the console regarding the “UP” or “DOWN” status of the switch fabric modules.

Switch fabric show commands

This section describes the show commands that display information about switch fabrics.

show sfm-links

Syntax: **show sfm-links** [*sfm-number* | **all**]

- *sfm-number* - Specifies an SFM for which you want to display link information.
- **all** - Displays link information for all SFMs in the chassis.

NOTE
If the number of non-operational links fall below the minimum threshold, the following warning message is displayed:

```
WARN: LP 3 has 8 links up, less than minimum to guarantee line rate traffic
forwarding
```

This warning is displayed to inform users that the line rate traffic will not be maintained.

The **show sfm-links all** command displays information about the current status of links between the SFMs and interface modules in the Brocade NetIron XMR and Brocade MLX series devices. Each line in the output represents a link between an SFM and an interface module.

```
Brocade# show sfm-links all
SFM#/FE#   | FE link# | LP#/TM#  | TM link#    | link state
----------+-----------+---------+-----------+---------
2 / 1      | 32       | 3 / 1    | 13          | UP
2 / 1      | 31       | 3 / 2    | 01          | UP
2 / 1      | 11       | 3 / 1    | 01          | UP
2 / 1      | 12       | 3 / 2    | 13          | UP
2 / 3      | 32       | 3 / 1    | 19          | UP
2 / 3      | 31       | 3 / 2    | 07          | UP
2 / 3      | 11       | 3 / 1    | 07          | UP
2 / 3      | 12       | 3 / 2    | 19          | UP
3 / 1      | 32       | 3 / 1    | 16          | UP
3 / 1      | 31       | 3 / 2    | 04          | UP
3 / 3      | 12       | 3 / 2    | 22          | UP
WARN: LP 3 has 8 links up, less than minimum to guarantee line rate traffic
forwarding
```

show version

Syntax: show version

This command displays information about the SFMs installed in your router chassis, as shown in the following example.

```
Brocade# show version
HW: NetIron XMR Router
Backplane (Serial #: Not Exist,  Part #: Not Exist)
NI-X-SF Switch Fabric Module 1 (Serial #: PR29050242,  Part #: 31523-100A)
FE 1: Type 00000,  Version 0
FE 3: Type 00000,  Version 0
NI-X-SF Switch Fabric Module 2 (Serial #: PR29050246,  Part #: 31523-100A)
FE 1: Type 00000,  Version 0
FE 3: Type 00000,  Version 0
NI-X-SF Switch Fabric Module 3 (Serial #: PR30050270,  Part #: 31523-100A)
FE 1: Type 00000,  Version 0
FE 3: Type 00000,  Version 0
```

show sfm-utilization

Syntax: show sfm-utilization [*sfm-number* | **all**]

- *sfm-number* - Specifies the SFM for which to display the utilization information.
- **all** - Displays utilization information for all SFMs in the chassis.

The **show sfm-utilization all** command displays bandwidth usage on all SFMs on the device, as shown in the following example.

```
Brocade# show sfm-utilization all
SFM#2
----------+----------+---------+----------+---------
last 1 second utilization = 0.4%
last 5 seconds utilization = 0.3%
last 1 minute utilization = 0.1%
last 5 minutes utilization = 0.0%
SFM#3
----------+----------+---------+----------+---------
last 1 second utilization = 0.4%
last 5 seconds utilization = 0.4%
last 1 minute utilization = 0.1%
last 5 minutes utilization = 0.0%
last 5 minutes utilization = 0.0%
```

To display bandwidth usage for a single SFM, enter the **show sfm-utilization** *sfm-number* command.

```
Brocade# show sfm-utilization 2
SFM#2
----------+----------+---------+----------+---------
last 1 second utilization = 0.4%
last 5 seconds utilization = 0.3%
last 1 minute utilization = 0.1%
last 5 minutes utilization = 0.0%
```

Switch fabric debug commands

There are no switch fabric-specific debug commands.

Common diagnostic scenarios

The following common scenarios may occur with switch fabric modules:

- The switch fabric module is not receiving power.
- If the switch fabric module is inactive (not switching packets), the links are down. Use the **show sfm-links all** command to identify which links are down.

Switch fabric module not receiving power

The switch fabric module front panel includes two LEDs, one labeled Pwr (power) and one labeled Active.

- If the Pwr LED is off for an extended period of time, the switch fabric module is not receiving power. Check the power connection and the power supply. Make sure the module is seated properly in the backplane.
- If the Pwr LED is on, but the Active LED is off, the module is not in active mode and cannot switch packets. In this case, refer to "Switch fabric module unable to switch packets".

Switch fabric module unable to switch packets

If the switch fabric module is receiving power, but is still unable to switch packets, disable it and then re-enable it to determine if the problem re-occurs. You may need to perform these steps on each switch fabric module in your device to determine which one is faulty. If the problem continues, contact Brocade Technical Support for further assistance.

Hot-swapping a switch fabric module

You can remove and install switch fabric modules while the Brocade NetIron XMR and Brocade MLX series device is powered on and running. For more information, refer to the *Brocade MLX Series and Brocade NetIron XMR Installation Guide*.

Power supplies, fans, and temperature

Information about power supplies, fans, and temperature readings are sent to the static buffer log. New messages replace older ones, so only the most recent message is displayed. For example, only the most recent temperature warning message is shown in the log. The static buffer is not configurable, and static buffer messages do not appear in dynamic buffer logs.

NOTE
Always cover empty chassis slots with the slot panels that shipped with your device. Operating the device with exposed empty slots can cause the system to overheat.

Power supply, fan, and temperature show commands

This section describes the show commands that display power supply, fan, and temperature information.

show log

Syntax: show log

You can view power supply, fan, and temperature information using the **show log** command. Command output resembles the following example.

```
Brocade# show log
Syslog logging: enabled (0 messages dropped, 0 flushes, 0 overruns)
    Buffer logging: level ACDMEINW, 82 messages logged
    level code: A=alert C=critical D=debugging M=emergency E=error
                I=informational N=notification W=warning

Static Log Buffer:
Jun  4 09:43:01:A:System: AC Power Supply 4 , 1st from left, Installed (O
Jun  4 09:43:01:A:System: AC Power Supply 4 , 2nd from left, Installed (O

Dynamic Log Buffer (1000 lines):
Jun  4 17:25:30:I:Security: telnet login by debra from src IP 10.55.1.103 to PRI
VILEGED EXEC mode
Jun  4 17:20:54:I:Security: ssh login from src IP 10.47.6.8 to USER EXEC mode
Jun  4 17:20:34:I:Security: ssh logout from src IP 10.47.6.8 from USER EXEC mode
Jun  4 17:20:31:I:Security: ssh login from src IP 10.47.6.8 to USER EXEC mode
Jun  4 14:20:42:I:Security: telnet logout by debra from src IP 10.55.1.103 from
USER EXEC mode
Jun  4 14:05:06:I:Security: telnet login by debra from src IP 10.55.1.103 to PRI
VILEGED EXEC mode
Jun  4 12:29:14:W:    Latched low RX Power warning, port 4/1
Jun  4 12:29:14:A:    Latched low RX Power alarm, port 4/1
Jun  4 12:24:15:I:System: Interface ethernet 4/1, state up
Jun  4 12:24:14:W:    Latched low RX Power warning, port 4/1
```

```
Jun  4 12:24:14:A:    Latched low RX Power alarm, port 4/1
Jun  4 12:24:05:I:System: Interface ethernet 4/1, state down - link down
Jun  4 12:24:02:I:System: Interface ethernet 4/1, state up
Jun  4 12:23:54:I:System: Interface ethernet 4/1, state down - link down
```

show fan-threshold

Syntax: show fan-threshold

This command displays the current settings of temperature thresholds and fan speeds, as shown in the following example.

```
Brocade# show fan-threshold
=== Thermal Sensor Control Block (THERMAL_SENSOR_TEST_RULE_MP) ===
Fan Speed Low: -1 - 60
Fan Speed Med: 57 - 70
Fan Speed Med-Hi: 67 - 80
Fan Speed Hi: 77 - 85
state = 0 (FAN_STATE_LOW)
max_ts_shut_off_count = 3
shut_off_count = 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
=== Thermal Sensor Control Block (THERMAL_SENSOR_TEST_RULE_SNM) ===
Fan Speed Low: -1 - 30
Fan Speed Med: 27 - 40
Fan Speed Med-Hi: 37 - 50
Fan Speed Hi: 47 - 75
state = 1 (FAN_STATE_MED)
max_ts_shut_off_count = 3
shut_off_count = 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
=== Thermal Sensor Control Block (THERMAL_SENSOR_TEST_RULE_LP) ===
Fan Speed Low: -1 - 50
Fan Speed Med: 46 - 55
Fan Speed Med-Hi: 51 - 60
Fan Speed Hi: 56 - 95
state = 0 (FAN_STATE_LOW)
max_ts_shut_off_count = 3
shut_off_count = 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
=== Thermal Sensor Control Block (THERMAL_SENSOR_TEST_RULE_LP_XPP) ===
Fan Speed Low: -1 - 50
Fan Speed Med: 45 - 65
Fan Speed Med-Hi: 60 - 75
Fan Speed Hi: 70 - 113
state = 1 (FAN_STATE_MED)
max_ts_shut_off_count = 3
shut_off_count = 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
=== Thermal Sensor Control Block (THERMAL_SENSOR_TEST_RULE_STANDBY_MP) ===
Fan Speed Low: -1 - 60
Fan Speed Med: 57 - 70
Fan Speed Med-Hi: 67 - 80
Fan Speed Hi: 77 - 85
state = 0 (FAN_STATE_LOW)
max_ts_shut_off_count = 3
shut_off_count = 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
=== Thermal Sensor Control Block (THERMAL_SENSOR_TEST_RULE_MP_CPU) ===
Fan Speed Low: -1 - 60
Fan Speed Med: 57 - 70
Fan Speed Med-Hi: 67 - 80
Fan Speed Hi: 77 - 95
state = 0 (FAN_STATE_LOW)
max_ts_shut_off_count = 3
```

```
shut_off_count = 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
=== Thermal Sensor Control Block (THERMAL_SENSOR_TEST_RULE_STANDBY_MP_CPU) ===
Fan Speed Low: -1 - 60
Fan Speed Med: 57 - 70
Fan Speed Med-Hi: 67 - 80
Fan Speed Hi: 77 - 95
state = 0 (FAN_STATE_LOW)
```

show chassis

Syntax: show chassis

The **show chassis** command displays power supply and fan information, as well as temperature readings for the chassis.

```
Brocade# show chassis
*** NetIron XMR 8000 CHASSIS ***

---POWERS ---
Power 1: Installed (Failed or Disconnected)
Power 2: Installed (Failed or Disconnected)
Power 3 (30351200 - AC 1200W): Installed (OK)
Power 4 (30351200 - AC 1200W): Installed (OK)
Total power budget for chassis = 2400W
Total power budget for LPs      = 2049W
Slot Power-On Priority and Power Usage:
Slot2 pri=1 module type=NI-X-OC48x4 4-port OC48/12 STM16/STM4 Module power usage
=132W
Slot3 pri=1 module type=NI-XMR-1Gx20-GC 20-port 10/100/1000 Copper Module power
usage=156W
Slot4 pri=1 module type=NI-XMR-10Gx2 2-port 10GbE Module power usage=165W

--- FANS ---
Right fan tray (fan 1): Status = OK, Speed = MED (75%)
Right fan tray (fan 2): Status = OK, Speed = MED (75%)
Right fan tray (fan 3): Status = OK, Speed = MED (75%)
Right fan tray (fan 4): Status = OK, Speed = MED (75%)

--- TEMPERATURE READINGS ---
Active Mgmt Module: 30.250C 44.125C
SNM1: 25.5C
SNM2: 23.0C
SNM3: 25.5C
LP2 Sensor1: 35.0C
LP2 Sensor2: 50.375C
LP3 Sensor1: 30.5C
LP3 Sensor2: 38.500C
LP4 Sensor1: 35.0C
LP4 Sensor2: 43.875C
LP4 Sensor3: UNUSED
Temperature Monitoring Poll Period is 60 seconds

--- MISC INFO ---
Backplane EEPROM MAC Address:  0000.00e2.ca00
```

show temperature

Syntax: show temperature

The **show temperature** command displays temperature readings for each interface module. The temperature is polled every 60 seconds.

```
Brocade# show temperature
SLOT #:       CARD TYPE:     SENSOR #    TEMPERATURE (C):
     17        ACTIVE MG        1            27.500C
     17        ACTIVE MG        2            48.125C
=======================================================
     1           LP            1            33.0C
     1           LP            2            47.625C
     3           LP            1            35.0C
     3           LP            2            60.250C
     5           LP            1            41.5C
     5           LP            2            57.0C
     5           LP            3            UNUSED
     5           LP            4            41.0C
     5           LP            5            57.250C
     5           LP            6            UNUSED

SNM #:      TEMPERATURE (C):
    1             28.0C
    2             23.5C
    3             31.5C
    4             22.5C
```

Common diagnostic scenarios

- The power supply is not providing power. Check all power connections, and replace a faulty power supply if necessary. For more information, refer to the *Brocade MLX Series and Brocade NetIron XMR Installation Guide*.
- The fans are not receiving power. Check all power connections, and replace a faulty power supply if necessary. For more information, refer to the *Brocade MLX Series and Brocade NetIron XMR Installation Guide*.
- The temperature is outside the normal operating range. Refer to “Temperature outside normal operating range”.

Temperature outside normal operating range

If the device detects temperatures outside the normal range, it will automatically perform one of the following functions depending on the severity of the reading:

- Leave the fan speed as is.
- Increase the fan speed.
- Decrease the fan speed.
- Shut down a module to prevent damage after the first warning.
- Generate a syslog message and an SNMP trap.

If none of these measures resolves the problem, you must perform the following steps.

1. Shut down the device immediately.
2. Inspect all fans for damage or failure.
3. Inspect electrical connections to the fans.

4. Replace any component that has been damaged by excessive temperature.

The following are the normal operating temperature, humidity, and altitude specifications for Brocade NetIron XMR and Brocade MLX series routers:

- Operating temperature: 0° to 40°C (32° to 104°F)
- Relative humidity: 5 to 90%, at 40°C (104°F), noncondensing
- Operating altitude: 0 to 3048 meters (0 to 10,000 ft)

Fiber optic modules

The most common problems with fiber optic modules are caused by dirty connectors. Optical cables that are contaminated in any way (with dust, hand oil, and so on) can degrade the optic eye pattern. Some of the following symptoms may be experienced:

- Port appears to not function (either no link or unstable link)
- Cyclic redundancy check (CRC) errors
- Port flapping
- Packet loss

Before inserting the fiber cable into the fiber optic transceiver, ensure that it is free of dust by cleaning the end. A “Fiber Swiper” cleaner is provided by Brocade for this purpose with each optic shipment (reference instructions provided with the Fiber Swiper).

It is very important that the end of an optical cable is clean when using any data rate. However, it is *critical* that cable ends are clean when using 10 Gigabit data rates. This must be the first step in troubleshooting symptoms such as those stated above. Always ensure that the optical cables are cleaned.

NOTE
When not using a fiber optic module port connector, replace the protective cover to prevent dust or dirt from contaminating the connector.

Fiber optic show commands

This section describes the show command that displays optics information.

show version

Syntax: show version

This command displays information about optic modules installed in the Brocade NetIron XMR and Brocade MLX series routers. Optics information resembles the output segment in the following example.

```
Brocade# show version
System Mode: MLX
Chassis: NetIron 8-slot (Serial #:       GOLD,  Part #: 35549-000C)
NI-X-SF  Switch Fabric Module 1 (Serial #: PR23050271,  Part #: 31523-100A)
FE 1: Type fe200,  Version 2
FE 3: Type fe200,  Version 2
NI-X-SF  Switch Fabric Module 2 (Serial #: SA21091164,  Part #: 35523-302A)
FE 1: Type fe200,  Version 2
```

```
FE 3: Type fe200,  Version 2
NI-X-SF  Switch Fabric Module 3 (Serial #: SA21091204,  Part #: 35523-302A)
FE 1: Type fe200,  Version 2
FE 3: Type fe200,  Version 2
==========================================================================
SL M2: NI-MLX-MR Management Module Active (Serial #: SA21091472, Part #:
35524-103C):
Boot     : Version 5.1.0T165 Copyright (c) 1996-2009 Brocade Communications
Systems, Inc.
Compiled on Aug 11 2010 at 14:06:58 labeled as xmprm05100
 (524038 bytes) from boot flash
Monitor  : Version 5.1.0T165 Copyright (c) 1996-2009 Brocade Communications
Systems, Inc.
Compiled on Aug 11 2010 at 14:06:30 labeled as xmb05100
 (524053 bytes) from code flash
IronWare : Version 5.1.0T163 Copyright (c) 1996-2009 Brocade Communications
Systems, Inc.
Compiled on Aug 28 2010 at 04:23:16 labeled as xmr05100b312
 (6985918 bytes) from Primary
Board ID : 00 MBRIDGE Revision : 32
916 MHz Power PC processor 7447A (version 8003/0101) 166 MHz bus
512 KB Boot Flash (AM29LV040B), 32 MB Code Flash (MT28F128J3)
1024 MB DRAM
Active Management uptime is 6 minutes 21 seconds
==========================================================================
SL M1: NI-MLX-MR Management Module Standby (Serial #: SA21091421, Part #:
35524-103C):
Boot     : Version 5.1.0T165 Copyright (c) 1996-2009 Brocade Communications
Systems, Inc.
Compiled on Aug 11 2010 at 14:06:58 labeled as xmprm05100
 (524038 bytes) from boot flash
Monitor  : Version 5.1.0T165 Copyright (c) 1996-2009 Brocade Communications
Systems, Inc.
Compiled on Aug 11 2010 at 14:06:30 labeled as xmb05100
 (524053 bytes) from code flash
IronWare : Version 5.1.0T163 Copyright (c) 1996-2009 Brocade Communications
Systems, Inc.
Compiled on Aug 28 2010 at 04:23:16 labeled as xmr05100b312
 (6985918 bytes) from Primary
Board ID : 00 MBRIDGE Revision : 32
916 MHz Power PC processor 7447A (version 8003/0101) 166 MHz bus
512 KB Boot Flash (AM29LV040B), 32 MB Code Flash (MT28F128J3)
1024 MB DRAM
Standby Management uptime is 5 minutes 33 seconds
==========================================================================
==========================================================================
SL 1: BR-MLX-1GCx24-X 24-port 10/100/1000Base-T Copper Module (Serial #:   Not
Exist, Part #: Not Exist)
License: MLX (LID: ÿÿÿÿÿÿÿÿÿÿÿ)
Boot     : Version 5.1.0T175 Copyright (c) 1996-2009 Brocade Communications
Systems, Inc.
Compiled on Aug 11 2010 at 14:07:20 labeled as xmlprm05100
 (492544 bytes) from boot flash
Monitor  : Version 5.1.0T175 Copyright (c) 1996-2009 Brocade Communications
Systems, Inc.
Compiled on Aug 11 2010 at 14:07:42 labeled as xmlb05100
 (493244 bytes) from code flash
IronWare : Version 5.1.0T177 Copyright (c) 1996-2009 Brocade Communications
Systems, Inc.
Compiled on Aug 28 2010 at 04:28:44 labeled as xmlp05100b312
```

```
 (4949977 bytes) from Primary
FPGA versions:
Valid PBIF Version = 3.24, Build Time = 8/4/2010 14:57:00

Valid XPP Version = 6.03, Build Time = 2/18/2010 16:38:00

Valid STATS Version = 0.08, Build Time = 2/18/2010 16:30:00

BCM56512GMAC 0
BCM56512GMAC 1
666 MHz MPC 8541 (version 8020/0020) 333 MHz bus
512 KB Boot Flash (MX29LV040C), 16 MB Code Flash (MT28F128J3)
1024 MB DRAM, 8 KB SRAM, 286331153 Bytes BRAM
PPCR0: 1024K entries CAM, 16384K PRAM, 2048K AGE RAM
LP Slot 1 uptime is 5 minutes 36 seconds
==========================================================================
SL 3: BR-MLX-10Gx4-X 4-port 10GbE Module (Serial #: BMY0319F00J, Part #:
60-1001875-07)
License: MLX (LID: doaFIGOhFFl)
Boot     : Version 5.1.0T175 Copyright (c) 1996-2009 Brocade Communications
Systems, Inc.
Compiled on Aug 11 2010 at 14:07:20 labeled as xmlprm05100
 (492544 bytes) from boot flash
Monitor  : Version 5.1.0T175 Copyright (c) 1996-2009 Brocade Communications
Systems, Inc.
Compiled on Aug 11 2010 at 14:07:42 labeled as xmlb05100
 (493244 bytes) from code flash
IronWare : Version 5.1.0T177 Copyright (c) 1996-2009 Brocade Communications
Systems, Inc.
Compiled on Aug 28 2010 at 04:28:44 labeled as xmlp05100b312
 (4949977 bytes) from Primary
FPGA versions:
Valid PBIF Version = 3.22, Build Time = 2/5/2010 14:43:00

Valid XPP Version = 6.04, Build Time = 2/3/2010 14:39:00

Valid XGMAC Version = 0.13, Build Time = 2/3/2010 14:42:00

X10G2MAC 0
X10G2MAC 1
666 MHz MPC 8541 (version 8020/0020) 333 MHz bus
512 KB Boot Flash (MX29LV040C), 16 MB Code Flash (MT28F128J3)
1024 MB DRAM, 8 KB SRAM, 286331153 Bytes BRAM
PPCR0: 1024K entries CAM, 16384K PRAM, 2048K AGE RAM
PPCR1: 1024K entries CAM, 16384K PRAM, 2048K AGE RAM
LP Slot 3 uptime is 5 minutes 38 seconds
==========================================================================
SL 4: NI-MLX-1Gx48-T 48-port 10/100/1000Base-T MRJ21 Module (Serial #: SA05091472,
Part #: 35663-20EA)
Boot     : Version 5.1.0T175 Copyright (c) 1996-2009 Brocade Communications
Systems, Inc.
Compiled on Aug 11 2010 at 14:07:20 labeled as xmlprm05100
 (492544 bytes) from boot flash
Monitor  : Version 5.1.0T175 Copyright (c) 1996-2009 Brocade Communications
Systems, Inc.
Compiled on Aug 11 2010 at 14:07:42 labeled as xmlb05100
 (493244 bytes) from code flash
IronWare : Version 5.1.0T177 Copyright (c) 1996-2009 Brocade Communications
Systems, Inc.
Compiled on Aug 28 2010 at 04:28:44 labeled as xmlp05100b312
```

```
 (4949977 bytes) from Primary
FPGA versions:
Valid PBIF Version = 3.24, Build Time = 8/4/2010 14:57:00

Valid XPP Version = 6.03, Build Time = 2/18/2010 16:38:00

Valid STATS Version = 0.08, Build Time = 2/18/2010 16:30:00

BCM56502GMAC 0
BCM56502GMAC 1
666 MHz MPC 8541 (version 8020/0020) 333 MHz bus
512 KB Boot Flash (AM29LV040B), 16 MB Code Flash (MT28F640J3)
1024 MB DRAM, 8 KB SRAM, 0 Bytes BRAM
PPCR0: 768K entries CAM, 8192K PRAM, 2048K AGE RAM
PPCR1: 768K entries CAM, 8192K PRAM, 2048K AGE RAM
LP Slot 4 uptime is 5 minutes 42 seconds
==========================================================================
SL 6: NI-MLX-10Gx4 4-port 10GbE Module (Serial #: SA12090950, Part #: 35600-202D)
Boot     : Version 5.1.0T175 Copyright (c) 1996-2009 Brocade Communications
Systems, Inc.
Compiled on Aug 11 2010 at 14:07:20 labeled as xmlprm05100
 (492544 bytes) from boot flash
Monitor  : Version 5.1.0T175 Copyright (c) 1996-2009 Brocade Communications
Systems, Inc.
Compiled on Aug 11 2010 at 14:07:42 labeled as xmlb05100
 (493244 bytes) from code flash
IronWare : Version 5.1.0T177 Copyright (c) 1996-2009 Brocade Communications
Systems, Inc.
Compiled on Aug 28 2010 at 04:28:44 labeled as xmlp05100b312
 (4949977 bytes) from Primary
FPGA versions:
Valid PBIF Version = 3.22, Build Time = 2/5/2010 14:43:00

Valid XPP Version = 6.04, Build Time = 2/3/2010 14:39:00

Valid XGMAC Version = 0.13, Build Time = 2/3/2010 14:42:00

X10G2MAC 0
X10G2MAC 1
666 MHz MPC 8541 (version 8020/0020) 333 MHz bus
512 KB Boot Flash (AM29LV040B), 16 MB Code Flash (MT28F640J3)
512 MB DRAM, 8 KB SRAM, 286331153 Bytes BRAM
PPCR0: 768K entries CAM, 8192K PRAM, 2048K AGE RAM
PPCR1: 768K entries CAM, 8192K PRAM, 2048K AGE RAM
LP Slot 6 uptime is 5 minutes 44 seconds
==========================================================================
All show version done
```

Fiber optic debug commands

There are no fiber optic-specific debug commands.

Testing network connectivity

You can test connectivity to other network devices by pinging those devices. You also can trace routes.

Pinging an IP address

To verify that a Brocade NetIron XMR and Brocade MLX series router can reach another device through the network, enter the **ping** command at any level of the CLI.

ping

Syntax: **ping** [*ip addr* | *hostname* | **vrf** *instance-name*] [**source** *ip addr*] [**count** *num*] [**timeout** *msec*] [**ttl** *num*] [**size** *byte*] [**quiet**] [**numeric**] [**no-fragment**] [**verify**] [**data** *1-to-4 byte hex*] [**brief**]

- *ip addr* - Specifies the IP address of the device.
- *hostname* - Specifies the host name.
- **vrf** *instance-name* - Specifies a VPN routing or forwarding instance as the origin of the ping packets.
- **source** *ip addr* - Specifies an IP address to be used as the origin of the ping packets.
- **count** *num* - Specifies how many ping packets the device sends.
- **timeout** *msec* - Specifies how many milliseconds the device waits for a reply from the pinged device.
- **ttl** *num* - Specifies the maximum number of hops.
- **size** *byte* - Specifies the size of the ICMP data portion of the packet.
- **quiet** - Hides informational messages such as a summary of the ping parameters sent to the device and displays only messages indicating the success or failure of the ping.
- **numeric** - Allows the display to list devices by IP address instead of by name.
- **no-fragment** - Turns on the "do not fragment" bit in the IP header of the ping packet.
- **verify** - Verifies that the data in the echo packet (the reply packet) is the same as the data in the echo request (the ping).
- **data** *1-to-4 byte hex* - Specifies a specific data pattern for the payload instead of the default data pattern.

NOTE
If you address the ping to the IP broadcast address, the device lists the first four responses to the ping.

Tracing a route

To determine the path through which the router can reach another network device, enter the **traceroute** command at any level of the CLI.

traceroute

Syntax: **traceroute** *destination_IPv4 address* [**maxttl** *value*] [**minttl** *value*] [**numeric**] [**timeout** *seconds*]

- *destination_IPv4 address* - Specifies the IPv4 address of the remote device.
- **maxttl** *value* - Specifies the maximum Time-to-Live (TTL) value. This value equals the maximum number of hops traversed by the **traceroute** command. Valid values are from 1 through 255. The default value is 1, incremented by 1 with each successive hop.

- **minttl** *value* - Specifies the minimum TTL value. This value equals the minimum number of hops traversed by the **traceroute** command. Valid values are from 1 through 255. The default value is 1, incremented by 1 with each successive hop.
- **numeric** - Allows the listing of devices by IP address rather than by name.
- **timeout** *seconds* - Specifies the time in seconds that traceroute can take to reach the next hop before the command times out. Valid values are from 1 through 120 seconds. The default time-out value is 2 seconds.

Each line of output represents a hop along the IP network path. For each packet sent, traceroute records the round-trip time (RTT) in milliseconds and the IP address of the router that returned the ICMP TTL-exceeded message. An asterisk (*) indicates that no information could be obtained for the specified hop or traceroute timed out.

```
Brocade# traceroute 10.157.22.199
Type Control-c to abort
Tracing the route to IP node (10.157.22.199) from 1 to 30 hops
   1     *       20 ms<1 ms 10.20.96.1
   2     4 ms   <1 ms   <1 ms 10.31.20.25
   3    <1 ms   <1 ms   <1 ms 10.16.200.121
   4    <1 ms   <1 ms   <1 ms 10.110.111.102
   5    <1 ms   <1 ms   <1 ms 10.49.131.1
   6    <1 ms   <1 ms   <1 ms 10.49.130.18
   7    <1 ms   <1 ms    1 ms 10.125.199.61
   8     1 ms    3 ms    2 ms 10.125.31.70
   9     3 ms    1 ms    1 ms 10.125.26.202
   10    1 ms    1 ms    1 ms 10.125.30.178
   11    3 ms    3 ms    3 ms 10.125.13.118
   12    3 ms    3 ms    3 ms 10.250.5.56
   13    *       *       *     ?
   14    *       *       *     ?
   15    *       *       *     ?
   16    *       *       *     ?
   17    *       *       *     ?
   18    *       *       *     ?
   19    *       *       *     ?
   20    *       *       *     ?
   21    *       *       *     ?
   22    *       *       *     ?
   23    *       *       *     ?
   24    *       *       *     ?
   25    *       *       *     ?
   26    *       *       *     ?
   27    *       *       *     ?
   28    *       *       *     ?
   29    *       *       *     ?
   30    *       *       *     ?
```

Chapter

Layer 1 Diagnostics 3

In this chapter

This chapter describes common Layer 1 diagnostic procedures for the Brocade NetIron XMR and Brocade MLX series routers. In general, Layer 1 issues are related to hardware, the most common being the following physical connectivity problems:

- Faulty ports
- Faulty cables
- Faulty hardware
- Input and output errors
- Cyclic redundancy check (CRC) errors
- Excessive or late collisions
- Overruns
- Output buffer failures

Ethernet diagnostics

The following sections describe how to troubleshoot Layer 1 issues for Ethernet interfaces.

Ethernet autonegotiation

10BASE-T, 100BASE-TX, and 1000BASE-T all use an RJ-45 connector, which creates the potential for connecting electrically-incompatible components to each other, and which can cause network disruption. The IEEE developed autonegotiation to eliminate the possibility of dissimilar technologies interfering with one other.

Autonegotiation uses electrical pulses generated by a device over a 10 Mbps, 100 Mbps, or 1000 Mbps link (when the link is not sending or receiving data) to detect the presence of a connection to another device. These unipolar, positive-only pulses have a duration of 100 ns, and are generated in trains of a maximum of 33 pulses. These pulse trains are referred to as fast link pulse (FLP) bursts. The time interval between the start of each burst is 16 ms, with a tolerance of 8 ms.

Autonegotiation and 10BASE-T

An FLP burst is not recognized as valid by a 10BASE-T device receiving it from an autonegotiation device. The 10BASE-T device will detect a failure of the link. To avoid this, when the autonegotiation device receives the 10BASE-T pulses, it automatically switches to 10BASE-T half-duplex mode. If the 10BASE-T device is operating in full-duplex mode, a duplex mismatch can occur.

Duplex mismatches

A duplex mismatch can occur between devices in the following situations:

- One device is manually set to half duplex and one device is manually set to full duplex.
- One device is set to autonegotiation and one device is manually set to full duplex.

Duplex mismatches are difficult to diagnose because the network still appears to be working. Simple tests, such as ping, report a valid connection even though network performance can be much slower than normal.

When one device operates in full duplex while the other one operates in half duplex, the connection works at a very low speed if both devices attempt to send frames at the same time. This is because a full-duplex device may transmit data while it is receiving, but if the other device is working in half duplex, it cannot receive data while it is sending. The half-duplex device senses a collision and attempts to resend the frame it was sending. Depending on timing, the half-duplex device may sense a late collision, which it will interpret as a hard error, and will not attempt to resend the frame. At the other end, the full-duplex device does not detect a collision and does not resend the frame, even if the half-duplex device has already discarded it as corrupted by collision.

The packet loss happens when both devices are transmitting at the same time, and may happen even when the link is used, from the user's perspective, in one direction only. A Transmission Control Protocol (TCP) stream requires that all packets sent be acknowledged by the receiving device, even if actual data is sent in one direction only. Packet collisions may occur with acknowledgement packets traveling in the other direction.

Because the full-duplex device does not expect incoming frames to be truncated by collision detection, the device reports Frame Check Sequence (FCS) errors. The combination of late collisions reported at the half-duplex end, and FCS errors reported by the full duplex end, can indicate a duplex mismatch.

Priority

When one device receives the technology abilities of the other device, both devices decide the best possible mode of operation supported (each device chooses the mode that is topmost on the following list). The priority among modes specified in the 2002 edition of IEEE 802.3 is as follows:

- 1000BASE-T full duplex
- 1000BASE-T half duplex
- 100BASE-T2 full duplex
- 100BASE-TX full duplex
- 100BASE-T2 half duplex
- 100BASE-T4
- 100BASE-TX half duplex
- 10BASE-T full duplex
- 10BASE-T half duplex

Currently, all network equipment manufacturers recommend using autonegotiation on all access ports. On rare occasions where autonegotiation may fail, you may still need to force settings.

Unidirectional Link Detection

Unidirectional Link Detection (UDLD) monitors a link between two Brocade devices and brings the ports on each side of the link down if the link fails at any point between the two devices. This feature is useful for links that are individual ports and for trunk links.

Ports with UDLD enabled 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.

NOTE
UDLD is supported only on Ethernet ports.

Configuring a UDLD holddown threshold

You can configure a UDLD holddown threshold to prevent network flapping that affects ring topologies. If a port fails after a specified number of times, UDLD brings the port down. A port is considered to have failed if it consistently loses UDLD packets during the specified duration. The port would then be considered as a *blocked* port and UDLD considers the port state as *down*. A syslog message is generated to indicate a UDLD holddown is in effect on that port. The port remains disabled until it is manually restored. To display UDLD holddown information, refer to"show link-keepalive ethernet" on page 62.

Ethernet show commands

This section describes the show commands that display information about Ethernet interfaces and UDLD.

show interface brief

Syntax: **show interface brief**

This command displays a summary of the information provided in the **show interface** command. Command output resembles the following example.

```
Brocade# show interface brief
Port    Link    Port-State Dupl    Speed   Trunk   Tag     Priori MAC Name          Type
1/1     Up      Forward    Full    100G    None    Yes     level0 0000.0004.0000    default-port
```

show interface ethernet

Syntax: **show interface ethernet** *slotnum/portnum*

This command displays information about a specific Ethernet interface, as shown in the following example.

```
Brocade# show interface ethernet 1/1
GigabitEthernet2/1 is disabled, line protocol is down, link keepalive is enabled
Hardware is GigabitEthernet, address is 0000.00e2.5900 (bia 0000.00e2.5900)
Configured speed 1Gbit, actual unknown, configured duplex fdx, actual unknown
```

```
Configured mdi mode AUTO, actual unknown
Member of 2 L2 VLANs, port is tagged, port state is Disabled
STP configured to ON, Priority is level7, flow control enabled
Force-DSCP disabled
mirror disabled, monitor disabled
Not member of any active trunks
Not member of any configured trunks
No port name
MTU 1522 bytes, encapsulation ethernet
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
0 packets output, 0 bytes, 0 underruns
Transmitted 0 broadcasts, 0 multicasts, 0 unicasts
0 output errors, 0 collisions, DMA transmitted 0 packets
```

show link-keepalive ethernet

Syntax: show link-keepalive ethernet *slot/port*

To display link keepalive information for Ethernet ports, enter the **show link-keepalive ethernet** command.

```
Brocade# show link-keepalive ethernet 4/1
Current State: up Remote MAC Addr: 0000.00d2.5100
Local Port: 4/1 Remote Port: 2/1
Local System ID: e0927400 Remote System ID: e0d25100
Packets sent: 254 Packets received: 255
Transitions: 1
```

show statistics

Syntax: show statistics ethernet *slot/port*

The *slot/port* variable specifies the port that you want to display statistics for.

To display statistical information about the traffic passing through a specified Ethernet port, enter the following command at any CLI level.

```
Brocade# show statistics ethernet 9/1
PORT 9/1 Counters:
             InOctets      210753498112          OutOctets     210753550720
                InPkts       1646511726            OutPkts       1646512119
      InBroadcastPkts                 0    OutBroadcastPkts                 0
      InMulticastPkts                  0    OutMulticastPkts                 0
        InUnicastPkts        1646511726      OutUnicastPkts       1646512142
           InDiscards                  0         OutDiscards                 0
             InErrors                  0           OutErrors                 0
          InCollisions                 0       OutCollisions                 0
                                          OutLateCollisions                 0
            Alignment                 0                 FCS                 0
        InFlowCtrlPkts                0     OutFlowCtrlPkts                 0
            GiantPkts                 0           ShortPkts                 0
         InBitsPerSec        3440829770       OutBitsPerSec        3440686411
         InPktsPerSec           3360185       OutPktsPerSec           3360085
        InUtilization            39.78%      OutUtilization            39.78%
```

Table 6 describes the output parameters of the **show statistics ethernet** command.

TABLE 6 Output parameters of the **show statistics ethernet** command

Field	Description
InOctets	The total number of good octets and bad octets received.
OutOctets	The total number of good octets and bad octets transmitted.
InPkts	The total number of packets received. The count includes rejected and local packets that are not transmitted to the switching core for transmission.
OutPkts	The number of good packets received. The count includes unicast, multicast, and broadcasts packets.
InBroadcastPkts	The total number of good broadcast packets received.
OutBroadcastPkts	The total number of good broadcast packets transmitted.
InMulticastPkts	The total number of good multicast packets received.
OutMulticastPkts	The total number of good multicast packets transmitted.
InUnicastPkts	The total number of good unicast packets received.
OutUnicastPkts	The total number of good unicast packets transmitted.
InDiscards	The total number of packets that were received and then dropped due to a lack of receive buffers.
OutDiscards	The total number of packets that were transmitted and then dropped due to a lack of transmit buffers.
InErrors	The total number of packets received that had Alignment errors or phy errors.
OutErrors	The total number of packets transmitted that had Alignment errors or phy errors.
InCollisions	The total number of packets received in which a Collision event was detected.
OutCollisions	The total number of packets transmitted in which a Collision event was detected.
OutLateCollisions	The total number of packets transmitted in which a Collision event was detected, but for which a receive error (Rx Error) event was not detected.
Alignment	The total number of packets received that were from 64 – 1518 octets long, but had either a bad FCS with an integral number of octets (FCS Error) or a bad FCS with a non-integral number of octets (Alignment Error).
FCS	The Frame Checksum error.
InFlowCtrlPkts	The total number of ingress flow control packets. “N/A” indicates that the interface module does not support flow control statistics.
OutFlowCtrlPkts	The total number of egress flow control packets. “N/A” indicates that interface module does not support flow control statistics.
GiantPkts	The total number of packets for which all of the following was true: • The data length was longer than the maximum allowable frame size. • No Rx Error was detected. This counter is only for 10GbE interfaces.
ShortPkts	The total number of packets received for which all of the following was true: • The data length was less than 64 bytes. • No Rx Error was detected. • No Collision or late Collision was detected.
InBitsPerSec	The number of bits received per second.

TABLE 6 Output parameters of the **show statistics ethernet** command

Field	Description
OutBitsPerSec	The number of bits transmitted per second.
InPktsPerSec	The number of packets received per second.
OutPktsPerSec	The number of packets transmitted per second.
InUtilization	The percentage of the port's bandwidth used by received traffic.
OutUtilization	The percentage of the port's bandwidth used by transmitted traffic.

Ethernet interface debug commands

There are no specific Ethernet interface debug commands.

Common diagnostic scenarios

The following issues can occur with Ethernet interfaces:

- Faulty hardware

  Whenever you encounter a connection problem, check for a faulty hardware. Replace cables, try another port, and check all cable connections. If you find a faulty port, contact Brocade Technical Support for assistance.

- Link failures

  Link failures can be due either to a failure of the transmission medium or of the devices at each end of a connection. Be sure to check all of the hardware involved in the link, including cables and ports.

- CSMA/CD

  The Carrier Sense Multiple Access (CSMA) with Collision Detection (CD) protocol controls access to shared Ethernet media. A switched network (for example, Fast Ethernet) may use a full-duplex mode with access to the full link speed between directly connected network interface cards (NICs), switch-to-NIC cables, or switch-to-switch cables.

- CRC errors

  The Cyclic Redundancy Check (CRC) length specifies whether the CRC portion of each frame transmitted on the interface is 16 bits or 32 bits long. The default is 32 bits.

  A CRC alignment error is generated when the total number of packets received is from 64 through 1518 octets, but contains either a bad FCS with an integral number of octets (FCS Error) or a bad FCS with a non-integral number of octets (Alignment Error).

- Runts

  Any received packet that is less than 64 bytes is illegal, and is called a runt. In most cases, runts arise from a collision, and although they indicate an illegal reception, they may occur on correctly functioning networks. The receiving Brocade device discards all runt frames.

- Giants

  Any received packet that is greater than the maximum frame size is called a giant. In theory, the jabber control circuit in the transceiver must prevent any node from generating such a frame, but certain failures in the physical layer may also give rise to oversized Ethernet frames. Like runts, giants are discarded by the receiving Brocade device.

- Misaligned frames

  Any frame that does not contain an integral number of received octets (bytes) is also illegal. A receiver has no way of knowing which bits are legal, and how to compute the CRC-32 of the frame. Such frames are therefore also discarded by the Ethernet receiver.

- Old software versions

  Feature issues are often caused because the device is running an old version of the software. Brocade recommends that you always update software to reflect the latest patches and versions. If you have questions about your software version, contact Brocade Technical Support for assistance.

Link fault signaling

Link fault signaling (LFS) is a physical layer protocol that enables communication on a link between two 10 Gigabit Ethernet devices. When LFS is configured on a Brocade 10 Gigabit Ethernet port, the port can detect and report fault conditions on transmit and receive ports. Brocade IronWare software supports LFS among all 10 Gigabit Ethernet devices, including LFS support between First and Second generation devices. LFS is disabled by default on 10 Gbps interfaces.

When LFS is enabled on an interface, the following syslog messages are generated whenever the interface goes up or down, or when the TX or RX fiber is removed from both sides of a link where LFS is configured.

```
interface ethernet1/1, state down - link down
interface ethernet1/1, state up
```

After the fiber is installed, the Link and Activity LEDs on the module light up when traffic is flowing across the link.

NOTE
LFS is *disabled* by default on 10 Gbps interfaces. It is *enabled* by default (and cannot be disabled) on 1 Gpbs interfaces.

LFS show commands

This section describes the show command that displays LFS information.

show link-fault-signaling

Syntax: **show link-fault-signaling**

This command displays the LFS state configured on ports.

```
Brocade# show link-fault-signaling
Global Remote Fault : OFF
PORT #: REMOTE FAULT:
PORT 1/1: OFF
PORT 1/2: OFF
PORT 1/3: OFF
PORT 1/4: ON
PORT 1/5: OFF
PORT 1/6: OFF
PORT 1/7: OFF
PORT 1/8: OFF
PORT 1/9: OFF
```

LFS debug commands

There are no specific link fault signaling debug commands.

Packet over SONET modules

Packet over SONET (POS) is a transport technology that encapsulates packet data such as an IP datagram directly into Synchronous Optical Network (SONET) using Point-to-Point Protocol (PPP) or High-level Data Link Control (HDLC). POS modules allow direct connection to interfaces within SONET.

MIBs for POS modules

Several MIB objects allow you to collect errors for POS links. SONET equipment detects alarms and error conditions from the three layers of the SONET protocol: section, line, and path. Other devices on the network are notified of these events. For more information about these MIBs, refer to the *Unified IP MIB Reference*.

Automatic protection switching

You can manually force a protect interface to take over as a working interface by entering the **aps force** command on the protect interface. This command is useful when you want to bring down a working interface for maintenance purposes.

Path Trace (J1 byte) field

The Path Trace (J1 byte) field in the SONET payload envelope (SPE) transmits a 64-byte, fixed-length string that the receiving Path Terminating Equipment uses to verify the connection to the sending device. You can configure the string that a POS interface transmits in the Path Trace field, and you can display Path Trace strings received and configured on the POS interface.

Cable specifications

Table 7 lists the cable specifications for POS modules.

TABLE 7 POS interface specifications

Transceiver	Power budget	Launch window	Transmit power	Receive power	Maximum distance
OC-3c POS interfaces					
Single-mode short-reach	13 dB	1270 to 1380 nm	-28 to -8 dBm	-31 to -8 dBm	9.75 miles (15 Km)
Single-mode intermediate-reach	29 dB	1280 to 1335 nm	-5 to 0 dBm	-34 to -8 dBm	26 miles (40 Km)
Multimode	11.5 dB	1270 to 1380 nm	-18 to -14 dBm	-30 to -14 dBm	1.3 miles (2 Km)
OC-12c POS interfaces					
Single-mode long-reach	25 dBm	1280 to 1335 nm	-3 to 2 dBm	-28 to -8 dBm	65 miles (100 Km)
Single-mode intermediate-reach	13 dBm	1274 to 1356 nm	-15 to -8 dBm	-28 to -7 dBm	9.32 miles (40 Km)[1]
Multimode short-reach	6 dBm	1270 to 1380 nm	-20 to -14 dBm	-26 to -14 dBm	1640 ft. (500 m)
OC-48c POS interfaces					
Single-mode short-reach	8 dB	1260 to 1580 nm	-10 to -3 dBm	-18 to -3 dBm	9.75 miles (15 Km)
Single-mode intermediate-reach	13 dB	1260 to 1360 nm	-5 to 0 dBm	-18 to 0 dBm	26 miles (40 Km)

1. If the transceiver part number for the OC-12c POS module is HFCT-5208B (before May 2000), the maximum distance is 15 Km, not 40 Km.

NOTE
It is recommended that you do not use Network Processor Architecture (NPA) POS modules and non-NPA POS modules in the same circuit. For example, if you have an NPA module on one end of a POS link, do not use a non-NPA module on the other end of the link.

POS show commands

Use the following commands to display information about POS modules.

show pos next-hop

Syntax: **show pos next-hop**

This command shows the next hop that was learned through POS interface. Command output resembles the following example.

```
Brocade# show pos next-hop
Total number of POS next-hop entries: 1
Entries in default routing instance:
      IP Address          Type        Age     Port
1     10.1.0.1            Other       0       4/1
```

show interface brief

Syntax: show interface brief [**pos** *slotnum/portnum* | **slot** *slotnum*]

This command displays brief interface information about the specified POS port or all the ports in the specified slot. The following example shows information for the module in slot 2.

```
Brocade# show interface brief slot 2
Port Link L2 State Dupl Speed Trunk Tag Priori MAC
1/1 Up Forward Full 622M None No level0 0000.00a0.4400
1/2 Up Forward Full 622M None No level0 0000.00a0.4401
1/3 Up Forward Full 622M None No level0 0000.00a0.4402
```

show interface pos

Syntax: show interface pos *slotnum/portnum*

This command displays detailed information about the configured (as opposed to transmitted) and received Path Trace strings. The following example shows that POS is enabled for port 1/1.

```
Brocade# show interface pos 1/1
PacketOverSonet 1/1 is up, line protocol is down (PPP down)
  Hardware is PacketOverSonet , address is 0000.00f3.5500 (bia 0000.00f3.5500)
  Configured speed 2488Mbit, actual 2488Mbit
  Crc: 32, Keep Alive: 10 sec, C2 Byte: 16, Scrambling: On, SSM = STU
  mirror disabled, monitor disabled
  Not member of any active trunks
  Not member of any configured trunks
  No port name
  MTU 4474 bytes, encapsulation PPP
  LCP     FSM: state CLOSED(2)
  IPCP    FSM: state INITIAL(0)
  IPV6CP  FSM: state INITIAL(0)
  OSINLCP FSM: state INITIAL(0)
  MPLSCP  FSM: state INITIAL(0)
  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
  218181 packets input, 125408099181 total bytes, 13090860 good bytes
  Input 7282 short pkts, 28169 long pkts, 0 abort pkts
  Input 3035 fcs err pkts, 0 drop pkts, 0 drop abort pkts,124565265788 invalid
pkts
  NP received 218183 packets, Sent to TM 218182 packets
  NP Ingress dropped 1 packets
  219582 packets output, 14060380 total bytes, 13174920 good bytes
  Output 0 short pkts, 0 long pkts, 0 abort pkts,  0 drop abort pkts
  NP transmitted 219584 packets, Received from TM 219584 packets
```

The following example shows that POS is disabled for port 4/1.

```
Brocade# show interface pos 4/1
PacketOverSonet 4/1 is disabled, line protocol is down
  Hardware is PacketOverSonet , address is 0000.00f3.a778 (bia 0000.00f3.a778)
  Configured speed 2488Mbit, actual unknown
  Crc: 32, Keep Alive: 10 sec, C2 Byte: 16, Scrambling: On, SSM = STU
  mirror disabled, monitor disabled
  Not member of any active trunks
  Not member of any configured trunks
  No port name
  MTU 4474 bytes, encapsulation PPP
  LCP     FSM: state INITIAL(0)
```

```
  IPCP    FSM: state INITIAL(0)
  IPV6CP  FSM: state INITIAL(0)
  OSINLCP FSM: state INITIAL(0)
  MPLSCP  FSM: state INITIAL(0)
  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 total bytes, 0 good bytes
  Input 0 short pkts, 0 long pkts, 0 abort pkts
  Input 0 fcs err pkts, 0 drop pkts, 0 drop abort pkts,  0 invalid pkts
  NP received 11 packets, Sent to TM 0 packets
  NP Ingress dropped 11 packets
  0 packets output, 0 total bytes, 0 good bytes
  Output 0 short pkts, 0 long pkts, 0 abort pkts,  0 drop abort pkts
  NP transmitted 0 packets, Received from TM 0 packets
```

show controller

Syntax: **show controller** [**curr15min** | **day** | **prev15min**] [**ethernet** *slotnum/portnum* | **pos** *portnum* | **slot** *slotnum*]

- **curr15min** - Shows statistics for the current 15-minute interval.
- **day** - Shows statistics for the day.
- **prev15min** - Shows statistics for the previous 15-minute interval.
- **ethernet** *slotnum/portnum* - Shows statistics for the specified interval for an Ethernet port.
- **pos** *portnum* - Shows statistics for the specified interval for a POS port.
- **slot** *slotnum* - Shows statistics for all the ports in the slot specified.

This command displays the alarm statistics for the specified port for the specified interval, as shown in the following example.

```
Brocade# show controller curr15min pos 4/1
 ------------------------------------------
  POS alarms statistics  - PORT pos4/1
 ------------------------------------------
  ACTIVE ALARMS :  None
  ACTIVE DEFECTS :  None

Elapsed time [7 min 3 secs]
FM-PARAMS
 Section
   LOS = 0  LOF = 0  TIM-S = 0
 Line
   AIS-L = 0  RFI-L = 0  SD-L = 0  SF-L = 0
 Path
   AIS-P = 0  RFI-P = 0  LOP = 0  PLM-P = 0
   AIS-PFE = 0  PLM-PFE = 0  UNEQ-P = 0  TIM-P = 0

PM-PARAMS
 Section
   CV = 0  ES = 0  SES = 0  SEFS = 0
 Line
   CV = 0  ES = 0  SES = 0  UAS = 0
   CV-FE  = 0  ES-FE = 0  SES-FE = 0  UAS-FE = 0
 Path
   CV = 0  ES = 0  SES = 0  UAS = 0
   CV-FE  = 0  ES-FE = 0  SES-FE = 0  UAS-FE = 0  PJ = 0
```

show statistics pos

Syntax: **show statistics pos** [*slotnum/portnum*]

This command shows statistical information for a specified port, as shown in the following example.

```
Brocade# show statistics pos 1/1

PORT 1/1 Counters:
     In Good Pkts           218181      Out Good Pkts            219582
         In Bytes     125409302971          Out Bytes          14060380
    In Good Bytes         13090860     Out Good Bytes          13174920
    In Short Pkts             7282     Out Short Pkts                 0
     In Long Pkts            28169      Out Long Pkts                 0
    In Abort Pkts                0     Out Abort Pkts                 0
     In FCS Error             3035
    In Discrd Pkt                0     Out Discrd Pkt                 0
      In Drop Pkt                0
   In Invalid Pkt     124566469576
     InBitsPerSec                0      OutBitsPerSec                 0
     InPktsPerSec                0      OutPktsPerSec                 0
    InUtilization             0.0%     OutUtilization              0.0%
```

show statistics brief pos

Syntax: **show statistics brief pos** [*slotnum/portnum*]

This command displays a brief summary of statistics for all POS modules. You can also display the statistics for a single module using the *slotnum* and *portnum* variables.

```
Brocade# show statistics pos brief
POS           Packets                     Errors
Port    [Receive  Transmit]  [Align     FCS   Giant  Short]
2/1         1475      12301       0    1378       3      0
```

show pos-timing

Syntax: **show pos-timing**

This command displays POS timing information, as shown in the following example.

```
Brocade# show pos-timing
Line card  1 Timing Info:
cur_state = Wait For Configuration
Driving daisy chain1 = False daisy chain2 = False
Clock Lock State: Freerun
Curr Selected Clock: Invalid Clock
Highest Priority Clock: Invalid Clock
Second Highest Priority Clock: Invalid Clock
Third Highest Priority Clock: Invalid Clock
```

POS APS show command

This section describes the show command that displays information about a POS APS configuration.

show aps

Syntax: show aps

This command displays information about a POS APS configuration.

```
Brocade# show aps
POS2/1 working group 1 channel 1 Enabled Selected
```

This output indicates that POS interface 2/1 is the working interface for channel 1 in APS group 1, and the interface is active. A tilde next to Selected (for example, ~Selected) indicates that the interface is not active.

Clearing POS queue level statistics

You can clear POS queue level statistics from the following queues:

- Per-queue or all queues
- Multicast queue
- CPU queue
- CPU copy queue

Clearing all POS queue level statistics

The following command clears all POS queue level statistics.

clear tm-voq-stat src_port pos dst_port pos

Syntax: clear tm-voq-stat src_port pos *source-port* **dst_port pos | ethernet** *destination-port priority*

You must specify the *source-port* and the *destination-port* for this command. Optionally, you can specify a priority to clear only queue level statistics for a single priority.

Clearing multicast POS queue level statistics

The following command clears multicast POS queue level statistics.

clear tm-voq-stat src_port pos multicast

Syntax: clear tm-voq-stat src_port pos *source-port* **multicast** *priority* | **all**

```
Brocade# clear tm-voq-stat src_port pos 1/1 multicast 0
```

You must specify the *source-port* for this command. Optionally, you can specify a priority to clear queue level statistics for a single priority or use the **all** parameter to clear all queue level statistics.

Clearing CPU POS queue level statistics

The following command clears CPU POS queue level statistics.

clear tm-voq-stat src_port pos cpu-queue

Syntax: clear tm-voq-stat src_port pos *source-port* **cpu-queue** *priority* | **all**

```
Brocade# clear tm-voq-stat src_port 1/1 cpu-queue 0
```

You must specify the *source-port* for this command. Optionally, you can specify a priority to clear queue level statistics for a single priority or use the **all** parameter to clear all queue level statistics.

Clearing CPU copy queue level statistics

The following command clears the CPU copy queue level statistics.

clear tm-voq-stat src_port pos cpu-copy-queue

Syntax: clear tm-voq-stat src_port pos *source-port* **cpu-copy-queue** *priority* | **all**

```
Brocade# clear tm-voq-stat src_port 1/1 cpu-copy-queue all
```

You must specify the *source-port* for this command. Optionally, you can specify a priority to clear queue level statistics for a single priority or use the **all** parameter to clear all queue level statistics.

Configuration notes

- When you change the maximum frame size setting for a POS module, you must write to memory and reboot for the change to take effect.
- Both sides of a POS link must use the same encapsulation type.
- The setting for POS keepalive messages must be the same on both ends of the link. Keepalive messages are enabled by default for POS interfaces.
- The Data-Link Connection Identifier (DLCI) is the circuit ID for the link and can be a number in the range from 1 through 1023. The circuit ID must be the same on both ends of the link.
- POS modules use their own field-programmable gate arrays (FPGAs).
- Bandwidth – You can change a 2488 Mbps port to run at 622 Mbps if needed.
- SPE scrambling is enabled by default. Both ends of a link must use the same scrambling algorithm.
- Cyclic Redundancy Check (CRC) – You can specify the length of the CRC field in each packet sent by the POS interface. The default length is 32 bits. You can change the length to 16 bits.
- POS overhead flags – You can change the following values in the POS frame header:
  - c2 – Payload content type. The default is 0x16, which specifies PPP with scrambling enabled.
  - j1 – Path trace byte. Default is 0x0.
  - j0 – Section trace byte. Default is 0x1.

Common diagnostic scenarios

Issues with POS modules are usually hardware-related. The following problems may occur:

- Memory defects are in the module hardware.
- The module is not sending bridge protocol data units (BPDUs).
- Free queue depth levels are rising and falling.
- Modules in a particular chassis slot are showing similar errors. In this case, the chassis slot connector may be faulty. Contact Brocade Technical Support if you suspect a faulty chassis slot.

802.1ag CFM

Connectivity Fault Management (CFM) is a standard protocol defined in IEEE 802.1ag that provides fault localization functionalities and monitors the health of a link in an end-to-end Ethernet framework. The CFM protocol which supports over Virtual Local Area Network (VLAN), Virtual Private LAN Service (VPLS), Virtual Lease Line (VLL), and Provider Backbone Bridging (PBB) networks monitors the connectivity of the ports in the network and gives appropriate scope to customers, providers, and operators.

CFM show commands

This section describes the CFM-related show commands.

show cfm

Syntax: **show cfm** [**domain name**] [**ma name**] **brief**

- **domain name** - Displays CFM information for the specified domain name.
- **ma name** - Displays CFM information for the specified maintenance association (MA) name.

This command displays the current configuration and status of CFM. Command output resembles the following example.

```
Brocade# show cfm
Domain: D1
Index: 1
Level: 3
  Maintenance association: MA
  Ma Index: 1
  CCM interval: 3.3 ms
  PBB-VPLS ID: 100
  Priority: 3
  ETH-AIS TX: DISABLED
  ETH-AIS RX: DISABLED
  ETH-AIS Interval: 10 sec
  MEP Direction MAC PORT PORT-STATUS-TLV
  ==== ========= ======== ==== ==============
  7900 DOWN 0000.002a.6234 ethe 2/5 N

  MIP VLAN/VC Port Level MAC
  ==== ======== ===== ===== ==========
              300 2/5 3 0000.002a.6234
```

show cfm brief

Syntax: **show cfm brief**

This command displays a summary of the configured Maintenance End Points (MEPs) and Remote MEPs (RMEPs). Command output resembles the following example.

```
Brocade# show cfm brief
Domain: D1
Index: 1
Level: 3 Num of MA: 1
  Maintenance association: MA
  MA Index: 1
  CCM interval: 3.3 ms
  PBB-VPLS ID: 100
  Priority: 3
  ETH-AIS TX: DISABLED
  ETH-AIS RX: DISABLED
  ETH-AIS Interval: 10 sec
  Num of MEP: 1 Num of RMEP: 1
  rmepstart: 0 rmepfail: 0 rmepok 1
```

show cfm connectivity

Syntax: **show cfm connectivity** [**domain** *name*] [**ma** *name*] [**rmep-id** *num*]

- **domain** *name* - Displays CFM information for the specified domain name.
- **ma** *name* - Displays CFM information for the specified MA name.
- **rmep-id** *num* - Displays CFM information for the specified Remote MEP (RMEP) ID.

This command displays connectivity statistics for the remote database. Command output resembles the following examples.

```
Brocade# show cfm connectivity
Domain: D1 Index: 1
Level: 3
  Maintenance association: MA
  MA Index: 1
  CCM interval: 3.3 ms
  PBB-VPLS ID: 100
  Priority: 3
  ETH-AIS TX: ENABLED
  ETH-AIS RX: ENABLED
  ETH-AIS Interval: 60 sec
  RMEP  MAC        VLAN/ISID AGE    PORT     SLOTS    STATE    AIS_STATE
  ====  =====      ========= =====  =====    ======   ======   =========
  11   0000.002a.cf94 11000  1523   2/6       2       OK       AIS recvd
```

This command displays connectivity statistics for the specified domain name. Command output resembles the following examples.

```
Brocade# show cfm connectivity domain customer ma customer rmep-id 11
Domain: D1 Level: 3
Maintenance association: MA PBB-VPLS VLAN/VPLS/VLL ID: 100 Priority: 3
CCM interval: 3.3 ms
RMEP   MAC            PORT  Oper   Age    CCM   RDI   Port    Intf    Intvl   Seq     core
                            State  Val    Cnt         Status  Status  Error   Error   Fault level
===   ======          ====  =====  =====  ===== ====  ====    =====   ======  ======  ========
11    0000.002a.cf94  2/6   OK     1532   16483  N     0      0        N      N       1
```

CFM debug commands

This section describes the CFM-related debug commands.

debug cfm

Syntax: [**no**] **debug cfm** *level*

This command enables 802.1ag debugging. The *level* variable is a value from 0 through 7. Setting the level to 7 enables debugging of all CFM packets.

Command output resembles the following example.

```
Brocade# debug cfm 1
        CFM:  debug level is 1
Dec 10 17:36:35 CFM: receive packet :rx_l2_bpdu 12
Dec 10 17:36:35 CFM: receive packet :rx_l2_bpdu 12
Dec 10 17:36:36 CFM: receive packet :rx_l2_bpdu 12
Dec 10 17:36:36 CFM: receive packet :rx_l2_bpdu 17
Dec 10 17:36:36 CFM: receive packet :rx_l2_bpdu 12
Dec 10 17:36:36 CFM: receive packet :rx_l2_bpdu 12
Dec 10 17:36:36 CFM: receive packet :rx_l2_bpdu 12
Dec 10 17:36:36 CFM: receive packet :rx_l2_bpdu 17
Dec 10 17:36:36 CFM: receive packet :rx_l2_bpdu 12
Dec 10 17:36:36 CFM: receive packet :rx_l2_bpdu 12
Dec 10 17:36:36 CFM: receive packet :rx_l2_bpdu 12
Dec 10 17:36:36 CFM: receive packet :rx_l2_bpdu 12
Dec 10 17:36:36 CFM: receive packet :rx_l2_bpdu 12
Dec 10 17:36:36 CFM: receive packet :rx_l2_bpdu 12
Dec 10 17:36:36 CFM: receive packet :rx_l2_bpdu 12
Dec 10 17:36:36 CFM: receive packet :rx_l2_bpdu 17
Dec 10 17:36:36 CFM: receive packet :rx_l2_bpdu 12
Dec 10 17:36:36 CFM: receive packet :rx_l2_bpdu 12
```

debug cfm ipc

Syntax: [**no**] **debug cfm ipc**

This command enables debugging and displays information for CFM IPC events. Command output resembles the following example.

```
Brocade# debug cfm ipc
Brocade(config-cfm-md-test)# show vlan-group 1
Brocade(config-cfm-md-test)# vlan-group 1
SYSLOG: <14>Dec 10 19:59:12 Dist1 VLAN Group 1: added by unknown from console
session.
Brocade(config-vlan-group-1)# add 3001 to 3010
SYSLOG: <14>Dec 10 19:59:19 Dist1 VLAN Group 1: VLAN: 3001  to VLAN: 3010  VLANs
added to by unknown from console session.
Brocade(config-vlan-group-1)# exit
Brocade(config)#cfm-enable
Brocade(config-cfm)# domain-name test id 1 level 7
Brocade(config-cfm-md-test)#  ma-name 3001 id 1 vlan-id 3001 priority 0
Dec 10 19:59:26 Send MA CO to LP: md test ma 3001 Add 1
Brocade(config-cfm-md-test-ma-3001)#  ma-name 3002 id 2 vlan-id 3002 priority 1
Dec 10 19:59:27 Send MA CO to LP: md test ma 3002 Add 1
Brocade(config-cfm-md-test-ma-3002)#  ma-name 3003 id 3 vlan-id 3003 priority 2
Dec 10 19:59:28 Send MA CO to LP: md test ma 3003 Add 1
Brocade(config-cfm-md-test-ma-3003)#  ma-name 3004 id 4 vlan-id 3004 priority 3
Dec 10 19:59:28 Send MA CO to LP: md test ma 3004 Add 1
Brocade(config-cfm-md-test-ma-3004)#  ma-name 3005 id 5 vlan-id 3005 priority 4
Dec 10 19:59:29 Send MA CO to LP: md test ma 3005 Add 1
Brocade(config-cfm-md-test-ma-3005)#  ma-name 3006 id 6 vlan-id 3006 priority 5
Dec 10 19:59:30 Send MA CO to LP: md test ma 3006 Add 1
```

```
Brocade(config-cfm-md-test-ma-3006)#  ma-name 3007 id 7 vlan-id 3007 priority 6
Dec 10 19:59:30 Send MA CO to LP: md test ma 3007 Add 1
Brocade(config-cfm-md-test-ma-3007)#  ma-name 3008 id 8 vlan-id 3008 priority 7
Dec 10 19:59:35 Send MA CO to LP: md test ma 3008 Add 1
Brocade(config-cfm-md-test-ma-3008)#end
Brocade(config)#vlan-group 1
Brocade(config-vlan-group-1)#tag ethernet 3/1
Dec 10 20:00:55 CFM configuration for Vlan 3001 deleted due to vlan configuration
changes
Dec 10 20:00:55  dot1ag_clean_cfm_port_for_ma:1
Dec 10 20:00:55 Send MA CO to LP: md test ma 3001 Add 0
Dec 10 20:00:55 CFM configuration for Vlan 3002 deleted due to vlan configuration
changes
Dec 10 20:00:55  dot1ag_clean_cfm_port_for_ma:1
Dec 10 20:00:55 Send MA CO to LP: md test ma 3002 Add 0
Dec 10 20:00:55 CFM configuration for Vlan 3003 deleted due to vlan configuration
changes
Dec 10 20:00:55  dot1ag_clean_cfm_port_for_ma:1
Dec 10 20:00:55 Send MA CO to LP: md test ma 3003 Add 0
Dec 10 20:00:55 CFM configuration for Vlan 3004 deleted due to vlan configuration
changes
Dec 10 20:00:55  dot1ag_clean_cfm_port_for_ma:1
Dec 10 20:00:55 Send MA CO to LP: md test ma 3004 Add 0
Dec 10 20:00:55 CFM configuration for Vlan 3005 deleted due to vlan configuration
changes
Dec 10 20:00:55  dot1ag_clean_cfm_port_for_ma:1
Dec 10 20:00:55 Send MA CO to LP: md test ma 3005 Add 0
Dec 10 20:00:55 CFM configuration for Vlan 3006 deleted due to vlan configuration
changes
Dec 10 20:00:55  dot1ag_clean_cfm_port_for_ma:1
Dec 10 20:00:55 Send MA CO to LP: md test ma 3006 Add 0
Dec 10 20:00:55 CFM configuration for Vlan 3007 deleted due to vlan configuration
changes
Dec 10 20:00:55  dot1ag_clean_cfm_port_for_ma:1
Dec 10 20:00:55 Send MA CO to LP: md test ma 3007 Add 0
Dec 10 20:00:55 CFM configuration for Vlan 3008 deleted due to vlan configuration
changes
Dec 10 20:00:55  dot1ag_clean_cfm_port_for_ma:1
Dec 10 20:00:55 Send MA CO to LP: md test ma 3008 Add 0
Brocade(config-vlan-group-1)#end
SYSLOG: <14>Dec 10 20:01:06 Dist1 Security: running-config was changed from
console
```

debug cfm md

Syntax: [**no**] **debug cfm md** *mdName* **ma** *maName* [**mep-id** *mepId* | **rmep-id** *rmepId*]

- **md** *mdName* - Enables debugging traces for the specified Maintenance Domain (MD).
- **ma** *maName* - Enables debugging traces for the specified Maintenance Association (MA).
- **mep-id** *mepId* - Enables debugging traces for the specified Maintenance End Point (MEP).
- **rmep-id** *rmepId* - Enables debugging traces for the specified Remote MEP (RMEP).

This command enables and displays debugging information at the MA, MEP, and RMEP level.

The **debug cfm md** *mdName* **ma** *maName* command displays debug logs for the specified MD at the MA level. Command output resembles the following example.

```
Brocade# debug cfm md md1 ma ma1
CFM:  MA debugging is on
```

```
Mar 21  05:24:43 CFM: TX CCM with MEP id 4 Seq num 759 on MD md1 MA ma1
Mar 21  05:24:53 CFM: TX CCM with MEP id 4 Seq num 760 on MD md1 MA ma1
```

The **debug cfm md** *mdName* **ma** *maName* **mep-id** *mepId* command displays debug logs for the specified MD at the MA and MEP level. Command output resembles the following example.

```
Brocade# debug cfm md md1 ma ma1 mep-id 4
CFM:  MEP debugging is on
Mar 21  05:25:13 CFM: TX CCM with MEP id 4 Seq num 762 on MD md1 MA ma1
Mar 21  05:25:23 CFM: TX CCM with MEP id 4 Seq num 763 on MD md1 MA ma1
Mar 21  05:25:33 CFM: TX CCM with MEP id 4 Seq num 764 on MD md1 MA ma1
```

The **debug cfm md** *mdName* **ma** *maName* **rmep-id** *rmepId* command displays debug logs for the specified MD at the MA and RMEP level. Command output resembles the following example.

```
Brocade# debug cfm md md1 ma ma1 rmep-id 6
CFM:  RMEP debugging is on
Mar 21  05:29:02 dot1aglp_process_equal_ccm: RMEP 6 in Domain md1 MA ma1
Mar 21  05:29:02     Seq number OK, received 7, ours 7
Mar 21  05:29:12 dot1aglp_process_equal_ccm: RMEP 6 in Domain md1 MA ma1
Mar 21  05:29:12     Seq number OK, received 8, ours 8
```

debug cfm packet

Syntax: [**no**] **debug cfm packet**

This command enables debugging and displays information about CFM packets. This command is used to monitor the receipt of the ETH-DM packet on the LP.

NOTE
This is a generic debug command and displays the information for all CFM packets as well.

---

debug cfm detail

Syntax: [**no**] **debug cfm detail**

This command is used to dump the contents (time stamps) of the ETH-DM packet.

NOTE
This is a generic debug command and displays the information for all CFM packets as well.

The **debug cfm packet** and **debug cfm detail** command output resembles the following example.

```
Brocade# debug cfm
Brocade(config-cfm)# cfm-enable
Brocade(config-cfm)# domain-name test id 1 level 7
Brocade(config-cfm-md-test)# ma-name 3001 id 1 vlan-id 3001 priority 0
Dec 10 20:38:03 Send MA CO to LP: md test ma 3001 Add 1
Dec 10 20:38:03  dot1ag_default_create_mip
Dec 10 20:38:03  dot1ag_create_cfm_port vlan_id 3001 port_id 3/1 is_mep 0 is_mp 1
Dec 10 20:38:03 Send CFM port to LP: vlan 3001 port 3/1
Dec 10 20:38:03         level 7 is valid, is MEP 0, mep-id 0, is MIP 1
Brocade(config-cfm-md-test-ma-3001)# ma-name 3002 id 2 vlan-id 3002 priority 1
Dec 10 20:38:03 Send MA CO to LP: md test ma 3002 Add 1
Dec 10 20:38:03  dot1ag_default_create_mip
Dec 10 20:38:03  dot1ag_create_cfm_port vlan_id 3002 port_id 3/1 is_mep 0 is_mp 1
Dec 10 20:38:03 Send CFM port to LP: vlan 3002 port 3/1
Dec 10 20:38:03         level 7 is valid, is MEP 0, mep-id 0, is MIP 1
Brocade(config-cfm-md-test-ma-3002)# ma-name 3003 id 3 vlan-id 3003 priority 2
Dec 10 20:38:04 Send MA CO to LP: md test ma 3003 Add 1
Dec 10 20:38:04  dot1ag_default_create_mip
```

```
Dec 10 20:38:04  dot1ag_create_cfm_port vlan_id 3003 port_id 3/1 is_mep 0 is_mp 1
Dec 10 20:38:04 Send CFM port to LP: vlan 3003 port 3/1
Dec 10 20:38:04         level 7 is valid, is MEP 0, mep-id 0, is MIP 1
Brocade(config-cfm-md-test-ma-3003)# ma-name 3004 id 4 vlan-id 3004 priority 3
Dec 10 20:38:05 Send MA CO to LP: md test ma 3004 Add 1
Dec 10 20:38:05  dot1ag_default_create_mip
Dec 10 20:38:05  dot1ag_create_cfm_port vlan_id 3004 port_id 3/1 is_mep 0 is_mp 1
Dec 10 20:38:05 Send CFM port to LP: vlan 3004 port 3/1
Dec 10 20:38:05         level 7 is valid, is MEP 0, mep-id 0, is MIP 1
Brocade(config-cfm-md-test-ma-3004)# ma-name 3005 id 5 vlan-id 3005 priority 4
Dec 10 20:38:06 Send MA CO to LP: md test ma 3005 Add 1
Dec 10 20:38:06  dot1ag_default_create_mip
Dec 10 20:38:06  dot1ag_create_cfm_port vlan_id 3005 port_id 3/1 is_mep 0 is_mp 1
Dec 10 20:38:06 Send CFM port to LP: vlan 3005 port 3/1
Dec 10 20:38:06         level 7 is valid, is MEP 0, mep-id 0, is MIP 1
Brocade(config-cfm-md-test-ma-3005)# ma-name 3006 id 6 vlan-id 3006 priority 5
Dec 10 20:38:06 Send MA CO to LP: md test ma 3006 Add 1
Dec 10 20:38:06  dot1ag_default_create_mip
Dec 10 20:38:06  dot1ag_create_cfm_port vlan_id 3006 port_id 3/1 is_mep 0 is_mp 1
Dec 10 20:38:06 Send CFM port to LP: vlan 3006 port 3/1
Dec 10 20:38:06         level 7 is valid, is MEP 0, mep-id 0, is MIP 1
```

Chapter

Layer 2 Protocol Diagnostics 4

In this chapter

This chapter describes Layer 2 troubleshooting and diagnostic processes for the Brocade NetIron XMR and Brocade MLX series routers.

CAUTION

Enabling diagnostic commands may degrade system performance. These commands are best used to troubleshoot specific problems while working with qualified Brocade service technicians. Whenever possible, troubleshoot your system during periods of low network traffic and user activity to preserve system performance.

MAC address learning

In MAC address learning, the source MAC address of each received packet is stored so that future packets destined for that address can be forwarded only to the interface where that address is located. (Packets destined for unrecognized addresses are forwarded out every bridge interface.) MAC address learning, defined in the IEEE 802.1 standard, helps minimize traffic on the attached LANs.

Address Resolution Protocol

Routers use Address Resolution Protocol (ARP) to learn the MAC addresses of devices on the network. The router sends an ARP request that contains the IP address of a device, and receives the MAC address for that device in an ARP reply. These *dynamically* learned entries are stored in the ARP cache. You can also manually configure MAC addresses, which are called *static* entries.

A *static* ARP entry in the ARP cache resembles the following example.

```
Index IP Address          MAC Address Port
  1   10.95.6.111        0000.003b.d2101/1
```

A *dynamic* entry in the ARP cache resembles the following example.

```
IP AddressMAC AddressTypeAgePort
1   10.95.6.1020000.00fc.ea21 Dynamic06
```

ARP age

The ARP age is the amount of time the device keeps a learned MAC address in the ARP cache. The device resets the timer to zero each time the ARP entry is refreshed and removes the entry if the timer reaches the ARP age. The default ARP age is 10 minutes.

Changing the ARP aging period

When the switch places an entry in the ARP cache, it also starts an aging timer for the entry. The aging timer ensures that the ARP cache does not retain learned entries that are no longer valid. An entry can become invalid when the device with the MAC address of the entry is no longer on the network.

ARP age affects dynamic entries only. You cannot change the ARP age on Layer 2 switches. If you set the ARP age to zero, aging is disabled and entries do not age out.

Proxy ARP

Routers use Proxy ARP to answer ARP requests for a host by replying with the router MAC address instead of the host address.

MAC address learning show commands

The following commands display information about the MAC address table.

show mac-address

Syntax: show mac-address

This command displays the MAC address table, which contains MAC addresses learned from other devices or added using the **static-mac-address** command. This table does not contain the MAC addresses of the Brocade device ports.

```
Brocade# show mac-address
Total entries from all ports = 75
MAC             Port   Age    CamF   CIDX0  CIDX1  CIDX2  CIDX3  CIDX4  CIDX5
0000.0000.0000  10     17293  00H    0      0      0      0      0      0
0000.009f.8086  1      12     0bH    23     15     0      6      0      0
0000.0009.914b  16     2130   00H    0      0      0      0      0      0
0000.009a.0163  16     130    00H    0      0      0      0      0      0
0000.009d.41a5  11     475    00H    0      0      0      0      0      0
0000.00c5.01d1  11     0      0cH    0      0      20     14     0      0
0000.009d.41df  11     570    00H    0      0      0      0      0      0
0000.0059.4226  16     240    00H    0      0      0      0      0      0
0000.0059.4235  16     130    00H    0      0      0      0      0      0
0000.008f.725b  2      135    00H    0      0      0      0      0      0
0600.0059.4264  16     0      0aH    0      14     0      21     0      0
0000.00c5.02a1  16     15     09H    5      0      0      33     0      0
```

NOTE
The information displayed in the columns with headings, CamF, and CIDX0 through CIDX5, is not relevant for day-to-day management of the device. This information is used by the technical support staff for debug purposes. Contact Brocade Technical Support for more information.

show mac vpls

Syntax: **show mac vpls** *vpls-id mac-address starting-entry number-of-entries*

- *vpls-id* - Specifies the Virtual Private LAN Service (VPLS) ID for which database entries are displayed.
- *mac-address* - Specifies the MAC address for which database entries are displayed.
- *starting-entry* - Specifies the point in the database from which the entries are displayed. Entering 0 causes entries to be displayed from the start; entering 200 causes the first 200 entries to be skipped; and so on.
- *number-of-entries* - Specifies the number of database entries to be displayed from the *starting-entry*.

This command displays the contents of the VPLS MAC database, which stores entries associating MAC addresses with VC Label Switched Paths (LSPs). The following example displays the VPLS MAC database on the management processor.

```
Brocade# show mac vpls
Total VPLS mac entries in the table: 2 (Local: 2, Remote: 0)

VPLS       MAC Address    L/R/IB Port  Vlan(In-Tag)/Peer ISID     Age
====       ===========    ====== ====  ================= ====     ===
1          0000.0004.0000 IB     1/2   300               30000    0
1          0000.0003.0000 L      1/1   200               NA       0
```

To display a specific entry in the VPLS MAC database on the management processor, enter the following command.

```
Brocade# show mac vpls 1 0000.0004.0000
VPLS: 1         MAC:  0000.0004.0000     Age:     0
    Local MAC       Port: 1/2                 VLAN:    300 ISID:    30000
    Associated B-MAC: 0000.0000.0201
    Trunk slot mask: 0x00000000
```

MAC address learning debug commands

This section describes the debug commands that generate information about Address Resolution Protocol (ARP).

debug ip arp

Syntax: [**no**] **debug ip arp** [**event** | **ipc** | **itc** | **packet**]

- **event** - Displays information about ARP events.
- **ipc** - Displays information about ARP IPC messages.
- **itc** - Displays information about ARP ITC messages.
- **packet** - Displays information about ARP packets.

This command displays information about ARP transactions.The **debug ip arp** command enables debugging for all ARP variables, or you can enable the options individually. Command output resembles the following example.

```
Brocade# debug ip arp
         ARP:  event debugging is on
         ARP:  packets debugging is on
         ARP:  ipc debugging is on
         ARP:  itc debugging is on
Dec 10 14:49:53 IP/ARP: Pkt rcvd on port mgmt1, src IP = 10.47.13.205
Dec 10 14:49:53 IP/ARP: rcvd packet src 10.47.13.205 000000b4f775: dst
10.47.13.254 000000000000: Port mgmt1
Dec 10 14:49:53 IP/ARP: src 10.47.13.205  fwd route does not match ingress port
1536
Dec 10 14:49:53 IP/ARP: Pkt rcvd on port mgmt1, src IP = 10.20.179.11
Dec 10 14:49:53 IP/ARP: rcvd packet src 10.20.179.11 000000adec00: dst
10.20.185.70 000000000000: Port mgmt1
Dec 10 14:49:54 IP/ARP: Pkt rcvd on port mgmt1, src IP = 10.20.176.3
Dec 10 14:49:54 IP/ARP: rcvd packet src 10.20.176.3 0000000fb600: dst 10.20.185.4
000000000000: Port mgmt1
Dec 10 14:49:54 IP/ARP: Pkt rcvd on port mgmt1, src IP = 10.20.68.133
Dec 10 14:49:54 IP/ARP: rcvd packet src 10.20.68.133 000000d5d07a: dst
10.20.68.129 000000000000: Port mgmt1
```

debug ip arp event

Syntax: [**no**] **debug ip arp event**

This command displays information about ARP events, and is useful in determining whether the router is sending and receiving ARP requests. Command output is similar to the following example, which shows send and receive activity for ARP packets.

```
Brocade# debug ip arp event
IP/ARP: sent request for 10.223.143.22
IP/ARP: sent packet src 10.223.143.16 000000e2b000: dst 10.223.143.22
000000000000: Port 1062
IP/ARP: sent request for 10.223.143.3
IP/ARP: sent packet src 10.223.143.16 000000e2b000: dst 10.223.143.3
000000000000: Port 1062
IP/ARP: sent request for 10.28.144.206
IP/ARP: sent packet src 10.28.144.205 000000e2b000: dst 10.28.144.206
000000000000: Port 843
```

```
IP/ARP: Received arp request from Lp for dest 10.28.144.206 Port: 843 Router: 1
IP/ARP: sent request for 10.28.181.171
IP/ARP: sent packet src 10.28.181.161 000000e2b000: dst 10.28.181.171
000000000000: Port 753
```

debug ip arp ipc

Syntax: [no] debug ip arp ipc

This command generates information about ARP interprocess communication (IPC) activity. Command output resembles the following example.

```
Brocade# debug ip arp ipc
IP/ARP: Received arp request from Lp for dest 10.142.108.94 Port: 1049 Router: 1
IP/ARP: Received arp request from Lp for dest 10.142.106.82 Port: 848 Router: 1
IP/ARP: Received arp request from Lp for dest 10.142.108.94 Port: 1049 Router: 1
IP/ARP: Received arp request from Lp for dest 10.142.106.82 Port: 848 Router: 1
IP/ARP: Received arp request from Lp for dest 10.142.108.94 Port: 1049 Router: 1
IP/ARP: Received arp request from Lp for dest 10.28.144.206 Port: 843 Router: 1
IP/ARP: Received arp request from Lp for dest 10.142.78.18 Port:846 Router: 1
IP/ARP: Received arp request from Lp for dest 10.28.144.206Port: 843 Router: 1
IP/ARP: Received arp request from Lp for dest 10.142.78.18 Port: 846 Router: 1
```

debug ip arp itc

Syntax: [no] debug ip arp itc

This command generates information about ARP inter-task communications (ITC) activity, which communicates between the processing tasks concerning activity or configuration changes. Command output resembles the following example, which indicates that a static IP address was added.

```
Brocade# debug ip arp itc
ARP: itc debugging is on
IP/ARP: Add static arp for Addr: 10.1.1.19 Mac: 100120013019 Port: 1
Vrf_index: 0 Add: 1
```

debug ip arp packet

Syntax: [no] debug ip arp packet

This command displays information about ARP packet activity. Command output resembles the following example, which indicates that the source router is polling routers 10.142.106.98, 10.28.181.122, 10.223.143.27, and 10.142.108.94 to learn their MAC addresses and add them to the source router ARP table.

```
Brocade# debug ip arp packet
IP/ARP: sent request for 10.142.106.98
IP/ARP: sent packet src 10.142.106.97 000000e2b000: dst 10.142.106.98
000000000000: Port 114
IP/ARP: sent request for 10.28.181.122
IP/ARP: sent packet src 10.28.181.121 000000e2b000: dst 10.28.181.122
000000000000: Port 1045
IP/ARP: sent request for 10.28.181.172
IP/ARP: sent packet src 10.28.181.161 000000e2b000: dst 10.28.181.172
000000000000: Port 753
IP/ARP: sent request for 10.223.143.27
IP/ARP: sent packet src 10.223.143.16 000000e2b000: dst 10.223.143.27
000000000000: Port 1062
IP/ARP: sent request for 10.142.106.82
IP/ARP: sent packet src 10.142.106.81 000000e2b000: dst 10.142.106.82
000000000000: Port 848
IP/ARP: sent request for 10.142.108.94
```

debug mac

Syntax: [**no**] **debug mac** [**action** | **cam** | **error** | **info** | **learning** | **mport** | **port security**]

- **action** - Displays information about the MAC database.
- **cam** - Displays Layer 2 CAM settings.
- **error** - Displays MAC table management error messages.
- **info** - Displays MAC table management information.
- **learning** - Displays MAC database learning information.
- **mport** - Displays Multiport MAC event messages.
- **port security** - Displays MAC table management port security messages.

The **debug mac** command generates information about MAC address databases, actions, settings, MAC table management, and MAC learning.

debug mac action

Syntax: [**no**] **debug mac action** [**vlan** *vlan_id*] [*mac_address*]

- **vlan** *vlan_id* - Displays all MAC address-related actions for the specified VLAN.
- *mac_address* - Displays specified MAC address-related actions for all VLANs.

To display a specific MAC address-related action for a specific VLAN, you must specify both the VLAN ID and MAC address options.

Use the **debug mac action** command to display actions-related information for the MAC database.

```
Brocade# debug mac action
info - mac_static_flush() -  execution
info - mac_pms_flush() - execution
info - mac_static_flush() -  execution
MAC ACTION - Normal SPECIFIC FLUSH
Ports: ethe 2/2 to 2/3
Vlans: 1
MAC ACTION - Premature ALL_SYSTEM FLUSH
MAC ACTION - Normal SPECIFIC FLUSH
Ports: All Ports
Vlans: All VLANs
```

If both the VLAN ID and MAC address options are specified, output such as the following is displayed.

```
Brocade# debug mac action vlan 100 0000.0052.0000
info - mac_static_flush() - execution
info - mac_pms_flush() - execution
info - mac_static_flush() - execution
MAC ACTION - Normal SPECIFIC FLUSH
MAC ACTION - Premature ALL_SYSTEM FLUSH
MAC ACTION - Normal SPECIFIC FLUSH
```

debug mac error

Syntax: [**no**] **debug mac error**

Use the **debug mac error** command to display major errors related to MAC learning. The output in the following example indicates an insufficient amount of buffer space and results in a queue overflow.

```
Brocade# debug mac error
error - macmgr_ipc_distribute_table. system out of ipc buffers info -
macmgr_ipc_learn_sa_entry. Not learning. port not in forwarding state info -
macmgr_ipc_free_sa_entry. Queue overflowing error - macmgr_ipc_sync_change_age.
system out of buffer
```

debug mac info

Syntax: [**no**] **debug mac info**

Use the **debug mac info** command to generate information about MAC database activity, such as flushing and table distribution.

```
Brocade# debug mac info
info - macmgr_ipc_flush_entry. Send flush.
info - macmgr_ipc_distribute_table. Distributing. mac table to slot 4
```

debug mac learning

Syntax: [**no**] **debug mac learning** [**vlan** *vlan_id*] [*mac_address*]

- **vlan** *vlan_id* - Displays all MAC addresses for the specified VLAN.
- *mac_address* - Displays specified MAC addresses for all VLANs.

To display a specific MAC address for a specific VLAN, you must specify both the VLAN ID and MAC address options.

Use the **debug mac learning** command to track a specific MAC address or set of MAC addresses based on the VLAN ID, MAC address, or both, and display debug information about the MAC addresses learned.

CAUTION

The debug mac learning command may generate large amounts of output and degrade system performance. Use this command with caution.

If you specify only the MAC address, the **debug mac learning** command output resembles the following example.

```
Brocade# debug mac learning 0000.0052.0000
learning: debugging is on
info - macmgr_ipc_learn_da_entry. Learn DA 0000.0052.0000 IPC received
info - macmgr_ipc_learn_sa_entry. Learn SA IPC received Mar 20 23:58:30
mac address 0000.0052.0000. Port 42 vlan 100 Mar 20 23:58:30 info -
macmgr_ipc_sync_add_entry. Send add entry for 0000.0052.0000 port 42 vlan 512.
```

If you specify only the VLAN ID, the **debug mac learning** command output resembles the following example.

```
Brocade# debug mac learning vlan 200
learning: debugging is on
info - macmgr_ipc_learn_da_entry. Learn DA 0000.0052.0000 IPC received
info - macmgr_ipc_learn_sa_entry. Learn SA IPC received Mar 20 23:58:30
mac address 0000.00b1.a030. Port 42 vlan 200 Mar 20 23:58:30 info -
macmgr_ipc_sync_add_entry. Send add entry for 0000.00b1.a030 port 42 vlan 200.
info - macmgr_ipc_learn_da_entry. Learn DA 0000.0078.9123 IPC received
info - macmgr_ipc_learn_da_entry. Learn DA 0000.0078.9123 IPC received
```

If you specify both the VLAN ID and MAC address, the **debug mac learning** command output resembles the following example.

```
Brocade# debug mac learning vlan 300 0000.0052.0000
learning: debugging is on
info - macmgr_ipc_learn_da_entry. Learn DA 0000.0052.0000 IPC received
info - macmgr_ipc_learn_sa_entry. Learn SA IPC received Mar 20 23:58:30
mac address 0000.0052.0000. Port 42 vlan 300 Mar 20 23:58:30 info -
macmgr_ipc_sync_add_entry. Send add entry for 0000.0052.0000 port 42 vlan 300
```

debug ip icmp

Syntax: [**no**] **debug ip icmp** [**events** | **external_loop** | **internal_loop** | **packets** | **port_loop**]

- **events** - Displays information about ICMP events.
- **external_loop** - Displays ICMP external loop activity.
- **internal_loop** - Displays ICMP internal loop activity.
- **packets** - Displays information about ICMP packets.
- **port_loop** - Displays port loop activity.

The **debug ip icmp** command displays information about IPv4 Internet Control Message Protocol (ICMP) transactions. This command is useful in determining if a router is sending or receiving ICMP messages, and for troubleshooting end-to-end connections.

debug ip icmp events

Syntax: [**no**] **debug ip icmp events**

This command generates information about ICMP events, such as sent and received echo (ping) requests, destination-unreachable messages, and redirect messages. Command output resembles the following example.

```
Brocade# debug ip icmp events
ICMP: rcvd echo request packet of length 40 from 10.1.1.2
ICMP: send echo request packet of length 60 to 10.1.1.2
```

debug ip icmp packets

Syntax: [**no**] **debug ip icmp packets**

This command generates information about ICMP packets. Command output resembles the following example.

```
Brocade# debug ip icmp packets
ICMP:dst (10.2.3.4), src (0.0.0.0) echo request type
ICMP: Received message from 10.102.50.254 to 10.47.16.33 port mgmt1 type 11 size
36
ICMP: rxed error message from 10.102.50.254:May 23 16:11:32  original destination
10.47.16.33 ICMP Time Exceeded
IP/ICMP: rxed message: size: 36
ICMP: Received message from 10.98.68.129 to 10.47.16.33 port mgmt1 type 11 size
36
ICMP: rxed error message from 10.98.68.129:May 23 16:11:33 original destination
10.47.16.33
ICMP Time Exceeded
```

debug ipv6 icmp

Syntax: [**no**] **debug ipv6 icmp**

This command generates information about IPv6 Internet Control Message Protocol (ICMP) activity, such as sending or receiving ICMP requests, responses, ICMP error messages, and redirected ICMP packets. Command output resembles the following example.

```
Brocade# debug ipv6 icmp
ICMPv6: Sending Echo Request to 2001:DB8:1::6, length 24
ICMPv6: Received Echo Request from 2001:DB8:1::6, length 24
```

Configuration notes

- Enabling port-priority changes the source MAC address of all ARP packets to a virtual MAC address.
- The location of the **static-mac-address** command in the CLI depends on whether you configure port-based VLANs on the device. If the device does not have more than one port-based VLAN (VLAN 1, which is the default VLAN that contains all the ports), the **static-mac-address** command is at the global CONFIG level of the CLI. If the device has more than one port-based VLAN, then the **static-mac-address** command is not available at the global CONFIG level. In this case, the command is available at the configuration level for each port-based VLAN.
- ARP is enabled by default and cannot be disabled.
- The ARP request broadcast is a MAC broadcast, which means it goes only to devices that are directly attached to the switch. A MAC broadcast is not routed to other networks. However, some routers, including Brocade Layer 3 switches, can be configured to reply to ARP requests from one network on behalf of devices on another network.
- If the router receives an ARP request packet it cannot deliver to the final destination because of the ARP time-out, and no ARP response is received, the router sends an ICMP Host Unreachable message to the source.

Super Aggregated VLANs

A Super Aggregated VLAN (SAV) contains multiple VLANs. This feature allows you to construct Layer 2 paths and channels. A path contains multiple channels, each of which is a dedicated circuit between two endpoints. The two devices at the endpoints of the channel appear to each other to be directly attached, yet the network that connects them remains transparent.

You can aggregate up to 4090 VLANs inside a SAV, for a total VLAN capacity on one router of 16,728,100 channels (4090 * 4090). Because devices connected through the channel are not visible to devices in other channels, each client has a private link to the other side of the channel. SAVs are useful for applications such as Virtual Private Network (VPN) or Transparent LAN Services (TLS) where clients need a private, dedicated Ethernet connection that can reach its subnet transparently across multiple networks. A SAV allows point-to-point and point-to-multipoint connections.

SAV show commands

This section describes the show commands that display VLAN information.

show vlan

Syntax: **show vlan**

This command displays information about configured VLANs, as shown in the following example.

```
Brocade# show vlan
Configured PORT-VLAN entries: 3
Maximum PORT-VLAN entries: 4090
Default PORT-VLAN id: 1
PORT-VLAN 1, Name DEFAULT-VLAN, Priority Level0
L2 protocols: NONE
Untagged Ports: ethernet 2/1 to 2/20 ethernet 3/1 to 3/20 ethernet
PORT-VLAN 2, Name [None], Priority Level0
L2 protocols: NONE
ip-protocol VLAN, Dynamic port disabled
Name: basic
PORT-VLAN 1001, Name [None], Priority Level0
L2 protocols: MRP
Tagged Ports: ethernet 3/1 ethernet 3/12 to 3/13 ethernet 3/20
Bytes received: 6000
```

show vlan ethernet

Syntax: **show vlan ethernet** *slot/port*

This command, with a *slot/port* entry, displays VLAN information for the specified port, as shown in the following example.

```
Brocade# show vlan ethernet 4/1
Port 4/1 is a member of 2 VLANs
VLANs 1 100
```

show vlan detail

Syntax: **show vlan detail** [*vlan-id*]

The *vlan-id* variable displays information about a specific VLAN.

This command displays detailed information about VLAN states, port types, and port modes, as well as control protocols configured on the VLAN, as shown in the following example.

```
Brocade# show vlan detail
Untagged Ports: ethernet 2/1 to 2/20 ethernet 4/4
Tagged Ports: None
Dual-mode Ports: ethernet 3/1 to 3/20jjj ethernet 4/1 to 4/3
Default VLAN: 1
Control VLAN: 4095
VLAN Tag-type: 0x8100
AVP Count       : 24
EVP Count       : 28
PORT-VLAN 1, Name DEFAULT-VLAN, Priority Level0
----------------------------------------------------------
Port   Type       Tag-Mode   Protocol   State
2/1    PHYSICAL   UNTAGGED   NON        DISABLED
2/2    PHYSICAL   UNTAGGED   NONE       DISABLED
2/3    PHYSICAL   UNTAGGED   NONE       DISABLED
2/4    PHYSICAL   UNTAGGED   NONE       DISABLED
2/5    PHYSICAL   UNTAGGED   NONE       DISABLED
.
. (output edited for brevity)
.
4/1    PHYSICAL   UNTAGGED   NONE       FORWARDING
4/2    PHYSICAL   UNTAGGED   NONE       FORWARDING
4/3    PHYSICAL   UNTAGGED   NONE       FORWARDING
4/4    PHYSICAL   UNTAGGED   NONE       DISABLED
PORT-VLAN 100, Name [None], Priority Level0
----------------------------------------------------------
Port   Type       Tag-Mode   Protocol   State
4/1    PHYSICAL   TAGGED     STP        FORWARDING
```

This command displays detailed information for a specific VLAN ID, as shown in the following example.

```
Brocade(config-vlan-10)# show vlan detail 10
PORT-VLAN 10, Name [None], Priority Level0
----------------------------------------------------------
Port  Type      Tag-Mode  Protocol  State
1/1   PHYSICAL  TAGGED    STP       DISABLED
1/2   PHYSICAL  TAGGED    STP       DISABLED
```

SAV debug commands

There are no debug commands specific to SAV.

Configuration notes

- A maximum of 1544 bytes is supported on ports where SAVs are configured. An additional 8 bytes over the untagged port maximum allows for support of two VLAN tags.
- For core devices, you must configure a VLAN tag-type (tag ID) that is different than the tag-type used on 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.

Common diagnostic scenarios

In an environment that includes a Brocade NetIron XMR and Brocade MLX series device with both VPLS and SAV configured, one VPLS endpoint port is connected to a carrier and will be receiving dual tags, with both tags using tag-type 8100. The outer tag is the carrier's VLAN and the inner tag is from the end user.

The frame structure is DA | SA | 8100 tag=1000 | 8100 tag=100 | Data from the carrier side. The port must support multiple VLANs coming in from the carrier.

The other VPLS endpoint goes to the end user with a single tag, and the frame structure is DA | SA | 8100 tag=100 | Data.

Use the following configuration on the two endpoints to support dual tags on one side of the VPLS and a single tag on the other side:

Carrier side endpoint configuration

Router MPLS

vpls vpls1000 1000

vpls-peer 10.2.2.2

vlan 1000

tagged ethe 3/1

End user side endpoint configuration

[Global config]

tag-type 9100 eth 3/1

Router MPLS

vpls vpls1000 1000

vpls-peer 10.1.1.1

vlan 100

untagged ethe 3/1

This configuration works in the following way:

Packets will ingress the carrier endpoint with dual tags, both with tag-type 8100. The VLAN ID 1000 will be stripped off the outer tag, and the inner tag customer VLAN ID 100 will be sent over the VPLS as payload. The packet will then egress the end user endpoint with a single tag-type 8100 VLAN ID of 100.

Packets will ingress the end user endpoint with a single VLAN tag, tag-type 8100, VLAN ID 100. The port is configured with SAV tag-type 9100 and is also configured as untagged, so the packets will be accepted and the VLAN ID 100 will be sent over the VPLS as payload. The packet will then egress the carrier endpoint tagged interface, where it will add the outer tag with tag-type 8100 VLAN ID 1000.

ERP

Ethernet Ring Protection (ERP) is a non-proprietary protocol that integrates an Automatic Protection Switching (APS) protocol and protection switching mechanisms to provide Layer 2 loop avoidance and fast reconvergence in Layer 2 ring topologies. ERP supports multi-ring and ladder topologies. ERP can also function with IEEE 802.1ag to support link monitoring when non-participating devices exist within the Ethernet ring.

ERP show commands

This section describes the show commands that display ERP information.

show erp

Syntax: **show erp** [*enter* | *erp_id*]

- *enter* - Specifies that you must press the Enter key (carriage return) after the **show erp** command to view ERP information for all links.
- *erp_id* - Displays ERP information for the specified ERP ID.

This command displays ERP statistics. Command output resembles the following example.

```
Brocade# show erp 7
ERP 7(version 2)- VLAN 504
==================================================================
Erp   ID   Status   Oper   Node   Topo
                    state  role   group
       1    enabled  Idle   rpl-owner -

Ring type   WTR        WTB      Guard    Holdoff   Msg
            time(min)  time(ms) time(ms) time(ms)  intv(ms)
Major-ring  5          7000     2000     0        1000

I/F    Port ERP port state     Interface status Interface type
L       1/12 blocking           normal           rpl
R       1/11 forwarding         normal           non-rpl
RAPS sent RAPS rcvd RAPS dropped RAPS ignored Oper state changes
3         3         0            0            0
```

show erp history-log

Syntax: **show erp history-log** [**brief** | *erp_id*]

- **brief** - Displays only the last 30 debug logs stored for ERP.
- *erp_id* - Specifies the ERP ID for which you want to display debug logs.

This command displays the debug logs for all the ERP instances. Command output resembles the following example.

```
Brocade# show erp history-log
timestamp       instance  slot/port prot_state    port_state trigger
Aug 25 09:18:51 5         1/5       protection    DISABLED   ERP_PL_REMOTE_FS
Aug 25 09:20:11 5         1/5       manual switch BLOCKING   ERP_PL_LOCAL_CLEAR_S
Aug 25 09:21:01 4         1/3       manual switch BLOCKING   ERP_PL_LOCAL_CLEAR_S
```

MRP

Metro Ring Protocol (MRP) is a proprietary protocol that prevents Layer 2 loops and provides fast reconvergence in Layer 2 ring topologies. MRP is especially useful in Metropolitan Area Networks (MANs), which typically require more flexibility than Spanning Tree Protocol (STP) can deliver.

This section describes how to use debug commands to monitor MRP environments for the Brocade NetIron XMR and Brocade MLX series routers.

Using MRP diagnostics

When you enable MRP diagnostics, the software tracks Ring Health Packets (RHPs) according to their sequence numbers, and calculates how long it takes an RHP to travel once through the entire ring. When you display the diagnostics, the CLI shows the average round-trip time for the RHPs 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 (this command is valid only on the master node).

diagnostics

Syntax: **diagnostics**

MRP show commands

This section describes the show command that displays MRP information.

show metro-ring

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

This command displays MRP diagnostics results. In the following example, the results are for metro ring 1.

```
Brocade# show metro-ring 1 diagnostics
Metro Ring 1
==============
diagnostics results

Ring       Diag       RHP average     Recommended     Recommended
id         state      time(microsec)  hello time(ms)  Prefwing time(ms)

1          enabled    < 126           100             300
Diag frame sent      Diag frame lost
6                           0
```

show metro-ring history-log

Syntax: **show metro-ring history-log** [**brief** | *mrp_id*]

- **brief** - Displays only the last 30 debug logs stored for MRP.
- *mrp_id* - Specifies the MRP ID for which you want to display debug logs.

This command displays the debug logs for all the MRP instances. Command output resembles the following example.

```
Brocade# show metro-ring history-log
timestamp       instance slot/port prot_state port_state trigger
Aug 25 09:18:51 5        1/5       -          DISABLED   MRP_TRUNK_DOWN
Aug 25 09:20:11 5        1/5       -          BLOCKING   MRP_REDUNDANT_PORT_UP
Aug 25 09:21:01 4        1/3       -          FRWDING  EMRP_TRUNK_PRIMARY_CHANGE
```

MRP debug commands

This section describes the MRP-related debug commands.

debug mrp bpdu

Syntax: [no] debug mrp bpdu

When this command is enabled, you will see an error message whenever a bridge protocol data unit (BPDU) is lost on the master node (not shown for member nodes).

Command output resembles the following example.

```
Brocade# debug mrp bpdu
        bpdu:  debugging is on
Dec 10 17:24:35 mrprhp_receive: receiving packet on port 6/2 old input port 6/2,
vlan VLAN: 22  for unconfigured ring 151 to add session
Dec 10 17:24:35 mrprhp_receive: receiving packet on port 6/2 old input port 6/2,
vlan VLAN: 22  for unconfigured ring 151 to add session
Dec 10 17:24:35 mrprhp_receive: receiving packet on port 6/2 old input port 6/2,
vlan VLAN: 22  for unconfigured ring 151 to add session
Dec 10 17:24:35 mrprhp_receive: receiving packet on port 6/2 old input port 6/2,
vlan VLAN: 22  for unconfigured ring 151 to add session
Dec 10 17:24:35 mrprhp_receive: receiving packet on port 6/2 old input port 6/2,
vlan VLAN: 22  for unconfigured ring 151 to add session
Dec 10 17:24:35 mrprhp_receive: receiving packet on port 6/2 old input port 6/2,
vlan VLAN: 22  for unconfigured ring 151 to add session
Dec 10 17:24:36 mrprhp_receive: receiving packet on port 6/2 old input port 6/2,
vlan VLAN: 22  for unconfigured ring 151 to add session
Dec 10 17:24:36 mrprhp_receive: receiving packet on port 6/2 old input port 6/2,
vlan VLAN: 22  for unconfigured ring 151 to add session
Dec 10 17:24:36 mrprhp_receive: receiving packet on port 6/2 old input port 6/2,
vlan VLAN: 22  for unconfigured ring 151 to add session
Dec 10 17:24:36 mrprhp_receive: receiving packet on port 6/2 old input port 6/2,
vlan VLAN: 22  for unconfigured ring 151 to add session
Dec 10 17:24:36 mrprhp_receive: receiving packet on port 6/2 old input port 6/2,
vlan VLAN: 22  for unconfigured ring 151 to add session
Dec 10 17:24:36 mrprhp_receive: receiving packet on port 6/2 old input port 6/2,
vlan VLAN: 22  for unconfigured ring 151 to add session
```

debug mrp diagnostics

Syntax: [no] debug mrp diagnostics

This command displays MRP diagnostic information. To activate diagnostic reporting, first enable MRP diagnostics debugging, then display the diagnostic information using the **show debug** command.

Command output resembles the following example.

```
Brocade# debug mrp diagnostics
        diags:  debugging is on
Dec 10 17:29:40 mrp-debug: mrpdiags_receive_packet. rpdu with sequence number
53102 has been lost. Reseting timers
```

debug mrp event

Syntax: [no] debug mrp event

This command displays information about MRP events, as shown in the following example.

```
Brocade# debug mrp event
        event: debugging is on

Apr 13 19:05:22 mrp-debug: **state PREFORWARDING for port 2/1 in ring 1 **
Apr 13 19:05:22 mrp-debug: ** state FORWARDING for port 2/1 in ring 1 **
Apr 13 19:05:22 mrpinfo - port 2/1, up 0
Apr 13 19:05:22 mrp-debug: ** state DISABLED for port 2/1 in ring 1 **
Apr 13 19:05:28 mrpinfo - port 2/1, up 1
Apr 13 19:05:28 mrp-debug: ** state BLOCKING for port 2/1 in ring 1 **
Apr 13 19:05:29 mrp-debug: mrpdiags_receive_packet. rpdu with sequence number
18
184 has been lost. Resetting timers
```

Configuration notes

MRP can be enabled on port-based VLANs, but cannot be enabled or disabled on protocol-based VLANs.

Spanning Tree Protocol and derivatives

The following sections describe diagnostic procedures for Spanning Tree Protocol (STP) and STP derivatives, including SSTP, MSTP, RSTP, and SuperSpan.

NOTE
Layer 2 protocols such as STP, RSTP, MRP, and VSRP can be enabled on port-based VLANs, but cannot be enabled or disabled on protocol-based VLANs.

STP

A control protocol, such as Spanning Tree Protocol (STP), can block one or more ports in a protocol-based VLAN that uses a virtual routing interface to route to other VLANs. For IP VLANs 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 as long as at least one port in the protocol-based VLAN is not blocked by STP.

SSTP

For VLANs where Single Spanning Tree Protocol (SSTP) is enabled, the ports become members of a single spanning tree. For VLANs where SSTP is disabled, ports are excluded from the single spanning tree. VLANs can be selectively added or removed from the single spanning tree domain as well.

RSTP

Rapid Spanning Tree Protocol (RSTP) provides rapid traffic reconvergence for point-to-point links within a few milliseconds (less than 500 milliseconds) following the failure of a bridge or bridge port. This reconvergence occurs more rapidly than that provided by STP because convergence in RSTP bridges is based on the explicit handshakes between designated ports and their connected root ports rather than on timer values.

MSTP

With Multiple Spanning Tree Protocol (MSTP), the entire network runs a common instance of RSTP. Within the common instance, one or more VLANs can be individually configured into distinct regions. The entire network runs the Common Spanning Tree (CST) instance and the regions run a local instance, or Internal Spanning Tree (IST). Because the CST treats each IST as a single bridge, ports are blocked to prevent loops that might occur within an IST and also throughout the CST. In addition, MSTP can co-exist with individual devices running STP or RSTP in the Common and Internal Spanning Tree instance (CIST). With the exception of the provisions for multiple instances, MSTP operates exactly like RSTP.

SuperSpan

SuperSpan is an STP enhancement that allows service providers (SPs) to use STP in both SP networks and customer networks. The Brocade NetIron XMR and Brocade MLX series devices are configured to tunnel each customer's STP BPDUs through the SP. From the customer's perspective, the SP network is a loop-free non-blocking device or network. The SP network behaves like a hub in the sense that the necessary blocking occurs in the customer network, not in the SP network.

STP show commands

This section describes the show command that displays STP information.

show spanning-tree

Syntax: **show spanning-tree** [**detail** [**vlan** *vlan-id* [**ethernet** *slot/port*]]| **protect** | **pvst-mode** | **root-protect** | **vlan** *vlan id*]

- **detail** - Displays detailed STP information.
- **vlan** *vlan-id* - Specifies a VLAN for detailed STP information.
- **ethernet** *slot/port* - Specifies an Ethernet port within the VLAN for detailed STP information.
- **protect** - Displays STP BPDU guard information.
- **pvst-mode** - Displays PVST information.
- **root-protect** - Displays STP root guard information.

- **vlan** *vlan id* - Displays STP information for the specified VLAN.

This command displays STP information, as shown in the following example (in this instance, for VLAN 10).

```
Brocade# show spanning-tree vlan 10
VLAN 10 - STP instance 1
--------------------------------------------------------------------
STP Bridge Parameters:
Bridge           Bridge Bridge Bridge Hold Last Topology Topology
Identifier       MaxAge Hello  FwdDly Time Change        Change
hex              sec    sec    sec    sec  sec           cnt
8000000000a04000 20     2      15     1    0             0
RootBridge       RootPath DesignatedBridge Root Max Hel Fwd
Identifier       Cost     Identifier       Port Age lo  Dly
hex                       hex              sec  sec     sec
8000000000a04000 0        8000000000a04000 Root 20  2   15
STP Port Parameters:
Port Prio Path State Designat- Designated       Designated
Num  rity Cost       ed Cost   Root             Bridge
1/3  128  4    DISABLED 0      0000000000000000 0000000000000000
1/13 128  4    DISABLED 0      0000000000000000 0000000000000000
```

show spanning-tree history-log

Syntax: show spanning-tree history-log [**brief** | **vlan** *vlan-id*]

- **brief** - Displays only the last 30 debug logs stored for STP.
- **vlan** *vlan-id* - Specifies the VLAN ID for which you want to display STP debug logs.

This command displays the STP debug logs for all the VLANs. Command output resembles the following example.

```
Brocade# show spanning-tree history-log
timestamp           PORT  VLAN ID    Event               Triggered starting API
==============================================================================
Sept  20 09:20:11    1/2   100        Superior BPDU recvd  stputil_process_bpdu
Sept  20 09:20:11    1/1   4095       MAX_ageTimer expiry  stptimer_update
Sept  20 11:18:40     1/3   4095      Superior BPDU recvd  stputil_process_bpdu
Sept  20 11:18:42    1/1   100        MAX_ageTimer expiry  stptimer_update
```

STP debug commands

This section describes the debug commands for STP, MSTP, and RSTP environments.

CAUTION

Enabling diagnostic commands may degrade system performance. These commands are best used to troubleshoot specific problems while working with qualified Brocade service technicians. Whenever possible, troubleshoot your system during periods of low network traffic and user activity to preserve system performance.

NOTE

Because STP and RSTP debug commands are virtually identical, only STP commands are described in detail in this chapter.

debug spanning-tree

Syntax: [**no**] **debug spanning-tree** [**config-bpdu** | **event** | **port** [**ethernet** *slot/port* | **pos** *slot/port*] | **reset** | **show** | **tcn-bpdu** | **verbose** | **vlan** *vlan id*]

This command generates information about all BPDUs and spanning tree events (by default) or specific events as defined by the variables shown in the syntax line. In many cases, generic debugging is not useful. If multiple STP instances are configured, it can be difficult to identify content for a specific instance from the extensive output that may be generated. Use the **debug spanning-tree port** and **debug spanning-tree vlan** commands to define specific instances for debugging.

- **config-bpdu** - Generates information about STP configuration bridge protocol data units (BPDUs).
- **event** - Generates information about STP non-BPDU events (timer, configuration, and so on).
- **port** - Restricts STP debugging to specific ports.
- **ethernet** *slot/port* - Restricts STP debugging to a specific Ethernet interface.
- **pos** *slot/port* - Restricts STP debugging to a specific POS interface.
- **reset** - Resets all STP debugging parameters to the default.
- **show** - Displays current STP debug settings.
- **tcn-bpdu** - Enables or disables STP TCN BPDU debugging.
- **verbose** - Enables or disables STP verbose debugging mode.
- **vlan** *vlan id* - Restricts STP debugging to specific VLAN.

This command generates information about all STP activity and events. Command output resembles the following example.

```
Brocade# debug spanning-tree
STP: debugging is on
STP: Sending Config BPDU - VLAN 3 Port 2/3
0000 00 00 00 800000000001c042 00000000
 800000000001c042 8043 0000 0000 0000 0000
STP: Sending Config BPDU - VLAN 3 Port 2/4
0000 00 00 00 800000000001c042 00000000
 800000000001c042 8044 0000 0000 0000 0000
STP: Sending Config BPDU - VLAN 2 Port 2/1
```

debug spanning-tree config-bpdu

Syntax: [**no**] **debug spanning-tree config-bpdu**

This command generates information about STP BPDUs. Command output resembles the following example.

```
Brocade# debug spanning-tree config-bpdu
STP: Received Config BPDU - VLAN 10 Port 3/20
        0000 00 00 01 80000000003d8500 00000000
        80000000003d8500 8050 0000 1400 0200 0f00
STP: Received Config BPDU - VLAN 10 Port 3/20
         0000 00 00 01 80000000003d8500 00000000
        80000000003d8500 8050 0000 1400 0200 0f00
STP: Received Config BPDU - VLAN 10 Port 3/20
        0000 00 00 01 80000000003d8500 00000000
```

```
        80000000003d8500 8050 0000 1400 0200 0f00
STP: Received Config BPDU - VLAN 10 Port 3/20
        0000 00 00 01 80000000003d8500 00000000
        80000000003d8500 8050 0000 1400 0200 0f00
```

debug spanning-tree event

Syntax: [**no**] **debug spanning-tree event**

This command displays information about non-BPDU events, such as timer, configuration, and so on. Command output resembles the following example.

```
Brocade# debug spanning-tree event
 STP: LISTENING - VLAN 10 Port 3/20
 STP: LEARNING - VLAN 10 Port 3/20
 STP: Sending TCN BPDU - VLAN 10 Port 3/20
 STP: FORWARDING - VLAN 10 Port 3/20
 STP: TCN ACK Received - VLAN 10 Port 3/20
```

debug spanning-tree port

Syntax: [**no**] **debug spanning-tree port** [**ethernet** *slot/port* | **pos** *slot/port*]

This command displays spanning tree information about a port for a specific interface. Command output resembles the following example.

```
Brocade# debug spanning-tree port ethernet 2/2
STP debugging turned on for ports ethe 2/2
55 STP: Sending Config BPDU - VLAN 2 Port 2/2
55 0000 00 00 00 800000000001c040 00000000
800000000001c040 8042 0000 0000 0000 0000
STP: Sending Config BPDU - VLAN 2 Port 2/2
0000 00 00 00 800000000001c040 00000000
800000000001c040 8042 0000 0000 0000 0000
STP: Sending Config BPDU - VLAN 2 Port 2/2
0000 00 00 00 800000000001c040 00000000
800000000001c040 8042 0000 0000 0000 0000
STP: Sending Config BPDU - VLAN 2 Port 2/2
0000 00 00 00 800000000001c040 00000000
800000000001c040 8042 0000 0000 0000 0000
```

debug spanning-tree reset

Syntax: [**no**] **debug spanning-tree reset**

This command resets all STP debugging parameters to the default. The default mode disables all STP debugs. The **show debug** command will no longer show STP debug as enabled. This command works in the same way as the **no debug spanning-tree** command.

debug spanning-tree show

Syntax: [**no**] **debug spanning-tree show**

This command generates information about the STP debug configuration. The following command output shows the default configuration.

```
Brocade# debug spanning-tree show
STP Debug Parameters
--------------------
STP debugging is ON [Mode: Brief]
NonBpduEvents ConfigBpduEvents TcnBpdusEvents are being tracked
Ports: All
VLANs: All
```

debug spanning-tree tcn-bpdu

Syntax: [**no**] **debug spanning-tree tcn-bpdu**

This command displays information about TCN BPDU events. Command output resembles the following example.

```
Brocade# debug spanning-tree tcn-bpdu
 STP: Sending TCN BPDU - VLAN 10 Port 3/20
 STP: TCN ACK Received - VLAN 10 Port 3/20
```

debug spanning-tree verbose

Syntax: [**no**] **debug spanning-tree verbose**

In verbose mode, STP BPDU information is translated into BPDU fields and values, which are easier to read than the default hex output. Command output resembles the following example.

```
Brocade# debug spanning-tree verbose

STP: Sending Config BPDU - VLAN 2 Port 2/2
         protocol-id: 0000
protocol-version: 00
type: 00
flags: 00
root-id: 800000000001c040
path-cost: 00000000
bridge-id: 800000000001c040
port-id: 8042
message-age: 0000
max-age: 0000
hello-time: 0000
hello-time: 0000
```

debug spanning-tree vlan

Syntax: [**no**] **debug spanning-tree vlan** *vlan id*

This command restricts debug output to a specific VLAN. Command output resembles the following example.

```
Brocade# debug spanning-tree vlan 2
STP: Sending Config BPDU - VLAN 2 Port 2/1
       0000 00 00 00 800000000001c040 00000000
```

MSTP show commands

This section describes the show command that displays MSTP information.

show mstp

Syntax: show mstp *instance-id* [**config** | **detail** | **region** *region-id*]

- *instance-id* - Specifies the ID of the MSTP instance.
- **config** - Displays MSTP configuration information.
- **detail** - Displays detailed MSTP information.
- **region** *region-id* - Displays information related to a particular MSTP Provider Bridge (PB) or Provider Backbone Bridging (PBB) region.

This command displays general MSTP information. Command output resembles the following example for MSTP over PBB operation.

```
Brocade# show mstp
Region 1:
-----------

MSTP Instance 0 (CIST) - VLAN Scope: None
----------------------------------------------------------------------------
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
8000000000af7800 20     2      15     20     20     2     15     20

Root             ExtPath   RegionalRoot     IntPath   Designated       Root
Bridge           Cost      Bridge           Cost      Bridge           Port
hex                        hex                        hex
8000000000af7800 0         8000000000af7800 0         8000000000af7800 Root

Port  Pri PortPath  P2P Edge Role       State      Designa-  Designated
Num       Cost      Mac Port                       ted cost  bridge
2/5   128 20000     F   F    DESIGNATED FORWARDING 0         8000000000af7800
3/5   128 20000     F   F    DESIGNATED FORWARDING 0         8000000000af7800

MSTP Instance 1 - VLANs: VPLS 1 VLAN 100
----------------------------------------------------------------------------
Bridge           Max RegionalRoot     IntPath   Designated       Root  Root
Identifier       Hop Bridge           Cost      Bridge           Port  Hop
hex              cnt hex                        hex                    cnt
8001001bedaf7800 20  8001001bedaf7800 0         8001001bedaf7800 Root  20

Port  Pri PortPath  P2P Edge Role       State      Designa-  Designated
Num       Cost      Mac Port                       ted cost  bridge
2/5   128 20000     F   F    DESIGNATED FORWARDING 0         8001001bedaf7800
3/5   128 20000     F   F    DESIGNATED FORWARDING 0         8001001bedaf7800

Region 2:
-----------

MSTP Instance 0 (CIST) - VLAN Scope: None
----------------------------------------------------------------------------
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
8000000000af7800 20     2      15     20     20     2     15     20

Root             ExtPath   RegionalRoot     IntPath   Designated       Root
Bridge           Cost      Bridge           Cost      Bridge           Port
hex                        hex                        hex
```

```
8000000000af7800 0         8000000000af7800 0         8000000000af7800 Root

Port  Pri PortPath  P2P Edge Role       State      Designa-  Designated
Num       Cost      Mac Port                       ted cost  bridge
2/5   128 20000     F   F    DESIGNATED FORWARDING 0         8000000000af7800
3/5   128 20000     F   F    DESIGNATED FORWARDING 0         8000000000af7800

MSTP Instance 1 - VLANs: VPLS 1 VLAN 200
----------------------------------------------------------------------------
Bridge           Max RegionalRoot     IntPath   Designated       Root  Root
Identifier       Hop Bridge           Cost      Bridge           Port  Hop
hex              cnt hex                        hex                    cnt
8001001bedaf7800 20  8001001bedaf7800 0         8001001bedaf7800 Root  20

Port  Pri PortPath  P2P Edge Role       State      Designa-  Designated
Num       Cost      Mac Port                       ted cost  bridge
2/5   128 20000     F   F    DESIGNATED FORWARDING 0         8001001bedaf7800
3/5   128 20000     F   F    DESIGNATED FORWARDING 0         8001001bedaf7800
```

show mstp config

Syntax: show mstp config

This command displays MSTP configuration information. Command output resembles the following example for MSTP over PBB operation.

```
Brocade# show mstp config
Mstp-region 1
Mstp-region name PBB-Domain
Mstp-region revision 1
Mstp-region instance 1 vpls 1 vlan 100
Mstp-region start

Mstp-region 2
Mstp-region name PB-Domain1
Mstp-region revision 1
Mstp-region instance 1 vpls 1 vlan 200
Mstp-region start
```

show mstp detail

Syntax: show mstp detail

This command displays detailed MSTP information. Command output resembles the following example for MSTP over PBB operation.

```
Brocade# show mstp detail
Region 10
--------

MSTP Instance 0 (CIST) - VLAN Scope: None
----------------------------------------------------------------------------
  Bridge: 8000000000198800 [Priority 32768, SysId 0, Mac 001bed198800]
  FwdDelay 15, HelloTime 2, MaxHops 20, TxHoldCount 6

Port 1/11 - Role: DESIGNATED - State: FORWARDING
  PathCost 20000, Priority 128, OperEdge F, OperPt2PtMac F, Boundary T
  Designated   - Root 8000000000034400, RegionalRoot 8000000000034400,
                 Bridge 8000000000198800, ExtCost 0, IntCost 2000
  ActiveTimers - helloWhen 2
```

```
  MachineState - PRX-RECEIVE, PTX-IDLE, PPM-SENDING_RSTP, PIM-CURRENT
                 PRT-ACTIVE_PORT, PST-FORWARDING, TCM-ACTIVE
  BPDUs        - Rcvd MST 132, RST 0, Config 0, TCN 0
                 Sent MST 3262, RST 0, Config 0, TCN 0

Port 1/12 - Role: DESIGNATED - State: FORWARDING
  PathCost 20000, Priority 128, OperEdge F, OperPt2PtMac F, Boundary T
  Designated   - Root 8000000000034400, RegionalRoot 8000001bed034400,
                 Bridge 8000000000198800, ExtCost 0, IntCost 2000
  ActiveTimers - helloWhen 2
  MachineState - PRX-RECEIVE, PTX-IDLE, PPM-SENDING_RSTP, PIM-CURRENT
                 PRT-ACTIVE_PORT, PST-FORWARDING, TCM-ACTIVE
  BPDUs        - Rcvd MST 132, RST 0, Config 0, TCN 0
                 Sent MST 3262, RST 0, Config 0, TCN 0

Port 1/13 - Role: DESIGNATED - State: FORWARDING
  PathCost 20000, Priority 128, OperEdge F, OperPt2PtMac F, Boundary T
  Designated   - Root 8000000000034400, RegionalRoot 8000000000034400,
                 Bridge 8000000000198800, ExtCost 0, IntCost 2000
  ActiveTimers - helloWhen 1
  MachineState - PRX-RECEIVE, PTX-IDLE, PPM-SENDING_RSTP, PIM-CURRENT
                 PRT-ACTIVE_PORT, PST-FORWARDING, TCM-ACTIVE
  BPDUs        - Rcvd MST 132, RST 0, Config 0, TCN 0
                 Sent MST 3262, RST 0, Config 0, TCN 0

Port 1/14 - Role: DESIGNATED - State: FORWARDING
  PathCost 20000, Priority 128, OperEdge F, OperPt2PtMac F, Boundary T
  Designated   - Root 8000000000034400, RegionalRoot 8000000000034400,
                 Bridge 8000000000198800, ExtCost 0, IntCost 2000
  ActiveTimers - helloWhen 1
  MachineState - PRX-RECEIVE, PTX-IDLE, PPM-SENDING_RSTP, PIM-CURRENT
                 PRT-ACTIVE_PORT, PST-FORWARDING, TCM-ACTIVE
  BPDUs        - Rcvd MST 132, RST 0, Config 0, TCN 0
                 Sent MST 3262, RST 0, Config 0, TCN 0

Port 1/15 - Role: DESIGNATED - State: FORWARDING
  PathCost 20000, Priority 128, OperEdge F, OperPt2PtMac F, Boundary T
  Designated   - Root 8000000000034400, RegionalRoot 8000000000034400,
                 Bridge 8000000000198800, ExtCost 0, IntCost 2000
  ActiveTimers - helloWhen 1
  MachineState - PRX-RECEIVE, PTX-IDLE, PPM-SENDING_RSTP, PIM-CURRENT
                 PRT-ACTIVE_PORT, PST-FORWARDING, TCM-ACTIVE
  BPDUs        - Rcvd MST 132, RST 0, Config 0, TCN 0
                 Sent MST 3262, RST 0, Config 0, TCN 0

Port 1/16 - Role: DESIGNATED - State: FORWARDING
  PathCost 20000, Priority 128, OperEdge F, OperPt2PtMac F, Boundary T
  Designated   - Root 8000000000034400, RegionalRoot 8000000000034400,
                 Bridge 8000000000198800, ExtCost 0, IntCost 2000
  ActiveTimers - helloWhen 1
  MachineState - PRX-RECEIVE, PTX-IDLE, PPM-SENDING_RSTP, PIM-CURRENT
                 PRT-ACTIVE_PORT, PST-FORWARDING, TCM-ACTIVE
  BPDUs        - Rcvd MST 133, RST 0, Config 0, TCN 0
                 Sent MST 3262, RST 0, Config 0, TCN 0

Port 1/17 - Role: DESIGNATED - State: FORWARDING
  PathCost 20000, Priority 128, OperEdge F, OperPt2PtMac F, Boundary T
  Designated   - Root 8000000000034400, RegionalRoot 8000000000034400,
                 Bridge 8000000000198800, ExtCost 0, IntCost 2000
  ActiveTimers - helloWhen 1
```

```
  MachineState - PRX-RECEIVE, PTX-IDLE, PPM-SENDING_RSTP, PIM-CURRENT
                 PRT-ACTIVE_PORT, PST-FORWARDING, TCM-ACTIVE
  BPDUs        - Rcvd MST 133, RST 0, Config 0, TCN 0
                 Sent MST 3262, RST 0, Config 0, TCN 0

Port 1/18 - Role: DESIGNATED - State: FORWARDING
  PathCost 20000, Priority 128, OperEdge F, OperPt2PtMac F, Boundary T
  Designated   - Root 8000000000034400, RegionalRoot 8000000000034400,
                 Bridge 8000000000198800, ExtCost 0, IntCost 2000
  ActiveTimers - helloWhen 1
  MachineState - PRX-RECEIVE, PTX-IDLE, PPM-SENDING_RSTP, PIM-CURRENT
                 PRT-ACTIVE_PORT, PST-FORWARDING, TCM-ACTIVE
  BPDUs        - Rcvd MST 133, RST 0, Config 0, TCN 0
                 Sent MST 3262, RST 0, Config 0, TCN 0

Port 1/19 - Role: DESIGNATED - State: FORWARDING
  PathCost 20000, Priority 128, OperEdge F, OperPt2PtMac F, Boundary T
  Designated   - Root 8000000000034400, RegionalRoot 8000000000034400,
                 Bridge 8000000000198800, ExtCost 0, IntCost 2000
  ActiveTimers - helloWhen 1
  MachineState - PRX-RECEIVE, PTX-IDLE, PPM-SENDING_RSTP, PIM-CURRENT
                 PRT-ACTIVE_PORT, PST-FORWARDING, TCM-ACTIVE
  BPDUs        - Rcvd MST 133, RST 0, Config 0, TCN 0
                 Sent MST 3262, RST 0, Config 0, TCN 0

Port 1/20 - Role: DESIGNATED - State: FORWARDING
  PathCost 20000, Priority 128, OperEdge F, OperPt2PtMac F, Boundary T
  Designated   - Root 8000000000034400, RegionalRoot 8000000000034400,
                 Bridge 8000000000198800, ExtCost 0, IntCost 2000
  ActiveTimers - helloWhen 1
  MachineState - PRX-RECEIVE, PTX-IDLE, PPM-SENDING_RSTP, PIM-CURRENT
                 PRT-ACTIVE_PORT, PST-FORWARDING, TCM-ACTIVE
  BPDUs        - Rcvd MST 133, RST 0, Config 0, TCN 0
                 Sent MST 3262, RST 0, Config 0, TCN 0

Port 3/1 - Role: ROOT - State: FORWARDING
  PathCost 2000, Priority 128, OperEdge F, OperPt2PtMac F, Boundary F
  Designated   - Root 8000000000034400, RegionalRoot 8000000000034400,
                 Bridge 8000000000034400, ExtCost 0, IntCost 0
  ActiveTimers - helloWhen 1 rrWhile 15 rcvdInfoWhile 5
  MachineState - PRX-RECEIVE, PTX-IDLE, PPM-SENDING_RSTP, PIM-CURRENT
                 PRT-ACTIVE_PORT, PST-FORWARDING, TCM-ACTIVE
  BPDUs        - Rcvd MST 1723, RST 0, Config 0, TCN 0
                 Sent MST 1952, RST 0, Config 0, TCN 0

Port 3/2 - Role: ROOT - State: FORWARDING
  PathCost 2000, Priority 128, OperEdge F, OperPt2PtMac F, Boundary F
  Designated   - Root 8000000000034400, RegionalRoot 8000000000034400,
                 Bridge 8000000000034400, ExtCost 0, IntCost 0
  ActiveTimers - helloWhen 1 rrWhile 15 rcvdInfoWhile 5
  MachineState - PRX-RECEIVE, PTX-IDLE, PPM-SENDING_RSTP, PIM-CURRENT
                 PRT-ACTIVE_PORT, PST-FORWARDING, TCM-ACTIVE
  BPDUs        - Rcvd MST 1723, RST 0, Config 0, TCN 0
                 Sent MST 1952, RST 0, Config 0, TCN 0

Port 3/5 - Role: ROOT - State: FORWARDING
  PathCost 2000, Priority 128, OperEdge F, OperPt2PtMac F, Boundary F
  Designated   - Root 8000000000034400, RegionalRoot 8000000000034400,
                 Bridge 8000000000034400, ExtCost 0, IntCost 0
  ActiveTimers - helloWhen 1 rrWhile 15 rcvdInfoWhile 5
```

```
  MachineState - PRX-RECEIVE, PTX-IDLE, PPM-SENDING_RSTP, PIM-CURRENT
                 PRT-ACTIVE_PORT, PST-FORWARDING, TCM-ACTIVE
  BPDUs        - Rcvd MST 1723, RST 0, Config 0, TCN 0
                 Sent MST 1952, RST 0, Config 0, TCN 0

Port 3/6 - Role: ROOT - State: FORWARDING
  PathCost 2000, Priority 128, OperEdge F, OperPt2PtMac F, Boundary F
  Designated   - Root 8000000000034400, RegionalRoot 8000000000034400,
                 Bridge 8000000000034400, ExtCost 0, IntCost 0
  ActiveTimers - helloWhen 1 rrWhile 15 rcvdInfoWhile 5
  MachineState - PRX-RECEIVE, PTX-IDLE, PPM-SENDING_RSTP, PIM-CURRENT
                 PRT-ACTIVE_PORT, PST-FORWARDING, TCM-ACTIVE
  BPDUs        - Rcvd MST 1723, RST 0, Config 0, TCN 0
                 Sent MST 1952, RST 0, Config 0, TCN 0

Port 5/1 - Role: DESIGNATED - State: FORWARDING
  PathCost 2000, Priority 128, OperEdge F, OperPt2PtMac F, Boundary F
  Designated   - Root 8000000000034400, RegionalRoot 8000000000034400,
                 Bridge 8000000000198800, ExtCost 0, IntCost 2000
  ActiveTimers - helloWhen 1
  MachineState - PRX-RECEIVE, PTX-IDLE, PPM-SENDING_RSTP, PIM-CURRENT
                 PRT-ACTIVE_PORT, PST-FORWARDING, TCM-ACTIVE
  BPDUs        - Rcvd MST 10024, RST 0, Config 0, TCN 0
                 Sent MST 9871, RST 0, Config 0, TCN 0

Port 5/2 - Role: DESIGNATED - State: FORWARDING
  PathCost 2000, Priority 128, OperEdge F, OperPt2PtMac F, Boundary F
  Designated   - Root 8000000000034400, RegionalRoot 8000000000034400,
                 Bridge 8000000000198800, ExtCost 0, IntCost 2000
  ActiveTimers - helloWhen 1
  MachineState - PRX-RECEIVE, PTX-IDLE, PPM-SENDING_RSTP, PIM-CURRENT
                 PRT-ACTIVE_PORT, PST-FORWARDING, TCM-ACTIVE
  BPDUs        - Rcvd MST 10024, RST 0, Config 0, TCN 0
                 Sent MST 9871, RST 0, Config 0, TCN 0

Port 5/3 - Role: DESIGNATED - State: FORWARDING
  PathCost 2000, Priority 128, OperEdge F, OperPt2PtMac F, Boundary F
  Designated   - Root 8000000000034400, RegionalRoot 8000000000034400,
                 Bridge 8000000000198800, ExtCost 0, IntCost 2000
  ActiveTimers - helloWhen 1
  MachineState - PRX-RECEIVE, PTX-IDLE, PPM-SENDING_RSTP, PIM-CURRENT
                 PRT-ACTIVE_PORT, PST-FORWARDING, TCM-ACTIVE
  BPDUs        - Rcvd MST 10024, RST 0, Config 0, TCN 0
                 Sent MST 9871, RST 0, Config 0, TCN 0

Port 5/4 - Role: DESIGNATED - State: FORWARDING
  PathCost 2000, Priority 128, OperEdge F, OperPt2PtMac F, Boundary F
  Designated   - Root 8000000000034400, RegionalRoot 8000000000034400,
                 Bridge 8000000000198800, ExtCost 0, IntCost 2000
  ActiveTimers - helloWhen 1
  MachineState - PRX-RECEIVE, PTX-IDLE, PPM-SENDING_RSTP, PIM-CURRENT
                 PRT-ACTIVE_PORT, PST-FORWARDING, TCM-ACTIVE
  BPDUs        - Rcvd MST 10024, RST 0, Config 0, TCN 0
                 Sent MST 9871, RST 0, Config 0, TCN 0

MSTP Instance 1 - VLANs: 801

----------------------------------------------------------------------------
  Bridge: 8001001bed198800 [Priority 32768, SysId 1, Mac 001bed198800]
```

```
Port 1/11 - Role: DESIGNATED - State: FORWARDING
  PathCost 20000, Priority 128
  Designated   - RegionalRoot 8001001bed034400, IntCost 2000
                 Bridge 8001001bed198800
  ActiveTimers -
  MachineState - PIM-CURRENT, PRT-ACTIVE_PORT, PST-FORWARDING, TCM-ACTIVE

Port 1/12 - Role: DESIGNATED - State: FORWARDING
  PathCost 20000, Priority 128
  Designated   - RegionalRoot 8001001bed034400, IntCost 2000
                 Bridge 8001001bed198800
  ActiveTimers -
  MachineState - PIM-CURRENT, PRT-ACTIVE_PORT, PST-FORWARDING, TCM-ACTIVE

Port 1/13 - Role: DESIGNATED - State: FORWARDING
  PathCost 20000, Priority 128
  Designated   - RegionalRoot 8001001bed034400, IntCost 2000
                 Bridge 8001001bed198800
  ActiveTimers -
  MachineState - PIM-CURRENT, PRT-ACTIVE_PORT, PST-FORWARDING, TCM-ACTIVE
```

show mstp region

Syntax: **show mstp region** *region-id*

The *region-id* variable specifies the ID of the configured MSTP region.

This command displays information related to a particular MSTP PB or PBB region. Command output resembles the following example for MSTP over PBB operation.

```
Brocade# show mstp region 10
Region 10
--------

MSTP Instance 0 (CIST) - VLAN Scope: None
----------------------------------------------------------------------------
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
8000000000198800 20     2      15     20     20     2     15     19

Root             ExtPath   RegionalRoot     IntPath   Designated       Root
Bridge           Cost      Bridge           Cost      Bridge           Port
hex                        hex                        hex
8000000000034400 0         8000000000034400 2000      8000000000034400 3/1

Port  Pri PortPath  P2P Edge Role       State      Designa-  Designated
Num       Cost      Mac Port                       ted cost  bridge
1/11  128 20000     F   F    DESIGNATED FORWARDING 0         8000000000198800
1/12  128 20000     F   F    DESIGNATED FORWARDING 0         8000000000198800
1/13  128 20000     F   F    DESIGNATED FORWARDING 0         8000000000198800
1/14  128 20000     F   F    DESIGNATED FORWARDING 0         8000000000198800
1/15  128 20000     F   F    DESIGNATED FORWARDING 0         8000000000198800
1/16  128 20000     F   F    DESIGNATED FORWARDING 0         8000000000198800
1/17  128 20000     F   F    DESIGNATED FORWARDING 0         8000000000198800
1/18  128 20000     F   F    DESIGNATED FORWARDING 0         8000000000198800
1/19  128 20000     F   F    DESIGNATED FORWARDING 0         8000000000198800
1/20  128 20000     F   F    DESIGNATED FORWARDING 0         8000000000198800
3/1   128 2000      F   F    ROOT       FORWARDING 0         8000000000034400
3/2   128 2000      F   F    ROOT       FORWARDING 0         8000000000034400
```

```
3/5   128 2000      F   F    ROOT       FORWARDING 0         8000000000034400
3/6   128 2000      F   F    ROOT       FORWARDING 0         8000000000034400
5/1   128 2000      F   F    DESIGNATED FORWARDING 0         8000000000198800
5/2   128 2000      F   F    DESIGNATED FORWARDING 0         8000000000198800
5/3   128 2000      F   F    DESIGNATED FORWARDING 0         8000000000198800
5/4   128 2000      F   F    DESIGNATED FORWARDING 0         8000000000198800

MSTP Instance 1 - VLANs: 801

----------------------------------------------------------------------------
Bridge           Max RegionalRoot     IntPath   Designated       Root  Root
Identifier       Hop Bridge           Cost      Bridge           Port  Hop
hex              cnt hex                        hex                    cnt
8001001bed198800 20  8001001bed034400 2000      8001001bed034400 3/1   19

Port  Pri PortPath  P2P Edge Role       State      Designa-  Designated
Num       Cost      Mac Port                       ted cost  bridge
1/11  128 20000     F   F    DESIGNATED FORWARDING 2000      8001001bed198800
1/12  128 20000     F   F    DESIGNATED FORWARDING 2000      8001001bed198800
1/13  128 20000     F   F    DESIGNATED FORWARDING 2000      8001001bed198800
1/14  128 20000     F   F    DESIGNATED FORWARDING 2000      8001001bed198800
1/15  128 20000     F   F    DESIGNATED FORWARDING 2000      8001001bed198800
1/16  128 20000     F   F    DESIGNATED FORWARDING 2000      8001001bed198800
1/17  128 20000     F   F    DESIGNATED FORWARDING 2000      8001001bed198800
1/18  128 20000     F   F    DESIGNATED FORWARDING 2000      8001001bed198800
1/19  128 20000     F   F    DESIGNATED FORWARDING 2000      8001001bed198800
1/20  128 20000     F   F    DESIGNATED FORWARDING 2000      8001001bed198800
3/1   128 2000      F   F    ROOT       FORWARDING 0         8001001bed034400
3/2   128 2000      F   F    ROOT       FORWARDING 0         8001001bed034400
3/5   128 2000      F   F    ROOT       FORWARDING 0         8001001bed034400
3/6   128 2000      F   F    ROOT       FORWARDING 0         8001001bed034400
5/1   128 2000      F   F    DESIGNATED FORWARDING 2000      8001001bed198800
5/2   128 2000      F   F    DESIGNATED FORWARDING 2000      8001001bed198800
5/3   128 2000      F   F    DESIGNATED FORWARDING 2000      8001001bed198800
5/4   128 2000      F   F    DESIGNATED FORWARDING 2000      8001001bed198800
```

show mstp history-log

Syntax: show mstp history-log [**brief** | **region** region-*id*]

- **brief** - Displays only the last 30 debug logs stored for MSTP.
- **region** region-*id* - Displays the MSTP debug logs for the specified region ID.

The **show mstp history-log brief** command displays only the last 30 debug logs stored for MSTP. Command output resembles the following example.

```
Brocade# show mstp history-log brief
Timestamp       RegId Port  MSTID Event            Triggered API
-------------------------------------------------------------------------------
Jun  7 00:29:24 10    5/1   1     tcWhile Xprd     mstptimer_update
Jun  7 00:29:24 10    5/1   0     tcWhile Xprd     mstptimer_update
Jun  7 00:29:24 10    1/11  1     tcWhile Xprd     mstptimer_update
Jun  7 00:29:24 10    1/11  0     tcWhile Xprd     mstptimer_update
Jun  7 00:28:48 10    5/1   1     tcWhile Xprd     mstptimer_update
Jun  7 00:28:48 10    5/1   0     tcWhile Xprd     mstptimer_update
Jun  7 00:28:48 10    1/11  1     tcWhile Xprd     mstptimer_update
Jun  7 00:28:48 10    1/11  0     tcWhile Xprd     mstptimer_update
Jun  7 00:28:25 10    5/1   0     Rx Pkt Delay- MP mstpmgr_rx_bpdu()
Jun  7 00:28:13 10    5/1   1     tcWhile Xprd     mstptimer_update
Jun  7 00:28:13 10    5/1   0     tcWhile Xprd     mstptimer_update
```

```
Jun  7 00:28:13 10    3/1   1     tcWhile Xprd     mstptimer_update
Jun  7 00:28:13 10    3/1   0     tcWhile Xprd     mstptimer_update
Jun  7 00:28:13 10    1/11  1     tcWhile Xprd     mstptimer_update
Jun  7 00:28:13 10    1/11  0     tcWhile Xprd     mstptimer_update
Jun  7 00:28:12 10    3/1   0     Rx Pkt Delay- MP mstpmgr_rx_bpdu()
Jun  7 00:27:39 10    -     1     RootRole select  mstp_updtRolesMsti
Jun  7 00:27:39 10    3/1   1     Superior PDU rx  mstp_pim_action
Jun  7 00:27:39 10    3/1   0     Superior PDU rx  mstp_pim_action
Jun  7 00:27:39 10    3/1   0     Rx Pkt Delay- MP mstpmgr_rx_bpdu()
Jun  7 00:27:39 10    5/1   1     TCM Detected     mstp_tcm_action
Jun  7 00:27:39 10    5/1   1     Moved to FWD     mstp_enableForwarding
Jun  7 00:27:39 10    5/1   0     TCM Detected     mstp_tcm_action
Jun  7 00:27:39 10    5/1   0     Moved to FWD     mstp_enableForwarding
Jun  7 00:27:39 10    3/1   1     TCM Detected     mstp_tcm_action
Jun  7 00:27:39 10    3/1   1     Moved to FWD     mstp_enableForwarding
Jun  7 00:27:39 10    3/1   0     TCM Detected     mstp_tcm_action
Jun  7 00:27:39 10    3/1   0     Moved to FWD     mstp_enableForwarding
Jun  7 00:27:39 10    1/11  1     TCM Detected     mstp_tcm_action
Jun  7 00:27:39 10    1/11  1     Moved to FWD     mstp_enableForwarding
```

The **show mstp history-log region** command displays the MSTP debug logs for the specified region ID. Command output resembles the following example.

```
Brocade# show mstp history-log region 10
Timestamp       RegId Port  MSTID Event            Triggered API
--------------------------------------------------------------------------------
Jun  7 00:29:24 10    5/1   1     tcWhile Xprd     mstptimer_update
Jun  7 00:29:24 10    5/1   0     tcWhile Xprd     mstptimer_update
Jun  7 00:29:24 10    1/11  1     tcWhile Xprd     mstptimer_update
Jun  7 00:29:24 10    1/11  0     tcWhile Xprd     mstptimer_update
Jun  7 00:28:48 10    5/1   1     tcWhile Xprd     mstptimer_update
Jun  7 00:28:48 10    5/1   0     tcWhile Xprd     mstptimer_update
Jun  7 00:28:48 10    1/11  1     tcWhile Xprd     mstptimer_update
Jun  7 00:28:48 10    1/11  0     tcWhile Xprd     mstptimer_update
Jun  7 00:28:25 10    5/1   0     Rx Pkt Delay- MP mstpmgr_rx_bpdu()
Jun  7 00:28:13 10    5/1   1     tcWhile Xprd     mstptimer_update
Jun  7 00:28:13 10    5/1   0     tcWhile Xprd     mstptimer_update
Jun  7 00:28:13 10    3/1   1     tcWhile Xprd     mstptimer_update
Jun  7 00:28:13 10    3/1   0     tcWhile Xprd     mstptimer_update
Jun  7 00:28:13 10    1/11  1     tcWhile Xprd     mstptimer_update
Jun  7 00:28:13 10    1/11  0     tcWhile Xprd     mstptimer_update
Jun  7 00:28:12 10    3/1   0     Rx Pkt Delay- MP mstpmgr_rx_bpdu()
Jun  7 00:27:39 10    -     1     RootRole select  mstp_updtRolesMsti
Jun  7 00:27:39 10    3/1   1     Superior PDU rx  mstp_pim_action
Jun  7 00:27:39 10    3/1   0     Superior PDU rx  mstp_pim_action
Jun  7 00:27:39 10    3/1   0     Rx Pkt Delay- MP mstpmgr_rx_bpdu()
Jun  7 00:27:39 10    5/1   1     TCM Detected     mstp_tcm_action
Jun  7 00:27:39 10    5/1   1     Moved to FWD     mstp_enableForwarding
Jun  7 00:27:39 10    5/1   0     TCM Detected     mstp_tcm_action
Jun  7 00:27:39 10    5/1   0     Moved to FWD     mstp_enableForwarding
Jun  7 00:27:39 10    3/1   1     TCM Detected     mstp_tcm_action
Jun  7 00:27:39 10    3/1   1     Moved to FWD     mstp_enableForwarding
Jun  7 00:27:39 10    3/1   0     TCM Detected     mstp_tcm_action
Jun  7 00:27:39 10    3/1   0     Moved to FWD     mstp_enableForwarding
Jun  7 00:27:39 10    1/11  1     TCM Detected     mstp_tcm_action
Jun  7 00:27:39 10    1/11  1     Moved to FWD     mstp_enableForwarding
Jun  7 00:27:39 10    1/11  0     TCM Detected     mstp_tcm_action
Jun  7 00:27:39 10    1/11  0     Moved to FWD     mstp_enableForwarding
Jun  7 00:27:11 10    5/1   0     Rx Pkt Delay- MP mstpmgr_rx_bpdu()
Jun  7 00:27:11 10    -     0     MSTP Timer Delay mstptimer_tick
```

```
Jun  7 00:27:11 10    -     0     MSTP Timer Delay mstptimer_tick
Jun  7 00:27:11 10    -     0     MSTP Timer Delay mstptimer_tick
Jun  7 00:27:11 10    1/11  0     Rx Pkt Delay- MP mstpmgr_rx_bpdu()
Jun  7 00:26:53 10    1/11  0     Rx Pkt Delay- MP mstpmgr_rx_bpdu()
Jun  7 00:26:45 10    -     0     MSTP Timer Delay mstptimer_tick
Jun  7 00:26:45 10    -     0     MSTP Timer Delay mstptimer_tick
Jun  7 00:26:45 10    -     0     MSTP Timer Delay mstptimer_tick
Jun  7 00:26:30 10    -     0     MSTP Timer Delay mstptimer_tick
Jun  7 00:26:30 10    -     0     MSTP Timer Delay mstptimer_tick
Jun  7 00:26:30 10    -     0     MSTP Timer Delay mstptimer_tick
Jun  7 00:26:07 10    -     0     MSTP Timer Delay mstptimer_tick
Jun  7 00:26:07 10    -     0     MSTP Timer Delay mstptimer_tick
Jun  7 00:26:07 10    -     0     MSTP Timer Delay mstptimer_tick
Jun  7 00:26:02 10    -     0     MSTP Timer Delay mstptimer_tick
Jun  7 00:26:00 10    3/1   8     rrWhile Xprd     mstptimer_update
Jun  7 00:26:00 10    3/1   7     rrWhile Xprd     mstptimer_update
Jun  7 00:26:00 10    3/1   6     rrWhile Xprd     mstptimer_update
Jun  7 00:26:00 10    3/1   5     rrWhile Xprd     mstptimer_update
Jun  7 00:26:00 10    3/1   4     rrWhile Xprd     mstptimer_update
Jun  7 00:26:00 10    3/1   3     rrWhile Xprd     mstptimer_update
Jun  7 00:26:00 10    3/1   2     rrWhile Xprd     mstptimer_update
Jun  7 00:26:00 10    3/1   1     rrWhile Xprd     mstptimer_update
Jun  7 00:26:00 10    3/1   0     recvInfWhil Xprd mstptimer_update
Jun  7 00:26:00 10    5/1   8     Moved to BLK     mstp_disableForwarding
Jun  7 00:26:00 10    1/11  8     Moved to BLK     mstp_disableForwarding
Jun  7 00:26:00 10    -     8     RootRole select  mstp_updtRolesMsti
Jun  7 00:26:00 10    5/1   7     Moved to BLK     mstp_disableForwarding
Jun  7 00:26:00 10    1/11  7     Moved to BLK     mstp_disableForwarding
Jun  7 00:26:00 10    -     7     RootRole select  mstp_updtRolesMsti
Jun  7 00:26:00 10    5/1   6     Moved to BLK     mstp_disableForwarding
Jun  7 00:26:00 10    1/11  6     Moved to BLK     mstp_disableForwarding
Jun  7 00:26:00 10    -     6     RootRole select  mstp_updtRolesMsti
Jun  7 00:26:00 10    5/1   5     Moved to BLK     mstp_disableForwarding
Jun  7 00:26:00 10    1/11  5     Moved to BLK     mstp_disableForwarding
```

MSTP debug commands

This section describes the MSTP-related debug commands.

debug mstp

Syntax: [**no**] **debug mstp** [**bpdu** | **event** | **mstid** *num* | **port** [**ethernet** *slot/port* | **pos** *slot/port*] | **region** *region-id* | **show** | **state** | **verbose**]

- **bpdu** - Displays MSTP bridge protocol data units (BPDUs).
- **event** - Displays MSTP state machine events.
- **mstid** *num* - Displays debugging information for a specific MSTP instance.
- **port** - Displays debugging information for a specific MSTP port.
- **ethernet** *slot/port* - Specifies the Ethernet interface.
- **pos** *slot/port* - Specifies the POS interface.
- **region** *region-id* - Displays debugging information related to a particular MSTP PB or PBB region.
- **show** - Displays the current MSTP debug parameters.

- **state** - Displays debugging information about the MSTP port state events.
- **verbose** - Displays MSTP debugging information in the verbose mode.

The **debug mstp** command generates debugging information for the configured MSTP debugging parameters. Command output resembles the following example.

```
Brocade# debug mstp
Aug 12 17:41:14 Region 1, MSTP: PTX IDLE->TRANSMIT_PERIODIC - Port 2/5
Aug 12 17:41:14 Region 1, MSTP: PTX IDLE->TRANSMIT_RSTP - Port 2/5
Aug 12 17:41:14 Region 1, MST 0, Port 2/5 - sent MST BPDU
        0000 03 02 54 8000000000af7800 00000000
        8000000000af7800 8035 0000 1400 0200 0f00
Aug 12 17:41:14 Region 1, MST 1, Port 2/5 - sent MSTI config message
        7e 8001000000af7800 00000000 8000 80 14
Aug 12 17:41:14 Region 1, MSTP: PTX IDLE->TRANSMIT_PERIODIC - Port 3/5
Aug 12 17:41:14 Region 1, MSTP: PTX IDLE->TRANSMIT_RSTP - Port 3/5
Aug 12 17:41:14 Region 1, MST 0, Port 3/5 - sent MST BPDU
        0000 03 02 54 8000000000af7800 00000000
        8000000000af7800 8065 0000 1400 0200 0f00
Aug 12 17:41:14 Region 1, MST 1, Port 3/5 - sent MSTI config message
        7e 8001000000af7800 00000000 8000 80 14
Aug 12 17:41:14 Region 2, MSTP: PRX RECEIVE->RECEIVE - Port 3/5
Aug 12 17:41:14 Region 2, MST 0, Port 3/5 - received BPDU
        0000 03 02 14 8000000000af7800 00000000
        8000000000af7800 8035 0000 0014 0002 000f
Aug 12 17:41:14 Region 2, MST 1, Port 3/5 - received MSTI config message
        7e 8001000000af7800 00000000 8000 80 14
Aug 12 17:41:14 Region 2, MSTP: PIM CURRENT->RECEIVE - MST 0, Port 3/5
Aug 12 17:41:14 Region 2, MSTP: PIM RECEIVE->OTHER - MST 0, Port 3/5
Aug 12 17:41:14 Region 2, MSTP: PIM CURRENT->RECEIVE - MST 1, Port 3/5
Aug 12 17:41:14 Region 2, MSTP: PIM RECEIVE->OTHER - MST 1, Port 3/5
Aug 12 17:41:14 Region 2, MSTP: PRX RECEIVE->RECEIVE - Port 2/5
```

debug mstp bpdu

Syntax: [**no**] **debug mstp bpdu**

This command displays MSTP BPDUs. Command output resembles the following example for MSTP PBB configurations.

```
Brocade# debug mstp bpdu
MSTP:  debugging is on
Aug 12 17:59:32 Region 1, MST 0, Port 2/5 - sent MST BPDU
        0000 03 02 54 8000000000af7800 00000000
        8000000000af7800 8035 0000 1400 0200 0f00
Aug 12 17:59:32 Region 1, MST 1, Port 2/5 - sent MSTI config message
        7e 8001000000af7800 00000000 8000 80 14
Aug 12 17:59:32 Region 1, MST 0, Port 3/5 - sent MST BPDU
        0000 03 02 54 8000000000af7800 00000000
        8000000000af7800 8065 0000 1400 0200 0f00
Aug 12 17:59:32 Region 1, MST 1, Port 3/5 - sent MSTI config message
        7e 8001000000af7800 00000000 8000 80 14
```

debug mstp event

Syntax: [**no**] **debug mstp event**

This command displays MSTP state events. Command output resembles the following example for MSTP PBB configurations.

```
Brocade# debug mstp event
Aug 12 18:02:30 Region 1, MSTP: PTX IDLE->TRANSMIT_PERIODIC - Port 2/5
Aug 12 18:02:30 Region 1, MSTP: PTX IDLE->TRANSMIT_RSTP - Port 2/5
Aug 12 18:02:30 Region 1, MSTP: PTX IDLE->TRANSMIT_PERIODIC - Port 3/5
Aug 12 18:02:30 Region 1, MSTP: PTX IDLE->TRANSMIT_RSTP - Port 3/5
Aug 12 18:02:30 Region 1, MSTP: PRX RECEIVE->RECEIVE - Port 3/5
Aug 12 18:02:30 Region 1, MSTP: PIM CURRENT->RECEIVE - MST 0, Port 3/5
Aug 12 18:02:30 Region 1, MSTP: PIM RECEIVE->OTHER - MST 0, Port 3/5
Aug 12 18:02:30 Region 1, MSTP: PIM CURRENT->RECEIVE - MST 1, Port 3/5
```

debug mstp mstid

Syntax: [**no**] **debug mstp mstid** *num* **region** *region-id*

- *num* - Specifies the ID of the MSTP instance in the configured regions.
- **region** *region-id* - Specifies the ID of the MSTP PB or PBB region.

This command displays debugging information for a specific MSTP instance in the configured regions. Command output resembles the following example for MSTP PBB configurations.

```
Brocade# debug mstp mstid 1
Aug 12 18:49:26 Region 1, MSTP: PTX IDLE->TRANSMIT_PERIODIC - Port 2/5
Aug 12 18:49:26 Region 1, MSTP: PTX IDLE->TRANSMIT_RSTP - Port 2/5
Aug 12 18:49:26 Region 1, MST 1, Port 2/5 - sent MSTI config message
        7e 8001000000af7800 00000000 8000 80 14
Aug 12 18:49:26 Region 1, MSTP: PTX IDLE->TRANSMIT_PERIODIC - Port 3/5
Aug 12 18:49:26 Region 1, MSTP: PTX IDLE->TRANSMIT_RSTP - Port 3/5
Aug 12 18:49:26 Region 1, MST 1, Port 3/5 - sent MSTI config message
        7e 8001000000af7800 00000000 8000 80 14
Aug 12 18:49:26 Region 2, MSTP: PRX RECEIVE->RECEIVE - Port 3/5
Aug 12 18:49:26 Region 2, MST 1, Port 3/5 - received MSTI config message
        7e 8001000000af7800 00000000 8000 80 14
Aug 12 18:49:26 Region 2, MSTP: PIM CURRENT->RECEIVE - MST 1, Port 3/5
Aug 12 18:49:26 Region 2, MSTP: PIM RECEIVE->OTHER - MST 1, Port 3/5
Aug 12 18:49:26 Region 2, MSTP: PRX RECEIVE->RECEIVE - Port 2/5
Aug 12 18:49:26 Region 2, MST 1, Port 2/5 - received MSTI config message
```

debug mstp port

Syntax: [**no**] **debug mstp port** [**ethernet** *slot/port* | **pos** *slot/port*]

- **ethernet** *slot/port* - Specifies the Ethernet interface.
- **pos** *slot/port* - Specifies the Packet over SONET (POS) interface.

This command displays debugging information for a specific MSTP port in the configured regions. Command output resembles the following example for MSTP PBB configurations.

```
Brocade# debug mstp port ethernet 1/15
MSTP debugging turned on for ports ethernet 1/15
```

debug mstp region

Syntax: [**no**] **debug mstp region** *region-id*

The *region-id* variable specifies the ID of the configured MSTP region.

This command displays debugging information related to a particular MSTP PB or PBB region. Command output resembles the following example for MSTP PBB configurations.

```
Brocade# debug mstp region 1
Aug 12 17:41:14 Region 1, MSTP: PTX IDLE->TRANSMIT_PERIODIC - Port 2/5
```

```
Aug 12 17:41:14 Region 1, MSTP: PTX IDLE->TRANSMIT_RSTP - Port 2/5
Aug 12 17:41:14 Region 1, MST 0, Port 2/5 - sent MST BPDU
        0000 03 02 54 8000000000af7800 00000000
        8000000000af7800 8035 0000 1400 0200 0f00
Aug 12 17:41:14 Region 1, MST 1, Port 2/5 - sent MSTI config message
        7e 8001000000af7800 00000000 8000 80 14
Aug 12 17:41:14 Region 1, MSTP: PTX IDLE->TRANSMIT_PERIODIC - Port 3/5
Aug 12 17:41:14 Region 1, MSTP: PTX IDLE->TRANSMIT_RSTP - Port 3/5
Aug 12 17:41:14 Region 1, MST 0, Port 3/5 - sent MST BPDU
        0000 03 02 54 8000000000af7800 00000000
        8000000000af7800 8065 0000 1400 0200 0f00
Aug 12 17:41:14 Region 1, MST 1, Port 3/5 - sent MSTI config message
        7e 8001000000af7800 00000000 8000 80 14
Aug 12 17:41:14 Region 1, MSTP: PRX RECEIVE->RECEIVE - Port 3/5
Aug 12 17:41:14 Region 1, MST 0, Port 3/5 - received BPDU
        0000 03 02 14 8000000000af7800 00000000
        8000000000af7800 8035 0000 0014 0002 000f
Aug 12 17:41:14 Region 1, MST 1, Port 3/5 - received MSTI config message
        7e 8001000000af7800 00000000 8000 80 14
Aug 12 17:41:14 Region 1, MSTP: PIM CURRENT->RECEIVE - MST 0, Port 3/5
Aug 12 17:41:14 Region 1, MSTP: PIM RECEIVE->OTHER - MST 0, Port 3/5
Aug 12 17:41:14 Region 1, MSTP: PIM CURRENT->RECEIVE - MST 1, Port 3/5
Aug 12 17:41:14 Region 1, MSTP: PIM RECEIVE->OTHER - MST 1, Port 3/5
Aug 12 17:41:14 Region 1, MSTP: PRX RECEIVE->RECEIVE - Port 2/5
```

debug mstp show

Syntax: [no] debug mstp show

This command displays the current MSTP debug parameters. Command output resembles the following example for MSTP PBB configurations.

```
Brocade# debug mstp show
MSTP debug Parameters
---------------------
MSTP debugging is OFF [Mode: Brief]
StateMachineEvents BpduEvents PortStateEvents are being tracked
Ports: All
MSTP instances: (indices) All
Regions: All
```

debug mstp state

Syntax: [no] debug mstp state

This command displays debugging information related to MSTP port state events. Command output resembles the following example for MSTP PBB configurations.

```
Brocade# debug mstp state
Aug 12 19:14:08 Region 1, MSTP: MST 0, Port 2/5 - State: LEARNING
Aug 12 19:14:08 Region 1, MSTP: MST 1, Port 2/5 - State: LEARNING
Aug 12 19:14:23 Region 1, MSTP: MST 0, Port 2/5 - State: FORWARDING
Aug 12 19:14:23 Region 1, MSTP: MST 1, Port 2/5 - State: FORWARDING
```

debug mstp verbose

Syntax: [no] debug mstp verbose

This command displays debugging information for the configured MSTP debug parameters in the verbose mode. Command output resembles the following example for MSTP PBB configurations.

```
Brocade# debug mstp verbose
Aug 12 19:20:34 Region 1, MST 0, Port 2/5 - received BPDU
        protocol-id: 0000
        protocol-version: 03
        type: 02
        flags:  agree forward learn designated propose
        root-id: 8000000000af7800
        path-cost: 00000000
        bridge-id: 8000000000af7800
        port-id: 8065
        message-age: 0000
        max-age: 0014
        hello-time: 0002
        forward-delay: 000f
Aug 12 19:20:34 Region 1, MST 1, Port 2/5 - received MSTI config message
        flags:  agree forward learn designated propose
        regional_root: 8001000000af7800
        int_path_cost: 00000000
        bridge_priority: 8000
        port_priority: 80
        remaining_hops: 14
Aug 12 19:20:34 Region 1, MSTP: PIM CURRENT->RECEIVE - MST 0, Port 2/5
Aug 12 19:20:34 Region 1, MSTP: PIM RECEIVE->OTHER - MST 0, Port 2/5
Aug 12 19:20:34 Region 1, MSTP: PIM CURRENT->RECEIVE - MST 1, Port 2/5
```

RSTP show commands

This section describes the show command that displays RSTP information.

show rstp

Syntax: **show rstp** [**detail** | **vlan** v*lan-id* | **vpls-id** *vpls-id*]

- **detail** - Displays detailed RSTP information.
- **vlan** v*lan-id* - Displays RSTP information for the specified VLAN.
- **vpls-id** *vpls-id* - Displays RSTP information for the specified Provider Backbone Bridging (PBB) VPLS instance.

This command displays RSTP information. Command output resembles the following example for RSTP PBB configurations.

```
Brocade# show rstp vpls-id 1
VPLS Instance ID 1  - RSTP instance 0
--------------------------------------------------------------------
RSTP (IEEE 802.1w) Bridge Parameters:
Bridge         Bridge Bridge Bridge Force   tx
Identifier        MaxAge Hello  FwdDly Version Hold
hex               sec    sec    sec            cnt
0001000480a04000  20     2      15     Default 3
RootBridge    RootPath DesignatedBridge Root Max  Hel Fwd
Identifier       Cost     Identifier       Port Age  lo  Dly
hex       hex                            sec  sec sec
0001000480a04000 0        0001000480a04000 Root 20   2   15
RSTP (IEEE 802.1w) Port Parameters:
<--- Config Params -->|<-------------- Current state ----------------->
Port Pri  PortPath P2P Edge Role     State    Designa- Designated
Num       Cost     Mac Port                   tedcost  bridge
1/3  128  20000    T   F    DISABLED DISABLED 0        0000000000000000
```

```
1/13 128  20000    T   F    DISABLED DISABLED 0        0000000000000000
```

show rstp detail

Syntax: **show rstp detail** [**vpls-id** *vpls-id*]

The **vpls-id** *vpls-id* parameter specifies the ID of the VPLS instance for which you want to display the detailed RSTP information.

This command displays detailed RSTP information. Command output resembles the following example for RSTP PBB configurations.

```
Brocade# show rstp detail vpls-id 1
VPLS Instance ID – 1  - RSTP instance 0
--------------------------------------------------------------------
RSTP (IEEE 802.1w) Bridge Parameters:
BridgeId 0001000480a04000, RootBridgeId 0001000480a04000
Control ports - ethernet 1/3 ethernet 1/13
ForceVersion 2, MigrateTime 3, TxHoldCount 3

RSTP (IEEE 802.1w) Port Parameters:
Port 1/3 - Role: DISABLED - State: DISABLED
Port 1/13 - Role: DISABLED - State: DISABLED
```

show rstp history-log

Syntax: **show rstp history-log** [**brief** | **vlan** *vlan-id*]

- **brief** - Displays the last 30 RSTP debug logs.
- **vlan** *vlan-id* - Displays the RSTP debug logs related to the specified VLAN.

This command displays the RSTP debug logs for all the VLANs. Command output resembles the following example.

```
Brocade# show rstp history-log
timestamp          PORT   VLAN ID    Event                            Triggered starting API
=====================================================================================================
Sept  20 09:20:11  1/1    100        PRT due to rrwhile timer expiry  rstputil_port_timer_update
Sept  20 09:21:11  1/3    4095       PRT due to rbwhile expiry        rstputil_port_timer_update
Sept  20 09:22:11  1/2    200        PRT due to rbwhile expiry        rstputil_port_timer_update
Sept  20 11:18:40  1/1    4095       Superior BPDU recvd              rstputil_process_bpdu
Sept  20 11:18:42  1/1    100        rcvdInfoWhile Timer Expiry       rstputil_port_timer_update
Sept  20 11:19:40  1/1    200        Superior BPDU recvd              rstputil_process_bpdu
```

RSTP debug commands

The **debug rstp** commands are similar in form and function to the **debug spanning-tree** commands in a non-PBB environment. To support RSTP over PBB networks, the existing RSTP-related debug commands are enhanced to enable debugging on a VPLS VLAN instance.

This section describes the debug commands for the RSTP over PBB environment.

debug rstp

Syntax: **debug rstp** [**bpdu** | **event** | **mct** | **port** [**ethernet** *slot/port* | **pos** *slot/port*] | **reset** | **show** | **verbose** | **vlan** v*lan-id* | **vpls-id** *vpls-id*]

- **bpdu** - Enables or disables RSTP BPDU debugging.

- **event** - Enables or disables RSTP non-BPDU events debugging.
- **mct** - Enables or disables RSTP MCT debugging.
- **port** - Displays debugging information for a specific RSTP instance on the ports.
- **ethernet** *slot/port* - Specifies the Ethernet interface.
- **pos** *slot/port* - Specifies the Packet over SONET (POS) interface
- **reset** - Resets all the RSTP debugging parameters to default.
- **show** - Displays the current RSTP debugging parameters.
- **verbose** - Enables or disables the RSTP verbose debugging mode.
- **vlan** *vlan-id* - Displays debugging information for a specific RSTP VLAN instance.
- **vpls-id** *vpls-id* - Displays debugging information for a specific RSTP VPLS instance.

The **debug rstp** command generates debugging information for the configured RSTP debugging parameters. Command output resembles the following example.

```
Brocade# debug rstp
RSTP:  debugging is on
Apr 16 09:02:32.865 RSTP: Received RSTP BPDU - VPLS Instance 1 Port 1/2
        0000 02 02 7e 80000000008ffc20 00000000
        80000000008ffc20 8092 0000 0014 0002 000f
Apr 16 09:02:32.865 RSTP: PIM CURRENT->RECEIVE - VPLS Instance 1, Port 1/2
Apr 16 09:02:32.865 RSTP: PIM RECEIVE->REPEAT - VPLS Instance 1, Port 1/2
Apr 16 09:02:32.866 RSTP: Received RSTP BPDU - VPLS Instance 1 Port 1/3
        0000 02 02 7e 80000000008ffc20 00000000
        80000000008ffc20 8093 0000 0014 0002 000f
Apr 16 09:02:32.866 RSTP: PIM CURRENT->RECEIVE - VPLS Instance 1, Port 1/3
Apr 16 09:02:32.866 RSTP: PIM RECEIVE->REPEAT - VPLS Instance 1, Port 1/3
Apr 16 09:02:34.148 RSTP: PTX IDLE->TRANSMIT_PERIODIC - VPLS Instance 1, Port 1/1
Apr 16 09:02:34.148 RSTP: PTX IDLE->TRANSMIT_RSTP - VPLS Instance 1, Port 1/1
Apr 16 09:02:34.148 RSTP: Sending RST BPDU - VPLS Instance 1 Port 1/1
Apr 16 09:02:34.148     0000 02 02 7c 80000000008ffc20 000007d0
        8000000000903a20 8001 0001 0014 0002 000f
Apr 16 09:02:34.148 RSTP: PTX IDLE->TRANSMIT_PERIODIC - VPLS Instance 1, Port 1/2
Apr 16 09:02:34.148 RSTP: PTX IDLE->TRANSMIT_PERIODIC - VPLS Instance 1, Port 1/3
Apr 16 09:02:34.148 RSTP: PTX IDLE->TRANSMIT_PERIODIC - VPLS Instance 1, Port 1/4
Apr 16 09:02:34.148 RSTP: PTX IDLE->TRANSMIT_RSTP - VPLS Instance 1, Port 1/4
Apr 16 09:02:34.148 RSTP: Sending RST BPDU - VPLS Instance 1 Port 1/4
Apr 16 09:02:34.148     0000 02 02 7c 80000000008ffc20 000007d0
        8000000000903a20 8004 0001 0014 0002 000f
Apr 16 09:02:34.865 RSTP: Received RSTP BPDU - VPLS Instance 1 Port 1/2
        0000 02 02 7e 80000000008ffc20 00000000
        80000000008ffc20 8092 0000 0014 0002 000f
Apr 16 09:02:34.865 RSTP: PIM CURRENT->RECEIVE - VPLS Instance 1, Port 1/2
Apr 16 09:02:34.865 RSTP: PIM RECEIVE->REPEAT - VPLS Instance 1, Port 1/2
Apr 16 09:02:34.865 RSTP: Received RSTP BPDU - VPLS Instance 1 Port 1/3
        0000 02 02 7e 80000000008ffc20 00000000
        80000000008ffc20 8093 0000 0014 0002 000f
Apr 16 09:02:34.866 RSTP: PIM CURRENT->RECEIVE - VPLS Instance 1, Port 1/3
Apr 16 09:02:34.866 RSTP: PIM RECEIVE->REPEAT - VPLS Instance 1, Port 1/3
Apr 16 09:02:36.148 RSTP: PTX IDLE->TRANSMIT_PERIODIC - VPLS Instance 1, Port 1/1
Apr 16 09:02:36.148 RSTP: PTX IDLE->TRANSMIT_RSTP - VPLS Instance 1, Port 1/1
Apr 16 09:02:36.148 RSTP: Sending RST BPDU - VPLS Instance 1 Port 1/1
Apr 16 09:02:36.148     0000 02 02 7c 80000000008ffc20 000007d0
        8000000000903a20 8001 0001 0014 0002 000f
Apr 16 09:02:36.148 RSTP: PTX IDLE->TRANSMIT_PERIODIC - VPLS Instance 1, Port 1/2
Apr 16 09:02:36.148 RSTP: PTX IDLE->TRANSMIT_PERIODIC - VPLS Instance 1, Port 1/3
Apr 16 09:02:36.148 RSTP: PTX IDLE->TRANSMIT_PERIODIC - VPLS Instance 1, Port 1/4
```

```
Apr 16 09:02:36.148 RSTP: PTX IDLE->TRANSMIT_RSTP - VPLS Instance 1, Port 1/4
Apr 16 09:02:36.148 RSTP: Sending RST BPDU - VPLS Instance 1 Port 1/4
Apr 16 09:02:36.148     0000 02 02 7c 80000000008ffc20 000007d0
        8000000000903a20 8004 0001 0014 0002 000f
Apr 16 09:02:36.865 RSTP: Received RSTP BPDU - VPLS Instance 1 Port 1/2
        0000 02 02 7e 80000000008ffc20 00000000
        80000000008ffc20 8092 0000 0014 0002 000f
Apr 16 09:02:36.865 RSTP: PIM CURRENT->RECEIVE - VPLS Instance 1, Port 1/2
Apr 16 09:02:36.865 RSTP: PIM RECEIVE->REPEAT - VPLS Instance 1, Port 1/2
Apr 16 09:02:36.865 RSTP: Received RSTP BPDU - VPLS Instance 1 Port 1/3
        0000 02 02 7e 80000000008ffc20 00000000
        80000000008ffc20 8093 0000 0014 0002 000f
Apr 16 09:02:36.866 RSTP: PIM CURRENT->RECEIVE - VPLS Instance 1, Port 1/3
Apr 16 09:02:36.866 RSTP: PIM RECEIVE->REPEAT - VPLS Instance 1, Port 1/3
Apr 16 09:02:38.148 RSTP: PTX IDLE->TRANSMIT_PERIODIC - VPLS Instance 1, Port 1/1
Apr 16 09:02:38.148 RSTP: PTX IDLE->TRANSMIT_RSTP - VPLS Instance 1, Port 1/1
Apr 16 09:02:38.148 RSTP: Sending RST BPDU - VPLS Instance 1 Port 1/1
Apr 16 09:02:38.148     0000 02 02 7c 80000000008ffc20 000007d0
        8000000000903a20 8001 0001 0014 0002 000f
Apr 16 09:02:38.148 RSTP: PTX IDLE->TRANSMIT_PERIODIC - VPLS Instance 1, Port 1/2
Apr 16 09:02:38.148 RSTP: PTX IDLE->TRANSMIT_PERIODIC - VPLS Instance 1, Port 1/3
Apr 16 09:02:38.148 RSTP: PTX IDLE->TRANSMIT_PERIODIC - VPLS Instance 1, Port 1/4
Apr 16 09:02:38.148 RSTP: PTX IDLE->TRANSMIT_RSTP - VPLS Instance 1, Port 1/4
Apr 16 09:02:38.148 RSTP: Sending RST BPDU - VPLS Instance 1 Port 1/4
Apr 16 09:02:38.148     0000 02 02 7c 80000000008ffc20 000007d0
        8000000000903a20 8004 0001 0014 0002 000f
Apr 16 09:02:38.865 RSTP: Received RSTP BPDU - VPLS Instance 1 Port 1/2
        0000 02 02 7e 80000000008ffc20 00000000
        80000000008ffc20 8092 0000 0014 0002 000f
Apr 16 09:02:38.865 RSTP: PIM CURRENT->RECEIVE - VPLS Instance 1, Port 1/2
Apr 16 09:02:38.865 RSTP: PIM RECEIVE->REPEAT - VPLS Instance 1, Port 1/2
Apr 16 09:02:38.865 RSTP: Received RSTP BPDU - VPLS Instance 1 Port 1/3
        0000 02 02 7e 80000000008ffc20 00000000
        80000000008ffc20 8093 0000 0014 0002 000f
Apr 16 09:02:38.866 RSTP: PIM CURRENT->RECEIVE - VPLS Instance 1, Port 1/3
Apr 16 09:02:38.866 RSTP: PIM RECEIVE->REPEAT - VPLS Instance 1, Port 1/3
```

debug rstp bpdu

Syntax: [no] debug rstp bpdu

This command enables or disables debugging of RSTP BPDUs. Command output resembles the following example.

```
Brocade# debug rstp bpdu
RSTP Bpdu debugging ON
```

debug rstp event

Syntax: [no] debug rstp event

This command enables or disables debugging of RSTP non-BPDU events. Command output resembles the following example.

```
Brocade# debug rstp event
RSTP Event debugging ON
```

debug rstp mct

Syntax: [no] debug rstp mct

This command enables or disables debugging of RSTP MCT events. Command output resembles the following example.

```
Brocade# debug rstp mct
RSTP MCT debugging ON
```

debug rstp port

Syntax: [**no**] **debug rstp port** [**ethernet** *slot/port* | **pos** *slot/port*]

- **ethernet** *slot/port* - Specifies the Ethernet interface.
- **pos** *slot/port* - Specifies the POS interface.

This command displays debugging information for a specific RSTP instance on the ports. Command output resembles the following example.

```
Brocade# debug rstp port ethernet 1/2
RSTP debugging turned on for ports ethernet 1/2
```

debug rstp reset

Syntax: [**no**] **debug rstp reset**

This command resets all the RSTP debugging parameters to default.

debug rstp show

Syntax: [**no**] **debug rstp show**

This command displays the current RSTP debugging parameters. Command output resembles the following example.

```
Brocade# debug rstp show
RSTP Debug Parameters
---------------------
RSTP debugging is OFF [Mode: Brief]
NonBpduEvents BpduEvents are being tracked
Ports: All
VLANs: All
VPLS ID: All
VPLS : All
```

debug rstp verbose

Syntax: [**no**] **debug rstp verbose**

This command displays debugging information for the configured RSTP debugging parameters in the verbose mode. Command output resembles the following example.

```
Brocade# debug rstp verbose
RSTP debugging set to VERBOSE mode
Brocade# debug rstp show
RSTP Debug Parameters
---------------------
RSTP debugging is OFF [Mode: Verbose]
BpduEvents are being tracked
Ports: All
VLANs:
VPLSs: 1
```

```
Brocade# debug rstp
RSTP:  debugging is on
Apr 16 09:04:56.148 RSTP: Sending RST BPDU - VPLS Instance 1 Port 1/1
Apr 16 09:04:56.148     protocol-id: 0000
        protocol-version: 02
        type: 02
        flags: 7c
        root-id: 80000000008ffc20
        path-cost: 000007d0
        bridge-id: 8000000000903a20
        port-id: 8001
        message-age: 0001
        max-age: 0014
        hello-time: 0002
        forward-delay: 000f
Apr 16 09:04:56.148 RSTP: Sending RST BPDU - VPLS Instance 1 Port 1/4
Apr 16 09:04:56.148     protocol-id: 0000
        protocol-version: 02
        type: 02
        flags: 7c
        root-id: 80000000008ffc20
        path-cost: 000007d0
        bridge-id: 8000000000903a20
        port-id: 8004
        message-age: 0001
        max-age: 0014
        hello-time: 0002
        forward-delay: 000f
Apr 16 09:04:56.867 RSTP: Received RSTP BPDU - VPLS Instance 1 Port 1/2
        protocol-id: 0000
        protocol-version: 02
        type: 02
        flags: 7e - agree fwd lrn designated propose
        root-id: 80000000008ffc20
        path-cost: 00000000
        bridge-id: 80000000008ffc20
        port-id: 8092
        message-age: 0000
        max-age: 0014
        hello-time: 0002
        forward-delay: 000f
```

debug rstp vlan

Syntax: [**no**] **debug rstp vlan** *vlan-id*

The *vlan-id* variable specifies the ID of the configured VLAN.

This command enables or disables RSTP debugging for a specific VLAN and displays debugging information for that VLAN. Command output resembles the following example.

```
Brocade# debug rstp vlan 2
RSTP debugging turned on for VLAN instance 2
```

debug rstp vpls-id

Syntax: [**no**] **debug rstp vpls-id** *vpls-id*

The *vpls-id* variable specifies the ID of the configured PBB VPLS instance.

This command enables or disables RSTP debugging for a specific VPLS instance and displays debugging information for that VPLS instance. Command output resembles the following example.

```
Brocade# debug rstp vpls-id 1
RSTP debugging turned on for vpls instance 1
Brocade# debug rstp
RSTP:  debugging is on
Apr 16 09:03:16.148 RSTP: PTX IDLE->TRANSMIT_PERIODIC - VPLS Instance 1, Port 1/1
Apr 16 09:03:16.148 RSTP: PTX IDLE->TRANSMIT_RSTP - VPLS Instance 1, Port 1/1
Apr 16 09:03:16.148 RSTP: Sending RST BPDU - VPLS Instance 1 Port 1/1
Apr 16 09:03:16.148     0000 02 02 7c 80000000008ffc20 000007d0
        8000000000903a20 8001 0001 0014 0002 000f
Apr 16 09:03:16.148 RSTP: PTX IDLE->TRANSMIT_PERIODIC - VPLS Instance 1, Port 1/2
Apr 16 09:03:16.148 RSTP: PTX IDLE->TRANSMIT_PERIODIC - VPLS Instance 1, Port 1/3
Apr 16 09:03:16.148 RSTP: PTX IDLE->TRANSMIT_PERIODIC - VPLS Instance 1, Port 1/4
Apr 16 09:03:16.148 RSTP: PTX IDLE->TRANSMIT_RSTP - VPLS Instance 1, Port 1/4
Apr 16 09:03:16.148 RSTP: Sending RST BPDU - VPLS Instance 1 Port 1/4
Apr 16 09:03:16.148     0000 02 02 7c 80000000008ffc20 000007d0
        8000000000903a20 8004 0001 0014 0002 000f
Apr 16 09:03:16.865 RSTP: Received RSTP BPDU - VPLS Instance 1 Port 1/2
        0000 02 02 7e 80000000008ffc20 00000000
        80000000008ffc20 8092 0000 0014 0002 000f
Apr 16 09:03:16.865 RSTP: PIM CURRENT->RECEIVE - VPLS Instance 1, Port 1/2
Apr 16 09:03:16.865 RSTP: PIM RECEIVE->REPEAT - VPLS Instance 1, Port 1/2
Apr 16 09:03:16.865 RSTP: Received RSTP BPDU - VPLS Instance 1 Port 1/3
        0000 02 02 7e 80000000008ffc20 00000000
        80000000008ffc20 8093 0000 0014 0002 000f
Apr 16 09:03:16.865 RSTP: PIM CURRENT->RECEIVE - VPLS Instance 1, Port 1/3
Apr 16 09:03:16.865 RSTP: PIM RECEIVE->REPEAT - VPLS Instance 1, Port 1/3
Apr 16 09:03:18.148 RSTP: PTX IDLE->TRANSMIT_PERIODIC - VPLS Instance 1, Port 1/1
Apr 16 09:03:18.148 RSTP: PTX IDLE->TRANSMIT_RSTP - VPLS Instance 1, Port 1/1
Apr 16 09:03:18.148 RSTP: Sending RST BPDU - VPLS Instance 1 Port 1/1
Apr 16 09:03:18.148     0000 02 02 7c 80000000008ffc20 000007d0
        8000000000903a20 8001 0001 0014 0002 000f
Apr 16 09:03:18.148 RSTP: PTX IDLE->TRANSMIT_PERIODIC - VPLS Instance 1, Port 1/2
Apr 16 09:03:18.148 RSTP: PTX IDLE->TRANSMIT_PERIODIC - VPLS Instance 1, Port 1/3
Apr 16 09:03:18.148 RSTP: PTX IDLE->TRANSMIT_PERIODIC - VPLS Instance 1, Port 1/4
Apr 16 09:03:18.148 RSTP: PTX IDLE->TRANSMIT_RSTP - VPLS Instance 1, Port 1/4
Apr 16 09:03:18.148 RSTP: Sending RST BPDU - VPLS Instance 1 Port 1/4
Apr 16 09:03:18.148     0000 02 02 7c 80000000008ffc20 000007d0
        8000000000903a20 8004 0001 0014 0002 000f
Apr 16 09:03:18.865 RSTP: Received RSTP BPDU - VPLS Instance 1 Port 1/2
        0000 02 02 7e 80000000008ffc20 00000000
        80000000008ffc20 8092 0000 0014 0002 000f
Apr 16 09:03:18.865 RSTP: PIM CURRENT->RECEIVE - VPLS Instance 1, Port 1/2
Apr 16 09:03:18.865 RSTP: PIM RECEIVE->REPEAT - VPLS Instance 1, Port 1/2
Apr 16 09:03:18.865 RSTP: Received RSTP BPDU - VPLS Instance 1 Port 1/3
        0000 02 02 7e 80000000008ffc20 00000000
        80000000008ffc20 8093 0000 0014 0002 000f
Apr 16 09:03:18.865 RSTP: PIM CURRENT->RECEIVE - VPLS Instance 1, Port 1/3
Apr 16 09:03:18.865 RSTP: PIM RECEIVE->REPEAT - VPLS Instance 1, Port 1/3
Apr 16 09:03:20.148 RSTP: PTX IDLE->TRANSMIT_PERIODIC - VPLS Instance 1, Port 1/1
Apr 16 09:03:20.148 RSTP: PTX IDLE->TRANSMIT_RSTP - VPLS Instance 1, Port 1/1
Apr 16 09:03:20.148 RSTP: Sending RST BPDU - VPLS Instance 1 Port 1/1
Apr 16 09:03:20.148     0000 02 02 7c 80000000008ffc20 000007d0
        8000000000903a20 8001 0001 0014 0002 000f
Apr 16 09:03:20.148 RSTP: PTX IDLE->TRANSMIT_PERIODIC - VPLS Instance 1, Port 1/2
Apr 16 09:03:20.148 RSTP: PTX IDLE->TRANSMIT_PERIODIC - VPLS Instance 1, Port 1/3
Apr 16 09:03:20.148 RSTP: PTX IDLE->TRANSMIT_PERIODIC - VPLS Instance 1, Port 1/4
Apr 16 09:03:20.148 RSTP: PTX IDLE->TRANSMIT_RSTP - VPLS Instance 1, Port 1/4
```

```
Apr 16 09:03:20.148 RSTP: Sending RST BPDU - VPLS Instance 1 Port 1/4
Apr 16 09:03:20.148     0000 02 02 7c 80000000008ffc20 000007d0
        8000000000903a20 8004 0001 0014 0002 000f
Apr 16 09:03:20.867 RSTP: Received RSTP BPDU - VPLS Instance 1 Port 1/2
        0000 02 02 7e 80000000008ffc20 00000000
        80000000008ffc20 8092 0000 0014 0002 000f
Apr 16 09:03:20.867 RSTP: PIM CURRENT->RECEIVE - VPLS Instance 1, Port 1/2
Apr 16 09:03:20.867 RSTP: PIM RECEIVE->REPEAT - VPLS Instance 1, Port 1/2
Apr 16 09:03:20.867 RSTP: Received RSTP BPDU - VPLS Instance 1 Port 1/3
        0000 02 02 7e 80000000008ffc20 00000000
        80000000008ffc20 8093 0000 0014 0002 000f
Apr 16 09:03:20.867 RSTP: PIM CURRENT->RECEIVE - VPLS Instance 1, Port 1/3

Apr 16 09:03:20.867 RSTP: PIM RECEIVE->REPEAT - VPLS Instance 1, Port 1/3
```

Configuration notes

- Changing the STP state of the primary port in a trunk group affects all ports in the trunk group.
- With RSTP, rapid convergence will not occur on ports connected to shared media devices, such as hubs. To take advantage of the rapid convergence provided by RSTP, make sure to explicitly configure all point-to-point links in a topology.
- When you enable SuperSpan globally, and then create a new VLAN, the new VLAN inherits the global SuperSpan state.
- In many cases, generic debugging is not useful. If multiple STP instances are configured, it can be difficult to identify content for a specific instance from the extensive output that may be generated. Use the **debug spanning-tree port** and **debug spanning-tree vlan** commands to define specific instances for debugging.

Common diagnostic scenarios

- Spanning Tree loops.
- Spanning Tree reacts to topology changes and port flapping.
- Port flapping can trigger a new Spanning Tree learning process.

LACP trunking

The Link Aggregation Control Protocol (LACP) allows ports on both sides of a redundant link to automatically configure themselves into a trunk link (aggregate link), eliminating the need for manual configuration. LACP has two modes:

- Active mode – When active link aggregation is enabled, the Brocade port can exchange standard LACP Data Unit (LACPDU) messages to negotiate trunk group configuration with the port on the other side of the link. In addition, the Brocade port actively sends LACPDU messages on the link to search for a link aggregation partner at the other end of the link, and can initiate an LACPDU exchange to negotiate link aggregation parameters with an appropriately configured remote port.
- Passive mode – In passive link aggregation, the Brocade port can exchange LACPDU messages with the port at the remote end of the link, but this port cannot search for a link aggregation port or initiate negotiation of an aggregate link. In passive mode, the port at the remote end of the link must initiate the LACPDU exchange.

When you enable link aggregation on a group of Brocade ports, the Brocade ports can negotiate with the ports at the remote ends of the links to establish trunk groups.

Trunk show commands

The following show commands display information about trunking configurations.

show lag

Syntax: show lag

This command displays trunk information for the ports managed by the CPU. The CPU assigns the hash values evenly to the trunk ports managed by the CPU.

```
Brocade# show lag
Max number of trunks: 128
Available: 127
Configured number of server trunks: 1

Configured trunks:

Trunk ID: 1
Type: Server
Ports_Configured: 2
Base FID: 0x0400
FID count: 16

Ports         14/6    14/8
Port Names    ve4012  TestCa*
Port_Status   disable disable

Operational trunks:

Trunk ID: 1
Type: Server
Duplex: None
Speed: None
Tag: Dual
Priority: level0
Active Ports: 0

Ports                14/6   14/8
Link_Status          down   down
```

Trunk debug commands

There are no debug commands specific to LACP trunking.

Configuration notes

- You cannot use 802.3ad link aggregation on a port configured as a member of a static trunk group.
- When LACP dynamically adds or changes a trunk group, the **show lag** command displays the trunk as both configured and active. However, the **show running-config** or **write terminal** commands do not contain a trunk command defining the new or changed trunk group.

- You can enable link aggregation on 802.1q tagged ports (ports that belong to more than one port-based VLAN).
- LACP cannot be configured on a VPLS endpoint port, and a VPLS endpoint cannot be configured on a physical port that has LACP enabled. This restriction is enforced by the CLI. If a VPLS endpoint receives any LACP traffic, this traffic will be dropped by the router at ingress. However, if the VPLS has CPU protection enabled, this traffic will be hardware flooded.
- Brocade recommends that you disable or remove the cables from the ports you plan to enable for dynamic link aggregation. Doing so prevents the possibility that LACP will use a partial configuration to talk to the other side of a link. A partial configuration does not cause errors, but sometimes requires LACP to be disabled and re-enabled on both sides of the link to ensure that a full configuration is used. It is easier to disable a port or remove its cable first. This applies both for active link aggregation and passive link aggregation.

Trunk formation rules

When troubleshooting trunks, make sure the following rules for trunk formation have been considered:

- Any number of ports between 2 and 20 within the same chassis can be used to configure a trunk port.
- Use the server trunk option when configuring a trunk group that connects to VPLS or Virtual Lease Line (VLL) endpoints.
- Ports in a trunk must have the same speed, negotiation mode, and Quality of Service (QoS) priority or the trunk is rejected.
- All ports configured in a trunk must be configured with the same port attributes.
- Primary port policy applies to all secondary ports. No trunk is rejected.
- The trunk is rejected if any trunk port has mirroring or monitoring configured.
- The trunk is rejected if any trunk port has VLAN or inner-VLAN translation configured.

Layer 2 requirements

The trunk is rejected if the trunk ports:

- Do not have the same untagged VLAN component.
- Do not share the same SuperSpan Customer ID (CID).
- Do not share the same VLAN membership.
- Do not share the same uplink VLAN membership.
- Do not share the same protocol-VLAN configuration.
- Are configured as primary and secondary interfaces.

Layer 3 requirements

The trunk is rejected if any secondary trunk port has any Layer 3 configurations, such as IPv4 or IPv6 addresses, OSPF, RIP, RIPng, IS-IS, and so on.

Layer 4 (ACL) requirements

All trunk ports must have the same ACL configurations or the trunk is rejected. You can have a maximum of 128 server trunks and 80 switch trunks.

NOTE
These trunking rules are applicable for statically configured trunks as well as dynamic trunks created using LACP.

Common diagnostic scenarios

- When adding a new port to a 4-port LACP trunk, the existing ports go down and come back up in order to establish the fifth port. For existing 5-port trunks, when one port flaps, the other ports in the trunk will not flap.
- LACP trunk links may not operate properly between Brocade devices and third-party devices because of a mismatch between the link configurations. If the link is fixed on the third-party side, the link on the Brocade side must be a trunk. If it is link-aggregated on the third-party side, then it needs to be the same on the Brocade side.
- When adding new ports to a static or dynamic (LACP) trunk, you must reconfigure the Link Aggregation Group (LAG).
- LACP links may not operate properly due to misconfigurations. Contact Brocade Technical Support for help with configuration issues.

UDLD

Unidirectional Link Detection (UDLD) monitors a link between two routers to quickly detect link failures. UDLD 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 individual port links and for trunk links.

UDLD show commands

This section describes the show commands that display UDLD information.

show link-keepalive

Syntax: show link-keepalive

This command displays UDLD information for all ports.

```
Brocade# show link-keepalive
Total link-keepalive enabled ports: 4
Keepalive Retries: 5 Keepalive Interval: 1 Sec.
Port     Physical Link     Link-keepalive    Logical Link
4/1      up                up                up
4/2      up                up                up
4/3      down              down              down
4/4      up                down              down
```

show interface brief

Syntax: show interface brief

This command displays concise information if a port is disabled by UDLD, as shown in the following example.

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

show link-keepalive ethernet

Syntax: show link-keepalive ethernet *slot/portnum*

This command displays detailed UDLD information for a specific port, as shown in the following example.

```
Brocade# show link-keepalive ethernet 4/1
Current State: up Remote MAC Addr: 0000.00d2.5100
Local Port: 4/1 Remote Port: 2/1
Local System ID: e0927400 Remote System ID: e0d25100
Packets sent: 254 Packets received: 255
Transitions: 1
```

UDLD debug commands

There are no debug commands specific to UDLD.

Clearing UDLD statistics

To clear UDLD statistics, enter the following command.

clear link-keepalive statistics

Syntax: clear link-keepalive statistics

This command clears the packets sent, packets received, and transitions counters in the **show link-keepalive ethernet** *slot/portnum* display.

Configuration notes

- UDLD is supported only on Ethernet ports.
- To configure UDLD on a trunk group, you must configure the feature on each port of the group individually.
- Configuring UDLD on the primary port of a trunk group enables the feature on that port only.
- Dynamic trunking is not supported. If you want to configure a trunk group that contains ports where UDLD is enabled, you must remove the UDLD configuration from the ports. After you create the trunk group, you can add the UDLD configuration again.

- When a specified number of port failures occurs, UDLD can bring the port down. Syslog messages are generated when this occurs.
- Once the UDLD holddown threshold for a port is reached, a warning message is generated and written to the system log. The message means that UDLD is blocked on the specified port until the holddown counter is cleared for the port.

Common diagnostic scenarios

- Port flapping occurs when UDLD is configured.

  Port flapping can occur because of a problem with a dirty optic module connection or other hardware issue.
- UDLD may bring a port down (port blocked) because of a malformed.
- UDLD packets may be sent from a port, but not received on a port.

  This may be a hardware issue with the port or the module.
- UDLD links may not operate properly between Brocade devices and third-party devices.

  UDLD is proprietary and not compatible between vendors. Contact Brocade Technical Support for assistance.
- UDLD interfaces may flap during a write memory exercise.

  CPU usage may be too high. Contact Brocade Technical Support for assistance.
- UDLD port numbers on remote devices may be reported incorrectly (wrong ID number) by the Brocade device.

  The remote port number reflects the port ID sent by the remote router or switch and interpreted by the local router. In cases where the local router interprets the port ID differently than the remote, the port ID shown may be incorrect.

VLAN Translation

VLAN Translation allows traffic from one VLAN to be transported across a different VLAN. Packet VLAN IDs from the original VLAN are changed at the ingress port of the translating VLAN. When the packets reach the egress point on the translating VLAN, the VLAN ID is translated back to the original ID.

This feature is useful for service providers who must carry traffic from different customers across their network while preserving the VLAN ID and priority information of the customer network.

VLAN Translation show commands

This section describes the show command that displays VLAN translation information.

show vlan

Syntax: **show vlan** *vlan-id*

This command displays global VLAN translation information, as shown in the following example.

```
Brocade# show vlan
Configured PORT-VLAN entries: 3
Maximum PORT-VLAN entries: 4090
Default PORT-VLAN id: 1
PORT-VLAN 1, Name DEFAULT-VLAN, Priority Level0
L2 protocols: NONE
Untagged Ports: ethernet 2/1 to 2/20 ethernet 3/1 to 3/20 ethernet
PORT-VLAN 2, Name [None], Priority Level0
L2 protocols: NONE
ip-protocol VLAN, Dynamic port disabled
Name: basic
PORT-VLAN 1001, Name [None], Priority Level0
L2 protocols: MRP
Tagged Ports: ethernet 3/1 ethernet 3/12 to 3/13 ethernet 3/20
Bytes received: 6000
```

To display information for a specific VLAN, enter a VLAN ID.

VLAN Translation debug commands

The debug commands described here apply to the management module processor.

debug vlan-translation

Syntax: [**no**] **debug vlan-translation** [**aging** | **cam** | **error** | **generic** | **packet**]

- **aging** - Displays aging information for a translation session entry.
- **cam** - Displays information about Content Addressable Memory (CAM) or Parameter Random Access Memory (PRAM) programming.
- **error** - Displays VLAN Translation errors.
- **generic** - Displays generic VLAN Translation information.
- **packet** - Displays information about VLAN Translation packet processing.

This command generates information about VLAN Translation activity.

debug vlan-translation cam

Syntax: [**no**] **debug vlan-translation cam**

This command generates information about VLAN CAM activity. Command output resembles the following example.

```
Brocade# debug vlan-translation cam
VLAN-ID-TRANS: Check for CAM optimization
```

This example indicates that a CAM optimization check has been done for a one-to-one VLAN Translation tunnel.

debug vlan-translation generic

Syntax: [**no**] **debug vlan-translation generic**

This command generates generic VLAN Translation information. Command output resembles the following example.

```
Brocade# debug vlan-translation generic
VLAN-ID-TRANS: send IPC message VLAN_TRANS_GRP_UPDATE: vtg id 1, entry index 10,
vlan id 100, vtg fid 0x0002, port mask 0, no 0
```

This example indicates that a configuration update change for a VLAN Translation group (vtg 1) has been sent to the line cards.

VSRP

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

VSRP provides Layer 2 redundancy, meaning that Layer 2 links are backed up. You can configure VSRP to provide redundancy for Layer 2 only or also for 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.

VSRP show commands

This section describes the show commands that display VSRP information.

show log

Syntax: show log

This command displays log messages, as shown in the following example.

```
Brocade#show log
Syslog logging: enabled (0 messages dropped, 1 flushes, 0 overruns)
    Buffer logging: level ACDMEINW, 5 messages logged
    level code: A=alert C=critical D=debugging M=emergency E=error
                I=informational N=notification W=warning

Dynamic Log Buffer (50 lines):
I:VSRP: VLAN 5 VRID 5 - transition to Backup
I:VSRP: VLAN 5 VRID 5 - transition to Master
I:VSRP: VLAN 5 VRID 5 - transition to Master-Confirm
I:VSRP: VLAN 5 VRID 5 - transition to Backup
I:System: Interface ethernet 1/6, state up
```

This log output shows all of the events that have taken place.

show vsrp

Syntax: show vsrp

This command displays VSRP diagnostic information, as shown in the following example.

```
Brocade# show vsrp
VLAN 5
 Auth-type no authentication
 VRID 5
 ========
  State           Administrative-status Advertise-backup Preempt-mode
  Initialize      Enabled               Disabled         True

  Parameter        Configured Current    Unit/Formula
  Priority         150        1          (150-0)*(0.0/1.0)
  Hello-interval   1          1          sec/10
  Dead-interval    3          3          sec/10
  Hold-interval    3          3          sec/10
  Initial-ttl      2          2          hops

  Member ports:      ethe 1/6
  Operational ports: None
info - vsrp_set_init() - init for vrid 5
info - vsrp_set_backup() - instance 5 set to backup state
info - vsrp_set_master_confirm() - vrid 5 set to master confirm
```

In this example, VRID 5 is in the initialization state, with a priority of 150. Because the current master's priority is lower than 150, VRID 5 changes from backup to master.

In the following section of output, the new master receives a Hello message with a higher priority (200) and reverts to backup. In the output, several VSRP PDUs are received, but none with a priority higher than 200, so the new master remains the master.

```
VSRP PDU received - vlan 5 - port 1/6

ver:2 type:1 vrid:5 pri:200 #ip:0 aut:0
        adv:1 dea:3 hld:3 ttl:2 scl:10 chksum:0x11ecApr 26 16:09:13
event:
==========
info - vsrp_set_backup() - instance 5 set to backup state

VSRP PDU received - vlan 5 - port 1/6

ver:2 type:3 vrid:5 pri:200 #ip:0 aut:0
        adv:1 dea:3 hld:3 ttl:2 scl:10 chksum:0x0fecApr 26
16:09:14======================================
```

VSRP debug commands

This section describes how to use debug commands to monitor VSRP environments.

debug vsrp

Syntax: [**no**] **debug vsrp** [**all** | **aware** | **error** | **events** | **packets** | **show** | **state** | **verbose** | **vlan** *vlan id*]

- **all** - Displays information about all VSRP events.
- **aware** - Displays information about VSRP aware.
- **error** - Displays VSRP errors.
- **events** - Displays VSRP events.

- **packets** - Displays information about VSRP packets.
- **show** - Displays VSRP debug parameters.
- **state** - Displays VSRP state changes.
- **verbose** - Displays VSRP information in verbose mode.
- **vlan** *vlan id* - Displays VSRP debug information for a specific VLAN.

This command generates information about VSRP activity. All VSRP routers go to the backup state when they are first reloaded. Command output is similar to the following, where router 1 goes to the master-confirm and master states. For Layer 3 VSRP, the master sends out the gratuitous ARP for the virtual router IP address, and then sends out the Hello packet as Layer 2 VSRP.

```
Brocade# debug vsrp all
VSRP debugging is setup for all attributes
Brocade# debug vsrp
    VSRP: debugging is on

May 17 11:06:18 VSRP: VLAN VLAN: 100, VRID 1, State BACKUP
May 17 11:06:25 VSRP: Packet received on port 1/6, vlan VLAN: 100
    ver:2 type:1 vrid:1 pri:111 #1p:1 aut:0
    adv:1 dea:3 hld:3 ttl:2 scl:10 chksum:0xa93c 10.53.5.1
May 17 11:06:35 VSRP: Packet received on port 1/6, vlan VLAN: 100
    ver:2 type:1 vrid:1 pri:111 #ip:1 aut:0
    adv:1 dea:3 hld:3 ttl:2 scl:10 chksum:0xa93c 10.53.5.1
May 17 11:06:42 VSRP: VLAN VLAN:100, VRID 1, State MASTER-CONFIRM
May 17 11:06:42 VSRP: vlan VLAN:10, VRID 10, state MASTER
    ver:2 type:1 vrid:10 pri:110 #ip:1 aut:0
    adv:1 dea:3 hld:3 ttl:2 scl:10 chksum:0xaa3c 10.53.5.1
    ver:2 type:1 vrid:10 pri:110 #ip:1 aut:0
    adv:1 dea:3 hld:3 ttl:3 scl:10 chksum:0xaa3c 10.52.5.1
May 17 11:07:04 VSRP: vlan VLAN:100, VRID 1 - send ARP requrest for ip 10.53.5.1
    ver:2 type:1 vrid:10 pri:110 #ip aut:0
```

Configuration notes

- The **delay-link-event** command delays the sending of port up or down events to Layer 2 protocols. If VSRP is enabled on the port, the ownership will not change until the port status has remained up or down for the configured amount of time to ensure that minor transient states of a link do not unintentionally cause a disruptive topology change in the network.
- To provide Layer 3 redundancy only, disable VSRP and use VRRP-E.
- When you configure VSRP, make sure each non-VSRP Brocade device has a separate link to each of the VSRP devices.

Common diagnostic scenarios

- Layer 2 VSRP packets may be mislabeled and have incorrect checksums.

  This occurs because Brocade uses the same packet format for Layer 2 VSRP and Layer 3 VSRP for consistency. The Layer 3 and Layer 4 fields in the packet are not used as they do not have any relevance in Layer 2 forwarding and processing. When a Layer 2 VSRP packet is received by the Brocade switch, it is parsed, interpreted as a VSRP packet, and processed. The Layer 2 VSRP packet is not a routed IP packet. It is a Layer 2 switched packet using an ether-type of 0x800 (IP). No router can route this packet, as the destination MAC of Layer 2 VSRP does not belong to any router's MAC address. So this packet only has relevance within the VLAN.

- VSRP may not work correctly if the host route table is not pointing to the correct default gateway.
- When VSRP is configured on a Brocade NetIron XMR router, it does not advertise the virtual MAC to the attached Layer 2 switch, and the default gateway (VSRP VRID IP-Address) is not reachable.
- A Layer 2 hitless upgrade causes MRP and VSRP ports to flap.

  VSRP was formerly not supported for hitless software upgrades. The MRP port state change log messages are normal and are generated by the new active management module, which now defines the roles of each port. This issue occurs because of old software.

VPORT Scaling

Virtual Ports (VPORTs) are port-VLAN abstractions as well as trunk port abstractions. VPORTs allow independent port states for VLANs of the same physical port and allow multiple ports to appear as a single port in the case of trunking. VPORTs control Layer 2 protocols associated with the VLAN port. To improve VPORT scalability up to 32k, optimization of ITC, IPC, and hardware updates are done, based on the effective VPORT (EVP) count. The EVP count is the number of VLAN and port combination in the system.

VPORT Scaling show commands

This section describes the show command that displays detailed information about the VLAN.

show vlan detail

Syntax: show vlan detail

This command displays detailed information about VLAN states, port types, port modes, actual VPORT (AVP) count, EVP count, and control protocols configured on the VLAN, as shown in the following example.

The AVP count is the number of VPORTs configured in the system, whereas the EVP count is the number of VLAN and port combination in the system.

```
Brocade# show vlan detail
Untagged Ports: ethernet 2/1 to 2/20 ethernet 4/4
Tagged Ports: None
Dual-mode Ports: ethernet 3/1 to 3/20jjj ethernet 4/1 to 4/3
Default VLAN: 1
Control VLAN: 4095
VLAN Tag-type: 0x8100
AVP Count       : 24
EVP Count       : 28
PORT-VLAN 1, Name DEFAULT-VLAN, Priority Level0
----------------------------------------------------------
Port   Type       Tag-Mode   Protocol   State
2/1    PHYSICAL   UNTAGGED   NON        DISABLED
2/2    PHYSICAL   UNTAGGED   NONE       DISABLED
2/3    PHYSICAL   UNTAGGED   NONE       DISABLED
2/4    PHYSICAL   UNTAGGED   NONE       DISABLED
2/5    PHYSICAL   UNTAGGED   NONE       DISABLED
.
. (output edited for brevity)
.
4/1    PHYSICAL   UNTAGGED   NONE       FORWARDING
4/2    PHYSICAL   UNTAGGED   NONE       FORWARDING
4/3    PHYSICAL   UNTAGGED   NONE       FORWARDING
4/4    PHYSICAL   UNTAGGED   NONE       DISABLED
PORT-VLAN 100, Name [None], Priority Level0
----------------------------------------------------------
Port   Type       Tag-Mode   Protocol   State
```

LLDP

Link Layer Discovery Protocol (LLDP) enables a station to advertise its capabilities and to discover other LLDP-enabled stations in the same 802 LAN. An LLDP agent can transmit and receive information to and from another LLDP agent located on an adjacent device, but it cannot solicit information from another LLDP agent, nor can it acknowledge information received from another LLDP agent.

LLDP debug commands

This section describes the LLDP-related debug commands.

debug lldp

Syntax: debug lldp

This command displays information about LLDP port state changes. Command output resembles the following example.

```
Brocade# debug lldp
          LLDP:  All Ports: debugging is on
Nov 17 19:29:37 LLDP port 1/1 : Tx INIT -> INIT ; Rx WAIT_PORT_OPER ->
WAIT_PORT_OPER
Nov 17 19:29:37 LLDP: Ethernet: 1/1  tx_enabled = 1 rx_enabled = 1
Nov 17 19:29:37 LLDP port 1/2 : Tx INIT -> INIT ; Rx WAIT_PORT_OPER ->
WAIT_PORT_OPER
Nov 17 19:29:37 LLDP: Ethernet: 1/2  tx_enabled = 1 rx_enabled = 1
```

```
Nov 17 19:29:37 LLDP port 1/3 : Tx INIT -> IDLE ; Rx INIT -> WAIT_FOR_FRAME
Nov 17 19:29:37 LLDP: Ethernet: 1/3  tx_enabled = 1 rx_enabled = 1
Nov 17 19:29:44 LLDP event : Packet recieved on port: 1/1
Nov 17 19:29:44 LLDP event : Packet recieved on port: 1/2
Nov 17 19:29:44 LLDP event : Packet recieved on port: 1/3
Nov 17 19:30:08 LLDP event : poll timer triggered
Nov 17 19:30:08 LLDP event : poll timer triggered, calling timer handler
Nov 17 19:30:08 LLDP event : Packet sent from port: 1/1
Nov 17 19:30:08 LLDP event : called transmit PDU on port 1/1
Nov 17 19:30:08 LLDP event : poll timer triggered
Nov 17 19:30:08 LLDP event : poll timer triggered, calling timer handler
Nov 17 19:30:08 LLDP event : Packet sent from port: 1/2
Nov 17 19:30:08 LLDP event : called transmit PDU on port 1/2
Nov 17 19:30:08 LLDP event : poll timer triggered
Nov 17 19:30:08 LLDP event : poll timer triggered, calling timer handler
Nov 17 19:30:08 LLDP event : Packet sent from port: 1/3
Nov 17 19:30:08 LLDP event : called transmit PDU on port 1/3
```

debug lldp port

Syntax: **debug lldp port ethernet** *slot/port*

The **ethernet** *slot/port* parameter limits the display of LLDP port state change information to the specific Ethernet port. Command output resembles the following example.

```
Brocade# debug lldp port ethernet 1/1
LLDP debug port set to 1/1.
Nov 17 19:29:37 LLDP port 1/1 : Tx INIT -> INIT ; Rx WAIT_PORT_OPER ->
WAIT_PORT_OPER
Nov 17 19:29:37 LLDP: Ethernet: 1/1  tx_enabled = 1 rx_enabled = 1
Nov 17 19:29:44 LLDP event : Packet recieved on port: 1/1
Nov 17 19:30:08 LLDP event : poll timer triggered
Nov 17 19:30:08 LLDP event : poll timer triggered, calling timer handler
Nov 17 19:30:08 LLDP event : Packet sent from port: 1/1
Nov 17 19:30:08 LLDP event : called transmit PDU on port 1/1
```

MMRP

Multiple MAC Registration Protocol (MMRP) allows dynamic registration of group MAC address or individual MAC addresses among the end stations and bridges in the same Local Area Network (LAN). MAC address registration information controls transfer of frames destined for the group MAC address to be forwarded only to the registered group members.

MMRP debug commands

This section describes the MMRP-related debug commands.

debug mmrp

Syntax: [**no**] **debug mmrp** [**cli** | **config** | **db-event** | **error-event** | **ethernet** | **event** | **itc** | **pdu** | **reset** | **rx-event** | **show** | **sm-event** | **timer** | **tx-event** | **verbose**]

- **cli** - Enables or disables MMRP CLI debugging.
- **config** - Enables or disables MMRP configuration debugging.
- **db-event** - Enables or disables MMRP database event debugging.

- **error-event** - Enables or disables MMRP error event debugging.
- **ethernet** - Enables or disables MMRP port debugging.
- **event** - Enables or disables MMRP event debugging.
- **itc** - Enables or disables MMRP inter-task communication (ITC) debugging.
- **pdu** - Enables or disables Multiple MAC Registration Protocol data unit (MMRPDU) message debugging.
- **reset** - Resets all the MMRP debugging parameters to default.
- **rx-event** - Enables or disables MMRP receive event debugging.
- **sm-event** - Enables or disables MMRP state machine event debugging.
- **timer** - Enables or disables MMRP timer debugging.
- **tx-event** - Enables or disables MMRP transmit event debugging.
- **verbose** - Enables or disables the MMRP verbose debugging mode.
- **show** - Displays the current MMRP debugging parameters.

debug mmrp show

Syntax: **debug mmrp show**

This command displays the current MMRP debugging parameters. Command output resembles the following example.

```
Brocade# debug mmrp show
MMRP Debug Parameters
--------------------
MMRP  debugging is OFF [Mode: Brief]
Event: OFF
PDU Tx: OFF
PDU Rx: OFF
PDU Error: OFF
Timer: OFF
CLI: OFF
Config: OFF
ITC: OFF
Rx-Event: OFF
Tx-Event: OFF
Db-Event: OFF
Error-Event: OFF
State-machine Event: OFF
```

MVRP

Multiple VLAN Registration Protocol (MVRP) allows the propagation of VLAN configuration information from one MVRP-aware switch to other MVRP-aware switches. With MVRP, an access switch is manually configured with all the desired VLANs for the network, and all the other switches on the network learn the VLAN configuration information dynamically. This avoids unnecessary flooding and provides a solution for better resource utilization and bandwidth conservation.

MVRP debug commands

This section describes the MVRP-related debug commands.

debug mvrp

Syntax: [**no**] **debug mvrp** [**cli** | **config** | **db-event** | **error-event** | **ethernet** | **event** | **itc** | **pdu** | **reset** | **rx-event** | **show** | **sm-event** | **timer** | **tx-event** | **verbose**]

- **cli** - Enables or disables MVRP CLI debugging.
- **config** - Enables or disables MVRP configuration debugging.
- **db-event** - Enables or disables MVRP database event debugging.
- **error-event** - Enables or disables MVRP error event debugging.
- **ethernet** - Enables or disables MVRP port debugging.
- **event** - Enables or disables MVRP event debugging.
- **itc** - Enables or disables MVRP inter-task communication (ITC) debugging.
- **pdu** - Enables or disables Multiple VLAN Registration Protocol data unit (MVRPDU) message debugging.
- **reset** - Resets all the MVRP debugging parameters to default.
- **rx-event** - Enables or disables MVRP receive event debugging.
- **show** - Displays the current MVRP debugging parameters.
- **sm-event** - Enables or disables MVRP state machine event debugging.
- **timer** - Enables or disables MVRP timer debugging.
- **tx-event** - Enables or disables MVRP transmit event debugging.
- **verbose** - Enables or disables the MVRP verbose debugging mode.

The **debug MVRP** command enables or disables MVRP-related debugging events.

debug mvrp show

Syntax: debug mvrp show

This command displays the current MVRP debugging parameters. Command output resembles the following example.

```
Brocade# debug mvrp show
MVRP Debug Parameters
--------------------
MVRP ethe 1/1 to 1/2  debugging is OFF [Mode: Brief]
 Event: OFF
 PDU Tx: OFF
 PDU Rx: OFF
 PDU Error: OFF
 Timer: OFF
 CLI: OFF
 Config: OFF
 ITC: OFF
 Rx-Event: OFF
 Tx-Event: OFF
 Db-Event: OFF
 Error-Event: OFF
 State-machine Event: OFF
```

MMRP

Multiple MAC Registration Protocol (MMRP) allows dynamic registration of group MAC address or individual MAC addresses among the end stations and bridges in the same Local Area Network (LAN). MAC address registration information controls transfer of frames destined for the group MAC address to be forwarded only to the registered group members.

MMRP debug commands

This section describes the MMRP-related debug commands.

debug mmrp

Syntax: [**no**] **debug mmrp** [**cli** | **config** | **db-event** | **error-event** | **ethernet** | **event** | **itc** | **pdu** | **reset** | **rx-event** | **show** | **sm-event** | **timer** | **tx-event** | **verbose**]

- **cli** - Enables or disables MMRP CLI debugging.
- **config** - Enables or disables MMRP configuration debugging.
- **db-event** - Enables or disables MMRP database event debugging.
- **error-event** - Enables or disables MMRP error event debugging.
- **ethernet** - Enables or disables MMRP port debugging.
- **event** - Enables or disables MMRP event debugging.
- **itc** - Enables or disables MMRP inter-task communication (ITC) debugging.
- **pdu** - Enables or disables Multiple MAC Registration Protocol data unit (MMRPDU) message debugging.
- **reset** - Resets all the MMRP debugging parameters to default.
- **rx-event** - Enables or disables MMRP receive event debugging.
- **sm-event** - Enables or disables MMRP state machine event debugging.
- **timer** - Enables or disables MMRP timer debugging.
- **tx-event** - Enables or disables MMRP transmit event debugging.
- **verbose** - Enables or disables the MMRP verbose debugging mode.
- **show** - Displays the current MMRP debugging parameters.

debug mmrp show

Syntax: **debug mmrp show**

This command displays the current MMRP debugging parameters. Command output resembles the following example.

```
Brocade# debug mmrp show
MMRP Debug Parameters
--------------------
MMRP  debugging is OFF [Mode: Brief]
Event: OFF
PDU Tx: OFF
PDU Rx: OFF
PDU Error: OFF
Timer: OFF
CLI: OFF
```

```
Config: OFF
ITC: OFF
Rx-Event: OFF
Tx-Event: OFF
Db-Event: OFF
Error-Event: OFF
State-machine Event: OFF
```

Chapter

MPLS Diagnostics 5

In this chapter

This chapter contains diagnostic information for the MPLS configurations.

MPLS

Multiprotocol Label Switching (MPLS) debug and show commands help the users to diagnose and determine the cause of faults for MPLS-related features.

The following sections describe show and debug commands that can be used to obtain information about MPLS activity.

MPLS show commands

This section describes the show commands that display MPLS information.

show mpls bypass-lsp

Syntax: **show mpls bypass-lsp** [**detail**]

This command displays detailed information about the MPLS bypass LSP, as shown in the following example.

```
Brocade# show mpls bypass-lsp detail
LSP b1, to 10.2.2.2
  From: 10.1.1.1, admin: UP, status: UP, tunnel interface(primary path): tnl1
  Times primary LSP goes up since enabled: 1
  Metric: 0, number of installed aliases: 0
  Maximum retries: 0, no. of retries: 0
  Pri. path: NONE, up: yes, active: yes
  Setup priority: 7, hold priority: 0
  Max rate: 0 kbps, mean rate: 0 kbps, max burst: 0 bytes
  Constraint-based routing enabled: yes
  Tie breaking: random, hop limit: 0
  LDP tunneling enabled: no
  Active Path attributes:
   Tunnel interface: tnl1, outbound interface: e1/4
   Tunnel index: 2, Tunnel instance: 1 outbound label: 3
   Path calculated using constraint-based routing: yes
```

```
   Path calculated using interface constraint: no
   Recorded routes:
    Protection codes: P: Local  N: Node  B: Bandwidth  I: InUse
    10.30.30.1
  exclude interface(s): e1/1
  Tunnel bandwidth
  Maximum BW: 0 kbps
  Reservable BW [priority] kbps:
    [0] 0    [1] 0    [2] 0    [3] 0
    [4] 0    [5] 0    [6] 0    [7] 0

  Riding ingress backup lsps: 1
  State   LSPname
  UP      to_DUT2
 Number of active riding ingress backup lsps: 0
 Number of inactive riding ingress backup lsps: 1
```

show mpls interface

Syntax: **show mpls interface** [**brief** | **ethernet** *slotnum/portnum* | **pos** *slotnum/portnum* | **tunnel** *tunnel-id* | **ve** *num*]

- **brief** - Displays information about the MPLS interface in brief.
- **ethernet** *slotnum/portnum* - Displays MPLS interface information for the specified Ethernet interface.
- **pos** *slotnum/portnum* - Displays MPLS interface information for the specified POS interface.
- **tunnel** *tunnel-id* - Displays MPLS interface information for the specified tunnel interface.
- **ve** *num* - Displays MPLS interface information for the specified virtual interface.

This command displays information about the MPLS interface. For an Ethernet interface, the command output resembles the following example.

```
Brocade# show mpls interface ethernet 1/1
  Admin: Up  Oper: Up
  Maximum BW: 10000000 kbps, maximum reservable BW: 10000000 kbps
  Admin group: 0x00000000
  Reservable BW [priority] kbps:
    [0] 10000000    [1] 10000000    [2] 10000000    [3] 10000000
    [4] 10000000    [5] 10000000    [6] 10000000    [7] 10000000
  Last sent reservable BW [priority] kbps:
    [0] 10000000    [1] 10000000    [2] 10000000    [3] 10000000
    [4] 10000000    [5] 10000000    [6] 10000000    [7] 10000000
  Configured Protecting bypass lsps: 1
    b1(UP)
```

The following example shows information in brief, such as the interface name, maximum bandwidth on the interface, administration group, administration status, operational status, and so on.

```
Brocade# show mpls interface brief
Interface  AdminGrp  Admin Oper MaxBW      MaxResvBW  BypsLSPcnt
     e1/1 0x00000000 Up    Up   10000000   95000000   0
     e1/2 0x00000000 Up    Up   10000000   10000000   0
     e1/3 0x00000000 Up    Up   10000000   10000000   0
```

show mpls memory

Syntax: **show mpls memory**

This command displays information about MPLS memory usage, as shown in the following example.

```
Brocade# show mpls memory
 Id SubId GenSize BlockSize GenBlocks CurrGens CurrBlocks FreeBlocks
---------------------------------------------------------------------
  1    0    10820       108       100        1        100         96
  1    1    33900       308       110        1        110        103
  1    2    10060      1004        10        1         10          5
  1    3     9032      3004         3       2          6         5
Total Allocated = 72844 bytes
Total Available = 62132 bytes
 Id SubId GenSize BlockSize GenBlocks CurrGens CurrBlocks FreeBlocks
---------------------------------------------------------------------
  2    0     8340        20       416        1        416        378
  2    1     9276        52       178        3        534         83
  2    2     9512        84       113        2        226        112
  2    3     9656       132        73        2        146         43
  2    4     9796       188        52        3        156         13
  2    5     9780       244        40        1         40         35
  2    6     9632       356        27        2         54         21
  2    7     9880       580        17        2         34          8
  2    8     9980       996        10        1         10          2
  2    9    10000      1996         5        9         45          5
Total Allocated = 252676 bytes
Total Available = 62032 bytes
Total non-pool memory: 529514 bytes
Mem-Type   Alloc  BytesAlloc TotalAlloc TotalFree  AllocPeak  AllocFail  FreeFail
Misc       11     1413432    11         0          11         0          0
Dbg-Cntr   102    3264       102        0          102        0          0
IF-Entry   0      0          1          1          1          0          0
RLDF-If    0      0          1          1          1          0          0
TNNL-Vif   1      884736     1          0          1          0          0
CORE-MPLS  122    11097734   129        7          122        0          0
Total memory allocated by MPLS: 13399166 bytes
```

show mpls rsvp interface

Syntax: **show mpls rsvp interface** [**brief** | **detail** | **ethernet** *slotnum/portnum* | **pos** *slotnum/portnum* | **ve** *num*]

- **brief** - Displays a brief summary of the RSVP interface.
- **detail** - Displays detailed information about the RSVP interface.
- **ethernet** *slotnum/portnum* - Displays RSVP interface information for the specified Ethernet interface.
- **pos** *slotnum/portnum* - Displays RSVP interface information for the specified POS interface.
- **ve** *num* - Displays RSVP interface information for the specified virtual routing interface.

This command displays debug messages related to the RSVP interface, as shown in the following example.

```
Brocade# show mpls rsvp interface
                                                     # of OutSegs      # of
Interface State MD5 Auth ReliableMsg Bundle SRefresh Act/Inact/Resv    Preempts
     e1/1    Up    OFF      OFF        OFF    OFF        1/0/0         0
     e1/3    Up    OFF      OFF        OFF    OFF        0/0/0         0
     e1/4    Up    OFF      OFF        OFF    OFF        1/0/0         0
```

For an Ethernet interface, the command output resembles the following example.

```
Brocade# show mpls rsvp interface ethernet 1/1
Interface State MD5 Auth ReliableMsg Bundle SRefresh
     e1/1 Up    OFF       OFF         OFF    OFF
                     Total      Since last clear
 PacketType   Sent   Received   Sent   Received
  Path        2      3          2      3
  Resv        3      3          3      3
  PathErr     0      0          0      0
  ResvErr     0      0          0      0
  PathTear    0      0          0      0
  ResvTear    0      0          0      0
  ResvConf    0      0          0      0
  Bundle      0      0          0      0
  Ack         0      0          0      0
  SumRefresh  0      0          0      0

 Errors                 Total                 Since last clear
  Rcv MD5 Auth Errors   0                     0
```

show mpls rsvp session

Syntax: **show mpls rsvp session** [**backup** | **brief** | **bypass** | [**destination** *destination_ip_address* [**source** *source_ip_address* [**tunnel-id** *tunnel_id*]]] | **detail** | **detour** | **down** | **egress** | **extensive** | **in-interface** *interface* | **ingress** | **name** | **out-interface** *interface* | **transit** | **up** | **wide**]

- **backup** - Displays information about backup RSVP sessions.
- **brief** - Displays brief information about RSVP sessions.
- **bypass** - Displays information about bypass RSVP sessions.
- **destination** *destination_ip_address* - Displays information about RSVP sessions for the specified destination IP address.
- **source** *source_ip_address* - Displays information about RSVP sessions for the specified source IP address.
- **tunnel-id** *tunnel_id* - Displays information about RSVP sessions for the specified tunnel ID.

NOTE
The **source** *source_ip_address* option must be used along with the **destination** *destination_ip_address* option. The **tunnel-id** *tunnel_id* option must be used along with the **source** *source_ip_address* option.

- **detail** - Displays detailed information about RSVP sessions.
- **detour** - Displays information about detour RSVP sessions.
- **down** - Displays information about inactive RSVP sessions.
- **egress** - Displays information about egress RSVP sessions.
- **extensive** - Displays extensive information about RSVP sessions.
- **in-interface** *interface* - Displays information about RSVP sessions that are coming into an interface. The *interface* parameter can be an Ethernet, POS, or virtual interface.
- **ingress** - Displays information about ingress RSVP sessions.
- **name** - Displays RSVP session information for the specified session name.

- **out-interface** *interface* - Displays information about RSVP sessions that are going out of an interface. The *interface* can be an Ethernet, POS, or virtual interface.
- **transit** - Displays information about transit RSVP sessions.
- **up** - Displays information about active RSVP sessions.
- **wide** - Displays the full LSP name in a single line when used with other RSVP session options, such as **backup**, **detour**, **ingress**, and so on.

This command displays RSVP session information. Command output resembles the following example.

```
Brocade# show mpls rsvp session destination 10.2.2.2 source 10.1.1.1 tunnel-id 2
Codes: DI:Ingress Detour  DT:Transit Detour  DM:Merged Detour
       DE:Egress Detour  BI:Ingress Backup BM: Merged Backup BE:Egress Backup
       RP:Repaired Session BYI: Bypass Ingress
To              From                 St Style Lbl_In  Lbl_Out Out_If LSPname
10.2.2.2         10.1.1.1              Up SE    -       3       e1/1   to_DUT2
  Tunnel ID: 2, LSP ID: 1
  Time left in seconds (PATH refresh: 0, ttd: 26
                        RESV refresh: 3, ttd: 123)
  Tspec: peak 0 kbps rate 0 kbps size 0 bytes m 20 M 65535
  Set-up priority: 6Holding Priority: 3
  Fast Reroute: Facility backup desired
   Setup priority: 7, hold priority: 0
   Session attribute flags:0x17
   (Label recording,SE Style,Protection: Local,Node)
   Bandwidth: 0 kbps, hop limit: 255
   Backup LSP: UP.  Nexthop (link) protection available.
    Up/Down times: 1, num retries: 0
  Explicit path hop count: 1
   10.10.10.2 (S)
  Received RRO count: 1
   Protection codes/Rtr Id flag: P: Local  N: Node  B: Bandwidth  I: InUse R: Rt
rId
   10.10.10.2
  PATH sentto:  10.10.10.10      (e1/1           ) (MD5 OFF)
  RESV rcvfrom: 10.10.10.10      (e1/1           ) (MD5 OFF)

To              From                 St Style Lbl_In  Lbl_Out Out_If LSPname
10.2.2.2         10.10.10.10(BI)       Dn -     -       -       e1/4   to_DUT2
  Tunnel ID: 2, LSP ID: 1
  Time left in seconds (PATH refresh: 0, ttd: 4294835)
  Tspec: peak 0 kbps rate 0 kbps size 0 bytes m 20 M 65535
  Explicit path hop count: 1
   10.30.30.1 (S)
  PATH rcvfrom: None (downstream only)
  PATH sentto:  10.30.30.30      (e1/4           ) (MD5 OFF)
  Riding bypass lsp: b1
```

The following example shows RSVP sessions that are going out of Ethernet interface 1/1.

```
Brocade# show mpls rsvp session destination 10.2.2.2 out-interface ethernet 1/1
Codes: DI:Ingress Detour  DT:Transit Detour  DM:Merged Detour
       DE:Egress Detour  BI:Ingress Backup BM: Merged Backup BE:Egress Backup
       RP:Repaired Session BYI: Bypass Ingress
To              From                 St Style Lbl_In  Lbl_Out Out_If LSPname
10.2.2.2         10.1.1.1              Up SE    -       3       e1/1   to_DUT2
10.2.2.2         10.10.10.10(BI)       Dn -     -       -       e1/4   to_DUT2
```

show mpls ted database

Syntax: show mpls ted database [**brief** | **detail**]

- **brief** - Displays brief information about Traffic Engineering Database (TED).
- **detail** - Displays detailed information about TED.

This command displays information about TED that contains topology information about nodes in an MPLS domain and the links that connect them. Command output resembles the following example.

```
Brocade# show mpls ted database detail
NodeID: 10.20.20.20, Type: Router
info from applied local policies:
  cspf-group member information (name/penalty):
    group1/100
     Type: P2P, To: 10.1.1.1, Local: 10.1.1.2, Remote: 10.1.1.1, Gen 16
info from applied local policies:
    cspf-group member information (name/penalty):
      group2/10
```

show mpls ted path

Syntax: show mpls ted path *ip address* [**path-name** *path_name*]

This command displays information related to the TED path for the specified destination IP address and path name, as shown in the following example.

```
Brocade# show mpls ted path 10.2.2.2 path-name to2
Path to 10.2.2.2 found! Time taken to compute: 0 msec
Hop-count: 1 Cost: 1 OSPF Area Id: 0
        Hop 1: 10.10.10.2, Rtr 10.2.2.2
```

If the specified path name does not exist, the **show mpls ted path** command displays the following error message.

```
Brocade# show mpls ted path 10.2.2.2 path-name to_DUT8
No path found for destination 10.2.2.2 with path name to_DUT8
```

show interfaces tunnel

Syntax: show interfaces tunnel *tunnel_id*

This command displays the port status and configuration information for the tunnel interface. The following example shows GRE tunnel statistics such as the number of MPLS packets forwarded to and from the GRE tunnels, and so on.

```
Brocade# show interfaces tunnel 2
Tunnel1 is up, line protocol is up
  Hardware is Tunnel
  Tunnel source 10.2.2.2
  Tunnel destination is 10.4.4.4
  Tunnel mode gre ip
  Port name is LDP-GRE-1
  Internet address is: 10.1.1.1/24
  Tunnel TOS 0, Tunnel TTL 10, Tunnel MTU 600 bytes
  Keepalive is not Enabled

  Tunnel Packet Statistics:
                 Unicast Packets                  Multicast Packets
In-Port(s)    [Rcv-from-tnnl    Xmit-to-tnnl]    [Rcv-from-tnnl  Xmit-to-tnnl]
```

```
e5/25 - e5/48     63329            0                0              0
e6/1 - e6/2       0                10789735539      0              0
```

show ip-tunnels

Syntax: show ip-tunnels *tunnel_id*

This command displays tunnel interface information such as the status of the tunnel interface, tunnel source, tunnel destination, and so on. The following example shows information for a GRE tunnel interface.

```
Brocade# show ip-tunnels 2
GRE tnnl 2 UP   : src_ip 10.2.2.2, dst_ip 10.3.3.3
   TTL 255, TOS 0, NHT 1, MTU 1476
```

show mpls forwarding

Syntax: show mpls forwarding [*A.B.C.D or A.B.C.D/L* | **in-label** *num*]

- *A.B.C.D or A.B.C.D/L* - Displays MPLS forwarding information for a specific IP address.
- **in-label** *num* - Displays MPLS forwarding information for a specific incoming label.

This command displays MPLS forwarding information, as shown in the following example.

```
Brocade# show mpls forwarding
Total number of MPLS forwarding entries: 1
      Dest-prefix        In-lbl  Out-lbl Out-intf Sig Next-hop
1     10.44.44.44/32     -       3       e2/15    R   10.1.1.4
2     10.44.44.44/32     1024    3       e2/15    L   10.1.1.4
3     0.0.0.0/0          100     101     e3/14    S   10.1.1.2
4     0.0.0.0/0          101     102     e1/9     S   10.1.1.41
```

show mpls lsp

Syntax: show mpls lsp [[**brief** | **wide**| **debug** | **detail** | **extensive**] | [**up** | **down** [**wide** | **detail** | **extensive**]] | [**name** *lsp-name* [**extensive** | **debug**]]

- **brief** - Displays status information about signalled Label Switched Path (LSPs).
- **wide** - Displays the full LSP name in a single line.
- **debug** - Displays debug information about the LSP.
- **detail** - Displays detailed information about the LSP.
- **extensive** - Displays extensive information about the LSP.
- **up** - Displays information about active LSPs.
- **down** - Displays information about inactive LSPs.
- **name** *lsp-name* - Displays LSP information for the specified LSP name.

The **show mpls lsp brief** command output resembles the following example.

```
Brocade# show mpls lsp brief
*: The LSP is taking a Secondary Path

Name   To      Admin      Oper   Tunnel Up/Dn  Retry  Active
               State      State  Intf   Times  No.    Path
t1  10.3.3.3   UP         UP*    tnl1   1      5      v2
```

The **show mpls lsp wide** command output resembles the following example.

```
Brocade# show mpls lsp wide
```

```
Name                            To      Admin      Oper   Tunnel Up/Dn  Retry  Active
                                        State      State  Intf   Times  No.    Path
tunnel1                         10.3.3.3   UP      UP     tnl0   1      0
tunnel2                         10.3.3.3   UP      UP     tnl2   1      0      ppath1
tunnelfromsanfranciscotonewyork 10.3.3.3 UP        UP     tnl3   1      0
```

The **show mpls lsp detail** command output resembles the following example.

```
Brocade# show mpls lsp detail
LSP t1, to 10.3.3.3
Path selected: pathsecondaryb, mode: manual revert-timer:
Path selected is up for 3 seconds for the latest instance, traffic will be
switched
it in 7 seconds.
From: 10.2.3.4, admin: UP, status: UP, tunnel interface: tnl1
Times primary LSP goes up since enabled: 1
Metric: 1, number of installed aliases: 0
Maximum retries: 0, no. of retries: 3
Pri. path: dir, active: no
Setup priority: 7, hold priority: 0, ipmtu 1400
Max rate: 0 kbps, mean rate: 0 kbps, max burst: 0 bytes
Constraint-based routing enabled: yes
Tie breaking: random, hop limit: 0
Sec. path: v2, active: active
Hot-standby: no, status: up
Setup priority: 7, hold priority: 0
Max rate: 0 kbps, mean rate: 0 kbps, max burst: 0 bytes
Constraint-based routing enabled: yes
hop limit: 0
Active Path attributes:
Tunnel interface: tnl1, outbound interface: e1/1
Tunnel index: 5, outbound label: 1966
Path calculated using constraint-based routing: no
Explicit path hop count: 1
10.10.10.2 (S) -> 10.20.20.2 (S)
Recorded routes:
Protection codes: P: Local N: Node B: Bandwidth I: InUse
10.10.10.2 -> 10.20.20.2
BFD session status: UP
Config param: global, min-tx: 1000, min-rx: 1000, multiplier: 3
Negotiated tx-interval: 1000, rx-interval: 3000, multiplier: 3
Local discriminator: 1, remote discriminator: 1
```

The **show mpls lsp extensive** command displays extensive information such as hop type, hop address, protection flags, label flags, and label about the Label Switched Path (LSP). Command output resembles the following example for a regular LSP.

```
Brocade# show mpls lsp extensive
LSP regular, to 10.80.80.80
  From: 10.40.40.40, admin: UP, status: UP, tunnel interface(primary path): tnl2
  Times primary LSP goes up since enabled: 2
  Metric: 0, number of installed aliases: 0
  Maximum retries: NONE, no. of retries: 0
  Pri. path: 4578, up: yes, active: yes
   Setup priority: 7, hold priority: 0
   Max rate: 0 kbps, mean rate: 50 kbps, max burst: 0 bytes
   Constraint-based routing enabled: yes
   Path calculated using constraint-based routing: yes
   Path calculated using interface constraint: no
   Path cost: 3
   Tie breaking: random, hop limit: 0
   LDP tunneling enabled: no
```

```
  Soft preemption enabled: no
  Active Path attributes:
   Tunnel interface: tnl2, outbound interface: e1/1
   Tunnel index: 9, Tunnel instance: 1 outbound label: 1025
   Explicit path hop count: 3
    10.1.1.5 (S) -> 10.11.11.7 (S) -> 10.7.7.8 (S)
   Recorded routes:
    Protection codes: P: Local  N: Node  B: Bandwidth  I: InUse
    10.1.1.5 -> 10.11.11.7 -> 10.7.7.8
History
    0 Feb  6 08:47:26 : LSP tunnel is Enabled
    1 Feb  6 08:47:26 : CSPF-Computation successful for Primary path 4578. Route:
->10.1.1.5->10.11.11.7->10.7.7.8
    2 Feb  6 08:47:26 : Primary Path 4578. RRO received:
                        -> 10.1.1.5 -> 10.11.11.7 -> 10.7.7.8
    3 Feb  6 08:47:26 : Primary path 4578 setup successful . Instance id 1
    4 Feb  6 08:47:26 : LSP tunnel is UP with Primary path 4578 as Active
    5 Feb  6 08:47:26 : Tunnel added or updated, out-interface: e1/1, out-label
1024
    6 Feb  6 08:50:22 : Path 4578 received path-error from 10.7.7.7. Error code
24, Error value: 8:MPLS Service Break
    7 Feb  6 08:50:22 : Primary path 4578 torn down. Instance id 1.
    8 Feb  6 08:50:22 : LSP tunnel is DOWN
    9 Feb  6 08:50:22 : Tunnel deleted
   10 Feb  6 08:50:22 : CSPF-Computation failed for Primary path 4578. Error
3:Strict node error
   11 Feb  6 08:51:47 : CSPF-Computation failed for Primary path 4578. Error 1:No
mapping[9 times]
   12 Feb  6 08:52:10 : CSPF-Computation successful for Primary path 4578. Route:
->10.1.1.5->10.11.11.7->10.7.7.8
   13 Feb  6 08:52:10 : Primary Path 4578. RRO received:
                        -> 10.1.1.5 -> 10.11.11.7 -> 10.7.7.8
   14 Feb  6 08:52:10 : Primary path 4578 setup successful . Instance id 1
   15 Feb  6 08:52:10 : LSP tunnel is UP with Primary path 4578 as Active

   16 Feb  6 08:52:10 : Tunnel added or updated, out-interface: e1/1, out-label
1025
```

The **show mpls lsp up** command output resembles the following example.

```
Brocade# show mpls lsp up detail
LSP to_DUT2, to 10.2.2.2
  From: 10.1.1.1, admin: UP, status: UP, tunnel interface(primary path): tnl0
  Times primary LSP goes up since enabled: 1
  Metric: 0, number of installed aliases: 0
  Maximum retries: 0, no. of retries: 0
  Pri. path: to2, up: yes, active: yes
   Setup priority: 7, hold priority: 0
   Max rate: 0 kbps, mean rate: 0 kbps, max burst: 0 bytes
   Constraint-based routing enabled: yes
   Tie breaking: random, hop limit: 0
   LDP tunneling enabled: no
  Active Path attributes:
   Tunnel interface: tnl0, outbound interface: e1/1
   Tunnel index: 1, Tunnel instance: 1 outbound label: 3
   Path calculated using constraint-based routing: yes
   Path calculated using interface constraint: no
   Explicit path hop count: 1
    10.10.10.2 (S)
   Recorded routes:
    Protection codes: P: Local  N: Node  B: Bandwidth  I: InUse
```

```
    10.10.10.2
  Fast Reroute: facility backup desired
   Backup LSP: UP, out-label: 3, outbound interface: e1/4 bypass_lsp: b1
   FRR Forwarding State: Pri(active), Backup(up)
```

show mpls lsp name

Syntax: show mpls lsp name lsp-*name* [**extensive** | **debug**]

- The **name** lsp-*name* parameter specifies the name of the LSP for which you want to display information.
- **extensive** - Displays extensive information about the specified LSP.
- **debug** - Displays debug information about the specified LSP.

This command displays global revertiveness configuration and Fast Reroute (FRR) adaptive configuration information for the specified LSP name.

```
Brocade# show mpls lsp name tunnel1
LSP tunnel1, to 10.3.3.3
  From: 10.2.2.2, admin: UP, status: UP, tunnel interface(primary path): tnl0
  Times primary LSP goes up since enabled: 1
  Metric: 0, number of installed aliases: 0  Adaptive
  Maximum retries: NONE, no. of retries: 0
  Pri. path: p1, up: yes, active: yes
   Setup priority: 7, hold priority: 0
   Max rate: 0 kbps, mean rate: 0 kbps, max burst: 0 bytes
   Constraint-based routing enabled: yes
   Path calculated using constraint-based routing: yes
   Path calculated using interface constraint: yes
   Tie breaking: random, hop limit: 0
   LDP tunneling enabled: no
  Active Path attributes:
   Tunnel interface: tnl0, outbound interface: e1/1
   Tunnel index: 1, Tunnel instance: 2 outbound label: 3
   Explicit path hop count: 1
    10.10.10.6 (S)
   Recorded routes:
    Protection codes: P: Local  N: Node  B: Bandwidth  I: InUse
    10.10.10.6
  Fast Reroute: facility backup desired
   Setup priority: 0, hold priority: 0
   Hop Limit: 3
   Exclude any of admin groups: 2
   Include any of admin groups: 5 6
   Include all of admin groups: 5
   Backup LSP: UP, out-label: 3, outbound interface: e1/3 bypass_lsp: by1
  Path cspf-group computation-mode: disabled
   Global revertiveness enabled with hold time 20 secs
   FRR Forwarding State: Pri(active), Backup(up)
```

show mpls statistics label

Syntax: show mpls statistics label [*Label_num* | *slot/port*]

This command displays the MPLS statistics for the specified interface, as shown in the following example.

```
Brocade# show mpls statistics label 6/1
In-label   In-Port(s)   In-Packet Count
```

```
    1024  e6/1 - e6/2                0
    1025  e6/1 - e6/2      21652924531
```

The following is the sample output if the *Label_num* variable is specified.

```
Brocade# show mpls statistics label 2047
In-label     In-Port(s)   In-Packet Count     In-Byte Count     Rate(in kbps)
    2047   e1/1 - e1/24               131             11004              1000
    2047   e2/1 - e2/24                 0                --                --
    2047   e3/1 - e3/24                 0                --                --
```

show mpls debug flooding-thresholds

Syntax: **show mpls debug flooding-thresholds**

This command displays the current bandwidth in percentage and thresholds in use, as shown in the following example.

```
Brocade# show mpls debug flooding-thresholds
Interface : e4/3
   New Reserved BW percentage             : 40
   Old Reserved BW percentage(Advertised) : 35
   UP threshold Config in use             : Default
   DOWN threshold Config in use           : Default
   Last up threshold trigger              : 30
   Last down threshold trigger            : None
   Last threshold trigger was from        : UP threshold
```

show mpls debug counter

Syntax: **show mpls debug counter** [**aall** | **ai3** | **amb** | **ambl** | **amh** | **ase** | **asel** | **bfd** | **i3** | **ipl** | **ipr** | **nbase-root** | **ntl** | **rcp** | **rcs** | **rldf** | **rlm** | **rri** | **rric** | **rrt** | **rsip** | **rspx** | **rstc** | **sck** | **yads**]

- **aall** - Displays counters for the ACL library.
- **ai3** - Displays counters for the I3 library.
- **amb** - Displays counters for the MIB manager.
- **ambl** - Displays counters for the MIB library.
- **amh** - Displays counters for the MIB handler.
- **ase** - Displays counters for the system manager.
- **asel** - Displays counters for the system manager lite.
- **bfd** - Displays counters for the BFD stub.
- **i3** - Displays counters for the I3 stub.
- **ipl** - Displays counters for the IP library.
- **ipr** - Displays counters for the Route (RSIR) stub.
- **nbase-root** - Displays counters for NBASE-ROOT.
- **ntl** - Displays counters for the NTL library.
- **rcp** - Displays counters for LDP Path Manager.
- **rcs** - Displays counters for LDP Session Controller.
- **rldf** - Displays counters for the RLDF stub.
- **rlm** - Displays counters for the label manager.
- **rri** - Displays counters for RSVP.

- **rric** - Displays counters for the RRIC stub.
- **rrt** - Displays counters for TE-MIB.
- **rsip** - Displays counters for the IP stub.
- **rspx** - Displays counters for the proxy stub.
- **rstc** - Displays counters for CPCS LDT.
- **sck** - Displays counters for the socket stub.
- **yads** - Displays counters for CAM DS.

This command displays the debug counters. The debug counter keeps track of each debug statement access regardless of the match condition.

```
Brocade# show mpls debug counter
Counter-mnemonic            Severity    HitCount
PCT_IPR | 2, 0              Audit       1
PCT_IPR | 16, 0             Dev         12
PCT_IPR | 16, 1             Dev         4
PCT_IPR | 16, 20            Dev         1
PCT_ASE | 2, 0              Dev         122
PCT_ASE | 6, 0              Dev         32
PCT_ASE | 24, 0             Audit       13
PCT_ASE | 26, 0             Dev         44
PCT_RLM | 10, 0             Audit       1
PCT_RLM | 54, 0             Audit       1
PCT_RLM | 88, 1             Dev         311
PCT_RLM | 90, 1             Dev         311
PCT_RLM | 91, 1             Dev         311
PCT_RLDF | 33, 8            Dev         1
PCT_RLDF | 48, 0            Dev         311
PCT_RLDF | 49, 75           Dev         304
PCT_RSIP | 13, 0            Audit       2
PCT_RSIP | 23, 0            Dev         26
PCT_RRI | 177, 2            Exception   25
PCT_RRI | 291, 1            Dev         1587
PCT_RRI | 292, 1            Dev         1587
PCT_RRI | 293, 1            Dev         1472
PCT_RRI | 294, 1            Dev         1472
```

show np extended-counters usage slot

Syntax: show np extended-counters usage slot *slotnum* [**detail**]

This command displays information about the Network Processor (NP) usage of extended counters.

- *slotnum* - Specifies the slot for which you want to display information about the extended counters.
- **detail** - Displays detailed information about the extended counters.

Command output resembles the following example.

```
Brocade# show np extended-counters usage slot 5
Slot 5:
  Extended-Counters Mode:
    Routed-Switched: Seperate
    Per Priority: Enabled
  Number of Unique Statstics ID (port-vlan's) allowed per NP: 1023
  Number of Unique Statstics ID (port-vlan's) allocated:
    NP 1: 29
    NP 2: 42
```

show mpls debug rsvp igp-sync

Syntax: show mpls debug rsvp igp-sync [**brief**|**link** |**lsp** |**stats**]

This command displays debug information related to the RSVP Interior Gateway protocol (IGP) synchronization.

- **brief** - Displays all the RSVP IGP synchronization summary.
- **link** - Displays the RSVP IGP synchronization link information.
- **lsp** - Displays the RSVP IGP synchronization details of one or more LSPs.
- **stats** - Displays the RSVP IGP synchronization debug statistics.

show mpls debug rsvp igp-sync brief

Syntax: show mpls debug rsvp igp-sync brief

This command displays all the RSVP IGP synchronization summary. The following is a sample output from this command.

```
Brocade# show mpls debug rsvp igp-sync brief
Rsvp IGP Sync summary
 RSVP-OSPF Sync: Disabled
 RSVP-ISIS Sync: Enabled
```

show mpls debug rsvp igp-sync link

Syntax: show mpls debug rsvp igp-sync link [*A.B.C.D*| **detail**|**down-ind**]

- *A.B.C.D* - Specifies the link IP address.
- **detail** - Displays detailed RSVP IGP synchronization information for all link entries.
- **down-ind** - Initiates a link down event for OSPF or ISIS link taking the neighbor IP address, link IP address, and area ID as the input parameters.

The following is a sample output from the **show mpls debug rsvp igp-sync link** command.

```
Brocade# show mpls debug rsvp igp-sync link
Link Address     Remote Router    LSP Count
10.1.1.1         10.11.11.11              1
10.1.1.2         10.11.11.11              4
10.2.2.2         10.11.11.11              3
10.1.1.4         10.11.11.11              2
10.1.1.3         10.22.22.22              2
10.1.1.4         10.22.22.22              4
10.1.1.4         10.33.33.33              2
```

The following is a sample output from the **show mpls debug rsvp igp-sync link detail** command.

```
Brocade# show mpls debug rsvp igp-sync link detail
Link Address: 10.1.1.1
 Neighbor router Id: 10.11.11.11
 LSP paths using this IGP link: 1
 LSP List:
  lsp04-1:path_1_to_4,
Link Address: 10.1.1.2
 Neighbor router Id: 10.11.11.11
 LSP paths using this IGP link: 4
 LSP List:
  lsp04-1:path_1_to_4,  lsp04-63:path_1_to_4_6,  lsp04-63:path_1_to_4_6
  lsp04-28:path_1_to_4_strict,
```

```
Link Address: 10.2.2.2
 Neighbor router Id: 10.11.11.11
 LSP paths using this IGP link: 3
 LSP List:
  dbyp-12.1.1.1-1:,  lsp04-28:sec_path_1_to_4_1,  lsp04-29:sec_path_1_to_4_1
Link Address: 10.1.1.4
 Neighbor router Id: 10.11.11.11
 LSP paths using this IGP link: 2
 LSP List:
  lsp04-29:,  lsp04-11:,
Link Address: 10.1.1.3
 Neighbor router Id: 10.22.22.22
 LSP paths using this IGP link: 2
 LSP List:
  lsp04-1:path_1_to_4,  lsp04-28:path_1_to_4_strict,
Link Address: 10.1.1.4
 Neighbor router Id: 10.22.22.22
 LSP paths using this IGP link: 4
 LSP List:
  lsp04-63:path_1_to_4_6,  lsp04-63:path_1_to_4_6,  lsp04-28:sec_path_1_to_4_1
  lsp04-29:sec_path_1_to_4_1,
Link Address: 10.1.1.4
 Neighbor router Id: 10.33.33.33
 LSP paths using this IGP link: 2
 LSP List:
  lsp04-1:path_1_to_4,  lsp04-28:path_1_to_4_strict,
```

The following is a sample output from the **show mpls debug rsvp igp-sync down-ind** command.

```
Brocade# show mpls debug rsvp igp-sync down-ind ospf 10.1.1.3 10.1.1.4
255.255.255.255
Generating Rsvp IGP Sync link down event for:
  IGP: ISIS
  Neighbor IP: 10.1.1.3
  Link IP: 10.1.1.4
  Area Id: 255.255.255.255
```

show mpls debug rsvp igp-sync lsp

Syntax: **show mpls debug rsvp igp-sync lsp** [*lsp_name* | **detail**]

- *lsp_name* - Specifies the LSP name.
- **detail** - Displays RSVP IGP synchronization details of all LSPs.

This command displays RSVP IGP synchronization details of one or more LSPs. The following is a sample output from the **show mpls debug rsvp igp-sync lsp** command when an LSP name is specified.

```
Brocade# show mpls debug rsvp igp-sync lsp lsp04-28
LSP: lsp04-28, Path: path_1_to_4_strict
 CSPF: enabled, Record Route: enabled, Frr: disabled
 CSPF hops: 3, Record route hops: 3
 RSVP IGP Sync Links:
  10.11.11.11->10.1.1.2, 10.22.22.22->10.1.1.3
  10.33.33.33->10.1.1.4,
 RRO hops: 3 hops
  10.1.1.2,  10.1.1.3,  10.1.1.4,
 CSPF hops: 3 hops
  10.1.1.2,  10.1.1.3,  10.1.1.4,
LSP: lsp04-28, Path: sec_path_1_to_4_2
```

```
 CSPF: enabled, Record Route: enabled, Frr: disabled
 CSPF hops: 2, Record route hops: 2
 RSVP IGP Sync Links:
  10.11.11.11->10.1.1.3,  10.33.33.33->10.1.1.4
 RRO hops: 2 hops
  10.1.1.3,  10.1.1.4,
 CSPF hops: 2 hops
  10.1.1.3,  10.1.1.4,
```

The following is a sample output from the **show mpls debug rsvp igp-sync lsp detail** command.

```
Brocade# show mpls debug rsvp igp-sync lsp detail
LSP: dbyp-10.1.1.1-1, Path:
 CSPF: enabled, Record Route: enabled, Frr: disabled
 CSPF hops: 1, Record route hops: 1
 RSVP IGP Sync Links:
  10.11.11.11->10.1.1.3,
 RRO hops: 1 hops
  10.1.1.3,
 CSPF hops: 1 hops
  10.1.1.3,
LSP: lsp04-1, Path: path_1_to_4
 CSPF: enabled, Record Route: enabled, Frr: enabled
 CSPF hops: 3, Record route hops: 4
 RSVP IGP Sync Links:
  10.11.11.11->10.1.1.1,  10.11.11.11->10.1.1.2
  20.22.22.22->20.1.1.3,  30.33.33.33->30.1.1.4
 RRO hops: 4 hops
  10.1.1.1,  10.1.1.2,  20.1.1.3,  30.1.1.4
 CSPF hops: 3 hops
  10.1.1.2,  20.1.1.3,  30.1.1.4,
LSP: lsp04-11, Path:
 CSPF: enabled, Record Route: enabled, Frr: disabled
 CSPF hops: 1, Record route hops: 1
 RSVP IGP Sync Links:
  10.11.11.11->10.1.1.4,
 RRO hops: 1 hops
  10.1.1.4,
 CSPF hops: 1 hops
  10.1.1.4,
```

show mpls debug rsvp igp-sync stats

Syntax: show mpls debug rsvp igp-sync stats

This command displays the RSVP IGP synchronization debug statistics. The following is a sample output from the **show mpls debug rsvp igp-sync stats** command.

```
Brocade# show mpls debug rsvp igp-sync stats
Rsvp IGP Sync debug statistics:
 Debug trace: Disabled
 Debug print: Disabled
  ospf_enable = 0
  ospf_disable = 0
  isis_enable = 1
  isis_disable = 0
  ospf_rmt_link_down = 0
  isis_rmt_link_down = 3
  total_rmt_link_down = 3
  ospf_lcl_link_down = 0
  isis_lcl_link_down = 3
```

```
  total_lcl_link_down = 0
  lsp_down_ind = 0
  alloc_fail = 0
  duplicate_entry = 0
  error = 0
```

show mpls debug rsvp igp-sync stats reset

Syntax: show mpls debug rsvp igp-sync stats reset

This command resets the RSVP IGP synchronization debug statistics. The following is a sample output from the **show mpls debug rsvp igp-sync stats reset** command.

```
Brocade# show mpls debug rsvp igp-sync stats reset
Rsvp IGP Sync summary
 RSVP-OSPF Sync: Enabled
 RSVP-ISIS Sync: Disabled
 Rsvp IGP Sync debug statistics:
 Debug trace: Disabled
 Debug print: Disabled
  ospf_enable = 0
  ospf_disable = 0
  isis_enable = 0
  isis_disable = 0
  igp_notify = 14
  igp_notify_noactn = 0
  invalid_link_addr = 0
  ospf_rmt_link_down = 8
  isis_rmt_link_down = 0
  total_rmt_link_down = 8
  ospf_lcl_link_down = 6
  isis_lcl_link_down = 0
  total_lcl_link_down = 6
  lsp_down_ind = 0
  isync_lsp_down = 0
  lsp_up_ind = 0
  link_db_built = 0
  rro_updt_rbld = 0
  igp_itc_send = 8
  igp_itc_rcvd = 8
  itc_send_failed = 0
  rtr_id_failed = 0
  no_rrt_tnnl_cb = 0
  link_db_failed = 0
  alloc_fail = 0
  duplicate_entry = 0
  error = 0

Resetting RSVP IGP Sync debug stats
```

MPLS debug commands

This section describes the debug commands used for monitoring the MPLS environment.

debug mpls?

Syntax: debug mpls?

This command displays all the module types of MPLS.

```
Brocade# debug mpls ?
all               turns on/off all MPLS debugs
api               MPLS API debug
cspf              MPLS cspf debug
dynamic-bypass    MPLS dynamic bypass debug
error             MPLS error msgs
forwarding        MPLS forwarding debug
ldp               MPLS ldp debug
lmgr              MPLS label manager debug
oam               MPLS OAM debug
routing           MPLS routing (rsir) debug
rsir              MPLS RSIR debug
rsvp              MPLS rsvp debug
```

debug mpls

Syntax: [no] debug mpls

This command is used to turn on the debugging output, while preserving the debugging configuration.

```
Brocade# debug mpls
Brocade# show debug
MPLS Debug Settings
   MPLS Debug is ON
   Forwarding
      All
```

debug mpls all

Syntax: [no] debug mpls all

This command is used to turn on debugging for all MPLS configurations.

debug mpls error

Syntax: [no] debug mpls error

This command is used to turn on error debugging for all the modules.

```
Brocade# debug mpls error
Brocade# show debug
MPLS Debug Settings
   MPLS Debug is OFF
   Error
      All
```

debug mpls routing error

Syntax: [**no**] **debug mpls routing error**

This command displays debugging information related to Interior Gateway Protocol (IGP) neighbor down events. Command output resembles the following example.

```
Brocade# debug mpls routing error
Sep 15 00:23:09 MPLS: ISIS level-2 neighbor 10.32.174.54 on interface eth 4/7 is
dead
```

MPLS clear command

Use the following clear command to clear MPLS counter information.

clear mpls debug counters

Syntax: **clear mpls debug counters** [**aall** | **ai3** | **amb** | **ambl** | **amh** | **ase** | **asel** | **bfd** | **i3** | **ipl** | **ipr** | **nbase-root** | **ntl** | **rcp** | **rcs** | **rldf** | **rlm** | **rri** | **rric** | **rrt** | **rsip** | **rspx** | **rstc** | **sck** | **yads**]

This command clears the debug counters.

NOTE
The **show mpls debug counter** and **clear mpls debug counters** commands are hidden CLI commands.

MPLS API

The Multiprotocol Label Switching (MPLS) Application Programming Interface (API) is the MPLS interface to applications, such as L2VPN (for VLL and VPLS), BGP, IPSTATIC, IS-IS, OSPF, and PBR. The applications must register for the events with the MPLS when they need to use the MPLS tunnels. The MPLS API generates events when the tunnels and pseudowires are established and released, and sends the events only to the applications that have been registered.

MPLS API show commands

This section describes the show commands that display MPLS API information.

show mpls debug api aggrhist

Syntax: **show mpls debug api aggrhist** *application_type*

This command displays detailed information about the aggregation history for a specific application.

show mpls debug api apihist

Syntax: **show mpls debug api apihist** *application_type*

This command displays information about the history of each event queued for the specified application. Command output resembles the following example.

```
Brocade# show mpls debug api apihist bgp
```

```
MPLS MPLS_APP_BGP API APPDQ history lifo
time            type        state     peer-addr       index      tnnl-vif vc_id
Aug 18 10:07:27 tnnl        up        10.2.2.1         1          0        --
Aug 18 10:07:22 tnnl        up        10.2.2.1         2          1        --
Aug 18 10:07:21 tnnl        up        10.2.2.1         3          2        --
Aug 18 09:51:59 tnnl        down      10.2.2.1         3          2        --
Aug 18 09:51:59 tnnl        down      10.2.2.1         2          1        --
Aug 18 09:51:59 tnnl        down      10.2.2.1         1          0        --
Aug 18 08:39:57 tnnl        up        10.2.2.1         1          0        --
Aug 18 08:39:56 tnnl        up        10.5.5.1         8          8        --
Aug 18 08:39:54 tnnl        up        10.5.5.1         6          6        --
Aug 18 08:39:53 tnnl        up        10.5.5.1         7          7        --
Aug 18 08:39:52 tnnl        up        !0.2.2.1         2          1        --
Aug 18 08:39:51 tnnl        up        10.2.2.1         3          2        --
```

show mpls debug api queue

Syntax: **show mpls debug api queue** *application_type*

This command displays an MPLS API queue summary for the specified application, as shown in the following example.

```
Brocade# show mpls debug api queue pbr
MPLS_APP_PBR queue
Num events processed 1
```

show mpls debug api registration

Syntax: **show mpls debug api registration** *application_type*

This command displays MPLS API registration-related information for the specified application. Command output resembles the following example.

```
Brocade# show mpls debug api registration ipstatic
DEST ADDR API reg detail
DEST ADDR 10.1.1.1
  snapshot_sent Y
  numsnapshots 1
 longest netmask 0xffffffff
```

show mpls debug l2vpn tunnels

Syntax: **show mpls debug l2vpn tunnels** *peer ip address*

This command displays information about the Layer 2 Virtual Private Network (L2VPN) tunnels.

MPLS API debug commands

This section describes the debug commands that generate debugging information about MPLS API.

debug mpls api

Syntax: [**no**] **debug mpls api**

This command displays debugging information about MPLS API, as shown in the following example.

```
Brocade# debug mpls api
Brocade# router isis
Aug 18 21:32:18 mpls_app_register_tnnl_name app_type MPLS_APP_ISIS is registered
```

```
Aug 18 21:32:18 sending event queued itc to app_type MPLS_APP_ISIS dest_app_id 15

Brocade# no router isis
Aug 18 21:32:10 mpls_app_deregister_igp_shortcut app_type MPLS_APP_ISIS is
de-registered

Brocade# ip route next-hop-enable-mpls
Jan 11 04:53:38 mpls_api dest addr registering MPLS_APP_IP_STATIC value 10.1.1.1
Jan 11 04:53:38 mpls_app_lookup_dest_ipaddr MPLS_APP_IP_STATIC app_type  init
remote addr 10.1.1.1
Jan 11 04:53:38 mpls_app_lookup_dest_ipaddr MPLS_APP_IP_STATIC app_type
registered for remote addr 10.1.1.1
Jan 11 04:53:38 send dest addr snapshot
Jan 11 04:53:38 sending event queued itc to app_type MPLS_APP_IP_STATIC
dest_app_id 12
Jan 11 04:53:38 mpls_app_tunnel_deregister
Jan 11 04:53:38 mpls_app_deregister_dest_addr app_type MPLS_APP_IP_STATIC
dest_addr 10.20.160.1
Jan 11 04:53:40 mpls_app_tunnel_deregister
Jan 11 04:53:40 mpls_app_deregister_dest_addr app_type MPLS_APP_IP_STATIC
dest_addr 10.20.160.1

Brocade# no ip route next-hop-enable-mpls
Jan 11 04:53:31 mpls_app_deregister_all app_type MPLS_APP_IP_STATIC is
de-registered

Brocade# set next-hop-lsp to-mlxE-1
Jan 11 04:55:59 mpls_app_register_tnnl_name app_type MPLS_APP_PBR tnnl name
to-mlxE-1 registered
Jan 11 04:55:59 mpls_app_send_tnnl_name_snapshot app_type MPLS_APP_PBR tnnl name
to-mlxE-1 sending snapshot
Jan 11 04:55:59 mpls_app_register_tnnl_name app_type MPLS_APP_PBR tnnl name
to-mlxE-1 sending snapshot
Jan 11 04:55:59 sending event queued itc to app_type MPLS_APP_PBR dest_app_id 27

Brocade# no set next-hop-lsp to-mlxE-1
Jan 11 04:55:52 mpls_app_deregister_name app_type MPLS_APP_PBR tnnl_name
to-mlxE-1 is deregistered

Brocade# next-hop-mpls
Jan 11 05:05:42 mpls_api dest addr registering MPLS_APP_BGP value 10.1.1.1
Jan 11 05:05:42 mpls_app_lookup_dest_ipaddr MPLS_APP_BGP app_type  init remote
addr 10.1.1.1
Jan 11 05:05:42 mpls_app_lookup_dest_ipaddr MPLS_APP_BGP app_type registered for
remote addr 10.1.1.1
Jan 11 05:05:42 send dest addr snapshot
Jan 11 05:05:42 sending event queued itc to app_type MPLS_APP_BGP dest_app_id 13
Jan 11 05:05:42 mpls_api dest addr registering MPLS_APP_BGP value 10.1.1.2
Jan 11 05:05:42 mpls_app_lookup_dest_ipaddr MPLS_APP_BGP app_type  init remote
addr 10.1.1.2
Jan 11 05:05:42 mpls_app_lookup_dest_ipaddr MPLS_APP_BGP app_type registered for
remote addr 10.1.1.2
Jan 11 05:05:42 send dest addr snapshot
Jan 11 05:05:43 mpls_api dest addr registering MPLS_APP_BGP value 10.10.1.10
Jan 11 05:05:43 mpls_app_lookup_dest_ipaddr MPLS_APP_BGP app_type  init remote
addr 10.10.1.10
Jan 11 05:05:43 mpls_app_lookup_dest_ipaddr MPLS_APP_BGP app_type registered for
remote addr 10.10.1.10
Jan 11 05:05:43 send dest addr snapshot
Jan 11 05:05:43 send dest addr snapshot no route
```

```
Jan 11 05:05:43 mpls_api dest addr registering MPLS_APP_BGP value 10.5.5.1
Jan 11 05:05:43 mpls_app_lookup_dest_ipaddr MPLS_APP_BGP app_type  init remote
addr 10.5.5.1
Jan 11 05:05:43 mpls_app_lookup_dest_ipaddr MPLS_APP_BGP app_type registered for
remote addr 10.5.5.1
Jan 11 05:05:43 send dest addr snapshot
Jan 11 05:05:43 sending event queued itc to app_type MPLS_APP_BGP dest_app_id 13
Brocade# no next-hop-mpls
Jan 11 05:06:22 mpls_app_tunnel_deregister
Jan 11 05:06:22 mpls_app_deregister_dest_addr deregistered app_type MPLS_APP_BGP
dest_addr 10.1.1.1
Jan 11 05:06:22 mpls_app_tunnel_deregister
Jan 11 05:06:22 mpls_app_deregister_dest_addr deregistered app_type MPLS_APP_BGP
dest_addr 10.1.1.2
Jan 11 05:06:22 mpls_app_tunnel_deregister
Jan 11 05:06:22 mpls_app_deregister_dest_addr deregistered app_type MPLS_APP_BGP
dest_addr 10.5.5.1
Jan 11 05:06:22 mpls_app_tunnel_deregister
Jan 11 05:06:22 mpls_app_deregister_dest_addr deregistered app_type MPLS_APP_BGP
dest_addr 10.10.1.10
```

debug mpls l2vpn events

Syntax: [no] debug mpls l2vpn events

This command displays debugging information about the events processing in MPLS Layer 2 VPN.

MPLS CSPF debug commands

Constrained Shortest Path First (CSPF) computes the shortest path that fulfills a set of constraints. This means that CSPF runs a shortest path algorithm after pruning any links that violate a given set of constraints, such as minimum bandwidth required per link (also known as bandwidth guaranteed constraint).

debug mpls cspf

Syntax: [no] debug mpls cspf [computation |mapping | ted | error | all]

This debug command displays information about CSPF computations, mapping, TE databases, and errors.

- **computation** - Displays CSPF computation information.
- **mapping** - Displays information about address mappings in the CSPF module.
- **ted** - Displays information about the TE database.
- **error** - Displays CSPF error messages.
- **all** - Displays all debug error messages related to the CSPF module.

debug mpls cspf computation

Syntax: [no] debug mpls cspf computation [all | detail | lsp]

- **all** - Displays all debug messages related to CSPF computation.
- **detail** - Displays more detailed information about CSPF computation.
- **lsp** - Displays CSPF computation messages for specific LSPs.

This command displays CSPF computation information filtered by more specific criteria. If an LSP does not come up with error code "No path found", it means that CSPF could not calculate a path for the destination with the specified set of constraints. Enable this debug tracing to know why the path computation failed. Route query-related events for LSP, including Detour path and Facility path, are traced by enabling this debugging module.

Command output resembles the following example.

```
Brocade# debug mpls cspf computation
MPLS: CSPF: Unable to find router ID corresponding to destination IP
10.100.100.100 in area 0. LSP not created
MPLS: CSPF: Strict Hop - Locate Link from SrcRtr 10.20.20.20 to DstRtr
10.100.100.100, DstIntfAddr 10.1.1.1
MPLS: CSPF: Strict Hop processing matching link to 10.100.100.100
[O[0]:10.1.2.2:10.1.2.1:10.20.20.20:10.100.100.100]
MPLS: CSPF: Link Constraints - Satisfied:
[O[0]:10.1.2.2:10.1.2.1:10.20.20.20:10.100.100.100]
CSPF: Strict Hop processing matching link to 10.100.100.100
[O[0]:10.1.1.2:10.1.1.1:10.20.20.20:10.100.100.100]
MPLS: CSPF: Link Constraints - Satisfied: [O[0]:10.1.1.2:10.1.1.1:
10.20.20.20:10.100.100.100]
MPLS: CSPF: Strict Hop - Found Link from SrcRtr 10.20.20.20 to DstRtr
10.100.100.100, DstIntfAddr 10.1.1.1
[O[0]:10.1.1.2:10.1.1.1:10.20.20.20:10.100.100.100]
MPLS: CSPF: Final CSPF route in area 0
           Hop 1: 10.1.1.1, Rtr 10.100.100.100
RSIR: route query for 10.1.1.1/32
RSIR: Route Query success, NH 10.1.1.0 EgressIf e1/1 Ingr 0 Egr 0
RSIR: Route Query success, NH 10.1.1.0 EgressIf e1/1 Ingr 0 Egr 0
MPLS: CSPF: Strict Hop - Locate Link from SrcRtr 10.20.20.20 to DstRtr
10.100.100.100, DstIntfAddr 10.1.2.1
MPLS: CSPF: Strict Hop processing matching link to 10.100.100.1
00 [O[0]:10.1.2.2:10.1.2.1:10.20.20.20:10.100.100.100]
MPLS: CSPF: Link Constraints - Satisfied: [O[0]:10.1.2.2:10.1.2.1:
10.20.20.20:10.100.100.100]
MPLS: CSPF: Strict Hop - Found Link from SrcRtr 10.20.20.20 to DstRtr
10.100.100.100, DstIntfAddr 10.1.2.1 [O[0]:10.1.2.2:10.1.2.1:10.20.20.20:10
.100.100.100]
MPLS: CSPF: Final CSPF route in area 0
           Hop 1: 10.1.2.1, Rtr 10.100.100.100
RSIR: route query for 10.1.2.1/32
RSIR: Route Query success, NH 10.1.2.0 EgressIf e1/2 Ingr 0 Egr 0
RSIR: Route Query success, NH 10.1.2.0 EgressIf e1/2 Ingr 0 Egr 0
MPLS: CSPF: Processing TE Link: From Node: [O:0:10.100.100.100] Link
[O[0]:10.1.3.1:10.1.3.2:10.100.100.100:10.20.20.20]
MPLS: CSPF: match cspf-group group1, penalty 65000, mode: add-penalty
RSIR: CSPF Begin FRR BACKUP_CSPF route calculation
MPLS: CSPF: match cspf-group group1, penalty 65000, mode: add-penalty
MPLS: CSPF: bypass lsp[bypass1] usable with cost 65001 and 0 LSP riding on it
MPLS: CSPF: bypass lsp[bypass1] is selected with cost 65001 and 1 LSP riding on
it
```

debug mpls cspf computation lsp

Syntax: [**no**] **debug mpls cspf computation lsp** [**name** *name* | **sess_obj** *source_ip_address destination_ip_address tunnel_id*]

This command displays CSPF computation information for specific LSPs.

- **name** *name* - Limits the display of information to debug messages for the LSP that matches with the specified LSP name.
- **sess_obj** *source_ip_address destination_ip_address tunnel_id* - Limits the display of information to debug messages for the LSP that matches with the specified session object, which includes source IP address, destination IP address, and tunnel ID.

Command output resembles the following example.

```
Brocade# debug mpls cspf computation lsp sess_obj 10.7.7.1 10.7.7.2 100
Dec 10 00:51:04 MPLS: CSPF: Begin Constrained Dijkstra from 10.7.7.2 to 10.7.7.1,
dstIntf 10.7.7.1
Dec 10 00:51:04 MPLS: CSPF: Node Constraints - Satisfied [I:2:PE2.00]
Dec 10 00:51:04 MPLS: CSPF:     Traversing TE Links node_from 10.7.7.2,
node_from->dist 0X00000000, node_from 0X36C6B072
Dec 10 00:51:04 MPLS: CSPF:            Processing TE Link: From Node: [I:2:PE2.00]
                        Link [I[2]:10.1.1.2:10.1.1.1:PE2.00:PE4.00]
Dec 10 00:51:04 MPLS: CSPF:             node_to [I:2:PE2.00], node_to->dist
0X80000000, te_link->te_metric 0X0000000A, node_to 0X36C6B63C
Dec 10 00:51:04 MPLS: CSPF: Link Constraints - Satisfied:
[I[2]:10.1.1.2:10.1.1.1:PE2.00:PE4.00]
Dec 10 00:51:04 MPLS: CSPF:            Processing TE Link: From Node: [I:2:PE2.00]
                        Link [I[2]:10.1.2.3:10.1.2.4:PE2.00:PE3.00]
Dec 10 00:51:04 MPLS: CSPF:             node_to [I:2:PE2.00], node_to->dist
0X80000000, te_link->te_metric 0X0000000A, node_to 0X36C6B5CA
Dec 10 00:51:04 MPLS: CSPF: Link Constraints - Satisfied:
[I[2]:10.1.2.3:10.1.2.4:PE2.00:PE3.00]
Dec 10 00:51:04 MPLS: CSPF:            Processing TE Link: From Node: [I:2:PE2.00]
                        Link [I[2]:10.1.17.2:10.1.17.1:PE2.00:PE1.00]
Dec 10 00:51:04 MPLS: CSPF:             node_to [I:2:PE2.00], node_to->dist
0X80000000, te_link->te_metric 0X0000000A, node_to 0X36C6B720
Dec 10 00:51:04 MPLS: CSPF: Link Constraints - Satisfied:
[I[2]:10.1.17.2:10.1.17.1:PE2.00:PE1.00]
Dec 10 00:51:04 MPLS: CSPF: Node Constraints - Satisfied [I:2:PE1.00]
Dec 10 00:51:04 MPLS: CSPF:     Traversing TE Links node_from 10.7.7.1,
node_from->dist 0X0000000A, node_from 0X36C6B720
Dec 10 00:51:04 MPLS: CSPF:            Processing TE Link: From Node: [I:2:PE1.00]
                        Link [I[2]:10.3.11.2:10.3.11.1:PE1.00:PE4.00]
Dec 10 00:51:04 MPLS: CSPF:             node_to [I:2:PE1.00], node_to->dist
0X0000000A, te_link->te_metric 0X00000064, node_to 0X36C6B63C
Dec 10 00:51:04 MPLS: CSPF:            Processing TE Link: From Node: [I:2:PE1.00]
                        Link [I[2]:10.8.3.2:10.8.3.1:PE1.00:PE3.00]
Dec 10 00:51:04 MPLS: CSPF:             node_to [I:2:PE1.00], node_to->dist
0X0000000A, te_link->te_metric 0X0000000A, node_to 0X36C6B5CA
Dec 10 00:51:04 MPLS: CSPF:            Processing TE Link: From Node: [I:2:PE1.00]
                        Link [I[2]:10.6.7.2:10.6.7.1:PE1.00:PE3.00]
Dec 10 00:51:04 MPLS: CSPF:             node_to [I:2:PE1.00], node_to->dist
0X0000000A, te_link->te_metric 0X00000064, node_to 0X36C6B5CA
Dec 10 00:51:04 MPLS: CSPF:            Processing TE Link: From Node: [I:2:PE1.00]
                        Link [I[2]:10.2.3.4:10.2.3.3:PE1.00:PE4.00]
Dec 10 00:51:04 MPLS: CSPF:             node_to [I:2:PE1.00], node_to->dist
0X0000000A, te_link->te_metric 0X00000064, node_to 0X36C6B63C
Dec 10 00:51:04 MPLS: CSPF:            Processing TE Link: From Node: [I:2:PE1.00]
                        Link [I[2]:10.1.17.1:10.1.17.2:PE1.00:PE2.00]
Dec 10 00:51:04 MPLS: CSPF:             node_to [I:2:PE1.00], node_to->dist
0X00000000, te_link->te_metric 0X00000064, node_to 0X36C6B072
Dec 10 00:51:04 MPLS: CSPF: begin equal cost paths, dst_node = 10.7.7.1
Dec 10 00:51:04 MPLS: CSPF:             Processing TE Link:
                        Link [I[2]:10.3.11.2:10.3.11.1:PE1.00:PE4.00]
Dec 10 00:51:04 MPLS: CSPF:                     Remote node 0X36C6B63C was not
scanned - Ignored
```

```
Dec 10 00:51:04 MPLS: CSPF:             Processing TE Link:
                        Link [I[2]:10.8.3.2:10.8.3.1:PE1.00:PE3.00]
Dec 10 00:51:04 MPLS: CSPF:                     Remote node 0X36C6B5CA was not
scanned - Ignored
Dec 10 00:51:04 MPLS: CSPF:             Processing TE Link:
                        Link [I[2]:10.6.7.2:10.6.7.1:PE1.00:PE3.00]
Dec 10 00:51:04 MPLS: CSPF:                     Remote node 0X36C6B5CA was not
scanned - Ignored
Dec 10 00:51:04 MPLS: CSPF:             Processing TE Link:
                        Link [I[2]:10.2.3.4:10.2.3.3:PE1.00:PE4.00]
Dec 10 00:51:04 MPLS: CSPF:                     Remote node 0X36C6B63C was not
scanned - Ignored
Dec 10 00:51:04 MPLS: CSPF:             Processing TE Link:
                        Link [I[2]:10.1.17.1:10.1.17.2:PE1.00:PE2.00]
Dec 10 00:51:04 MPLS: CSPF:                     Remote node 0X36C6B072 is a parent
- Ignored
Dec 10 00:51:04 MPLS: CSPF: Node Constraints - Satisfied [I:2:PE1.00]
Dec 10 00:51:04 MPLS: CSPF: Node Constraints - Satisfied [I:2:PE2.00]
Dec 10 00:51:04 CSPF: Equal (cost and hop) paths found 1:
Dec 10 00:51:04         Path 1:
Dec 10 00:51:04                 [ 1] 10.1.17.1
Dec 10 00:51:04 MPLS: CSPF: Tie-breaking selected path: 1
Dec 10 00:51:04 MPLS: CSPF: Final CSPF route in area 2
Dec 10 00:51:04         Hop 1: 10.1.17.1, Rtr 10.7.7.1
Dec 10 00:51:04 MPLS: CSPF: Unable to find router ID corresponding to hop 10.1.1.2
in area 2. LSP not created
```

MPLS forwarding debug commands

This section describes the debug commands that generate MPLS forwarding information.

debug mpls forwarding

Syntax: [**no**] **debug mpls forwarding** [**all** | **ldp** | **rsvp** | **resource** | **error**]

- **all** - Displays all debug messages related to MPLS forwarding.
- **ldp** - Displays LDP-related forwarding information.
- **rsvp** - Displays RSVP-related forwarding information.
- **resource** - Displays information about available MPLS resources.
- **error** - Displays MPLS forwarding-related error messages.

The MPLS control plane interacts with the data forwarding plane through the forwarding interface. This debugging command displays RSVP, LDP, and resource usage-related information.

Command output resembles the following example.

```
Brocade# debug mpls forwarding
RLDF: ADD out_cb(0X14319B54), out-s-idx=3, out-int=2, out-lbl=0
lsp_cb=0X1431A220
RLDF: ADD xc_cb(0X14319910), in_cb=0X1308137C, out_cb=0X14319B54,
lsp_cb=0X1431A220
RLDF : Check BW
   Path_TSpec valid: BW = 0 Kb/sec
Resv_TSpec valid: BW = 0 Kb/sec
Alloc BW[outseg idx: 3]: setup/hold priority 3/7:
Path_TSpec valid: BW = 0 Kb/sec
Resv_TSpec valid: BW = 0 Kb/sec
```

```
Allocated BW 0 kbps for priority 0
RLDF: Update XC: lsp_cb 0X00000000, in_cb 0X00000000, out_cb 0X14319B54, xc_cb
0X14319910
RLDF: Update XC: lsp_xc_id 3, in-lbl 0, in-if port_id 65535, out-seg_idx 3, out-if
e1/2
RLDF: update_tnnl_vif_nht_index: Old 65535, new 1
RLDF - tnl 6 goes up
RLDF - tnnl add - For an UP tunnel, the out_seg are the same
RLDF - tnnl add - For an UP tunnel, the out_seg are the same
RLDF - rx'ed DISASSOCIATE_FEC_XC for fec 10.100.100.100
RLDF - tnl 5 goes down
RLDF - tnl 5 deleted from mpls route table and indicated to application
RLDF - LDP Tnnl 5 for fec 10.100.100.100 deleted
RLDF: DEL xc_cb (0X1306BF0C) lsp_cb (0X1306C394)
RLDF: DEL out_cb(0X1306C150), out-s-idx=1, out-int=1, out-lbl=0
lsp_cb=0X1306C394
RLDF: DEL in_cb(0X13081D38), lsp_cb 0X1306C394
RLDF: free lsp_cb=0X1306C394 lsp_xc_index=0X00000001
RLDF: ADD lsp_cb(0X1306C5D8), lsp_xc_idx=0X00000004
RLDF: ADD out_cb(0X1306BF0C), out-s-idx=4, out-int=1, out-lbl=0
lsp_cb=0X1306C5D8
RLDF: Add in_cb(0X13081E9C),lblsp-idx=0, in-if=0,in-lbl=0 lsp_cb 0X1306C5D8
RLDF: ADD xc_cb(0X1306C81C), in_cb=0X13081E9C, out_cb=0X1306BF0C,
lsp_cb=0X1306C5D8
RLDF: Update XC: lsp_cb 0X00000000, in_cb 0X00000000, out_cb 0X1306BF0C, xc_cb
0X1306C81C
RLDF: Update XC: lsp_xc_id 4, in-lbl 0, in-if port_id 65535, out-seg_idx 4, out-if
e1/1
RLDF - rx'ed ASSOCIATE_FEC_XC for fec 10.100.100.100
RLDF - tnl 5 goes up
RLDF - tnl 5 added to mpls route table and indicated to application
RLDF - LDP tnnl 0X00000005 installed for fec 10.100.100.100, outgoing port e1/1
```

debug mpls forwarding ldp

Syntax: [**no**] **debug mpls forwarding ldp** [**fec-key** [**all** | **prefix** [**all** | *prefix_ip*] | **vc** [**all** | *vc_id*]]]

- **fec-key all** - Enables or disables all Label Distribution Protocol (LDP) forwarding debugging for all the Forwarding Equivalence Classes (FECs).
- **prefix all** - Enables or disables all LDP forwarding prefix FEC debugging.
- **prefix** *prefix_ip* - Enables or disables LDP forwarding prefix FEC debugging for the specified prefix IP address and subnet mask.
- **vc all** - Enables or disables all LDP forwarding VC FEC debugging.
- **vc** *vc_id* - Enables or disables LDP forwarding VC FEC debugging for the speciifed VC ID.

This command displays LDP-related forwarding debugging information based on the specified filter options. Command output resembles the following example.

```
Brocade# debug mpls forwarding ldp fec-key prefix 10.130.130.3/32
Brocade(config-vif-15)# enable
Feb 22 14:12:06.274 RLDF: implicit xc, update_flags: 0X00000015, lspxc_flags:
0X00000000, lsp_ref_type: 2, in_seg_ref_type: 0, out_seg_ref_type: 2,
lsp_xc_index: 803

Feb 22 14:12:06.274 RLDF: implicit xc, add transit out_seg, out_if: 1553

Feb 22 14:12:06.274 RLDF: Attempt to recover fec_type: 2, fec_addr: 10.130.130.3,
fec_prefix_len: 32
```

```
Feb 22 14:12:06.274 RLDF: rldf_check_recovery_cbs: not recover lsp_cb 0X00000000,
recovery state 0X000000FF
Feb 22 14:12:06.274 RLDF: In attempt to recover lsp_cb, DIDNOT Recover
Feb 22 14:12:06.274 RLDF: ADD lsp_cb(0X2D9B9F70), lsp_xc_idx=0X00000323,
lsp_sync_index=0X00000004, sync_info: 0X2F597810
Feb 22 14:12:06.274 RLDF: ADD out_cb(0X2D98A598), out-s-idx=1473, out-int=1553,
out-lbl=0 lsp_cb=0X2D9B9F70
Feb 22 14:12:06.274 RLDF: implicit xc, create ldp tunnel

Feb 22 14:12:06.275 RLDF: Updating sync node 0X2F597810 in sync library
Feb 22 14:12:06.275 RLDF: update_tnnl_vif_nht_index: Old 65535, new 1
Feb 22 14:12:06.275 RLDF: ADD tnnl_vif_index=7 allocate accounting_label=1048580
Feb 22 14:12:06.275 RLDF - tnl 7 goes up

Feb 22 14:12:06.275 RLDF - tnl 7 added to mpls route table and indicated to
application

Feb 22 14:12:06.275 RLDF: Updating sync node 0X2F597810 in sync library
Feb 22 14:12:06.318 RLDF: slib pack lsp cb with size 186, buffer available was
1838, action 3
Feb 22 14:12:06.318 RLDF: mpls_sync_pack_lsp: fec_type: 2, fec_addr:
10.130.130.3, fec_prefix_len: 32, ldp_flags: 0X00000005, ldp_in_seg_label: 0
```

debug mpls forwarding rsvp

Syntax: [**no**] **debug mpls forwarding rsvp** [**all** | **lsp session-obj** [**all** | *source_ip_address destination_ip_address tunnel_id*]]

- **all** - Enables or disables all RSVP forwarding debugging.
- **lsp session-obj all** - Enables or disables all RSVP session object debugging.
- **lsp session-obj** *source_ip_address destination_ip_address tunnel_id* - Enables or disables RSVP session object debugging for the specified RSVP session object.

The **debug mpls forwarding rsvp all** command output resembles the following example.

```
Brocade# debug mpls forwarding rsvp all
Num_hops downstream 1, upstream 1, exclude 1
Dec 10 00:59:58 MPLS: CSPF: RSIR: BYPASS_CSPF: Finding Bypass path to: 10.1.17.1
Dec 10 00:59:58 RSIR: BYPASS_CSPF: Constraints: BW: 0 kbps Setup Prio: 4
Hop-Limit: 255
 exc-any: 0x0X00000000 inc-any: 0x0X00000000 inc-all: 0x0X00000000
Dec 10 00:59:58 RSIR: BYPASS_CSPF: Removed link local IP: 10.1.17.2
Dec 10 00:59:58 RSIR: BYPASS_CSPF: Exclude_hops[0]: 0.0.0.0
Dec 10 00:59:58 RSIR: CSPF Begin FRR BYPASS_CSPF route calculation
Dec 10 00:59:58 RSIR: CSPF End FRR BYPASS_CSPF route calculation. Route not found,
error_code: 1
Dec 10 00:59:58 CSPF failed to calculate bypass route to 10.1.17.1
Dec 10 00:59:58 MPLS: CSPF: RSIR: BYPASS_CSPF: My IP address from upstream entry:
10.1.17.2
Num_hops downstream 1, upstream 1, exclude 1
Dec 10 00:59:58 MPLS: CSPF: RSIR: BYPASS_CSPF: Finding Bypass path to: 10.1.17.1
Dec 10 00:59:58 RSIR: BYPASS_CSPF: Constraints: BW: 0 kbps Setup Prio: 4
Hop-Limit: 255
 exc-any: 0x0X00000000 inc-any: 0x0X00000000 inc-all: 0x0X00000000
Dec 10 00:59:58 RSIR: BYPASS_CSPF: Removed link local IP: 10.1.17.2
Dec 10 00:59:58 RSIR: CSPF Begin FRR BYPASS_CSPF route calculation
Dec 10 00:59:58 RSIR: CSPF End FRR BYPASS_CSPF route calculation. Route not found,
error_code: 1
```

The **debug mpls forwarding rsvp lsp session-obj** command output resembles the following example.

```
Brocade# debug mpls forwarding rsvp lsp sess-obj 10.110.110.1 10.130.130.1 13
Brocade (config)# router mpls
Brocade(config-mpls)# lsp t-2
Brocade(config-mpls-lsp-t-2)# to 10.130.130.3
Brocade(config-mpls-lsp-t-2)# enable
Connecting signaled LSP t-2
Feb 22 13:49:56.767 RLDF: ADD lsp_cb(0X2D9B9000), lsp_xc_idx=0X0000031E,
lsp_sync_index=0X00000004, sync_info: 0X00000000
Feb 22 13:49:56.767 RLDF: Add in_cb(0X2D95D2D0),lblsp-idx=0, in-if=0,in-lbl=0
lsp_cb 0X2D9B9000
Feb 22 13:49:56.767 RLDF : Check BW
Feb 22 13:49:56.767    Path_TSpec valid: BW = 0 Kb/sec
DUT1(config-mpls)#Feb 22 13:49:56.771 RLDF: ADD out_cb(0X2D98A598),
out-s-idx=1468, out-int=1553, out-lbl=0 lsp_cb=0X2D9B9000
Feb 22 13:49:56.772 RLDF: ADD xc_cb(0X2D998618), in_cb=0X2D95D2D0,
out_cb=0X2D98A598, lsp_cb=0X2D9B9000
Feb 22 13:49:56.772 RLDF : Check BW
Feb 22 13:49:56.772    Path_TSpec valid: BW = 0 Kb/sec
Feb 22 13:49:56.772    Resv_TSpec valid: BW = 0 Kb/sec
Feb 22 13:49:56.772 Alloc BW[outseg idx: 1468]: setup/hold priority 7/0:
Feb 22 13:49:56.772    Path_TSpec valid: BW = 0 Kb/sec
Feb 22 13:49:56.772    Resv_TSpec valid: BW = 0 Kb/sec
```

debug mpls forwarding resource

Syntax: [**no**] **debug mpls forwarding resource**

This command displays information about available MPLS resources. Command output resembles the following example.

```
Brocade# debug mpls forwarding resource
Check BW:
    Path_TSpec valid: BW = 50 Kb/sec
    Resv_TSpec valid: BW = 50 Kb/sec
 Alloc BW[outseg idx: 7]: setup/hold priority 7/0:
    Path_TSpec valid: BW = 50 Kb/sec
    Resv_TSpec valid: BW = 50 Kb/sec
 Allocated BW 50 kbps for priority 0
```

MPLS routing debug commands

This section describes the debug commands that generate MPLS routing information.

debug mpls routing

Syntax: [**no**] **debug mpls routing** [**all** | **error** | **interface** | **prefix**]

This command displays MPLS routing-related information.

- **all** - Displays all debug messages related to MPLS routing.
- **error** - Displays MPLS routing-related error messages.
- **interface** - Limits the messages to specific interfaces. This filter captures the event of a particular IP interface indication (or polling) by routing stub module to MPLS.
- **prefix** - Limits the messages to specific prefixes. The purpose of this filter is to trace a particular IP route notification by routing module to MPLS.

Command output resembles the following example.

```
Brocade# debug mpls routing
RSIR skip route add: 10.1.1.0/24, ingr(0), egr(0)
RSIR port state change indication for e1/1: Admin: UP Oper: UP
RSIR IP address indication to RRI for e1/1: addr add 10.1.1.2/24
RSIR IP address indication to RCP for e1/1: addr add 10.1.1.2/24
RSIR IP address indication to RCS for e1/1: addr add 10.1.1.2/24
RSIR route add(RSVP) indication: 10.1.2.0/24, idx 0x0X0A3682E6, sh_cut
0x0X00000000 nh 10.1.2.0, intf e1/2, ingr 0, egr 0, r_flag 0x0X00000000
RSIR skip route add: 10.1.2.0/24, ingr(0), egr(0)
RSIR port state change indication for e1/2: Admin: UP Oper: UP
RSIR IP address indication to RRI for e1/2: addr add 10.1.2.2/24
RSIR IP address indication to RCP for e1/2: addr add 10.1.2.2/24
RSIR IP address indication to RCS for e1/2: addr add 10.1.2.2/24
RSIR route add(RSVP) indication: 10.1.3.0/24, idx 0x0X0A3682F6, sh_cut
0x0X00000000 nh 10.1.3.0, intf e1/3, ingr 0, egr 0, r_flag 0x0X00000000
RSIR skip route add: 10.1.3.0/24, ingr(0), egr(0)
RSIR port state change indication for e1/3: Admin: UP Oper: UP
RSIR IP address indication to RRI for e1/3: addr add 10.1.3.2/24
RSIR IP address indication to RCP for e1/3: addr add 10.1.3.2/24
RSIR IP address indication to RCS for e1/3: addr add 10.1.3.2/24
RSIR port state change indication for e1/4: Admin: UP Oper: UP
```

debug mpls routing interface

Syntax: [**no**] **debug mpls routing interface** [**all** | **ethernet** *slot/port* | **pos** *slot/port* | **ve** *index*]

This command displays MPLS routing-related information to the specific interfaces.

- **all** - Displays all debug messages related to MPLS routing for all the interfaces.
- **ethernet** *slot/port* - Limits the messages to specific Ethernet interfaces.
- **pos** *slot/port* - Limits the messages to specific POS interfaces.
- **ve** *index* - Limits the messages to specific virtual Ethernet interfaces.

Command output resembles the following example.

```
Brocade# debug mpls routing interface all
Dec 10 01:02:29 RSIR port state change indication for e1/14: Admin: UP Oper: UP
Dec 10 01:02:29 RSIR IP address indication to RRI for e1/14: addr add 10.1.2.3/24
Dec 10 01:02:29 RSIR port state change indication for e1/14: Admin: UP Oper: UP
Dec 10 01:02:29 RSIR IP address indication to RCP for e1/14: addr add 10.1.2.3/24
Dec 10 01:02:29 RSIR IP address indication to RCS for e1/14: addr add 10.1.2.3/24
Dec 10 01:02:29 RSIR port state change indication for e1/14: Admin: UP Oper: UP
Dec 10 01:02:29 RSIR IP address indication to RRI for e1/14: addr add 10.1.2.3/24
Dec 10 01:02:29 RSIR port state change indication for e1/14: Admin: UP Oper: UP
Dec 10 01:02:29 RSIR IP address indication to RCP for e1/14: addr add 10.1.2.3/24
Dec 10 01:02:29 RSIR IP address indication to RCS for e1/14: addr add 10.1.2.3/24
nDec 10 01:02:30 RSIR port state change indication for e1/14: Admin: UP Oper: UP
Dec 10 01:02:30 RSIR IP address indication to RRI for e1/14: addr add 10.1.2.3/24
Dec 10 01:02:30 RSIR port state change indication for e1/14: Admin: UP Oper: UP
Dec 10 01:02:30 RSIR IP address indication to RCP for e1/14: addr add 10.1.2.3/24
Dec 10 01:02:30 RSIR IP address indication to RCS for e1/14: addr add 10.1.2.3/24
```

debug mpls routing prefix

Syntax: [**no**] **debug mpls routing prefix** *ip-address prefix-length*

- *ip-address* - Specifies the IP address.
- *prefix-length* - Specifies the prefix length.

This command limits the display of MPLS routing-related information to specific prefixes.

MPLS RSVP debug commands

This section describes the debug commands that generate MPLS RSVP information.

debug mpls rsvp

Syntax: [no] debug mpls rsvp [all | error |event | packets | session | tunnel]

- all - Displays all messages related to the MPLS RSVP module.
- error - Displays debug messages related to MPLS RSVP errors.
- event - Displays debug messages related to MPLS RSVP events. This includes those events that are not session-specific, for example, interface up or down, IP route indication, and so on.
- packets - Displays debug messages related to MPLS RSVP packets.
- session - Displays debug messages related to a specific MPLS RSVP session.
- tunnel - Displays debug messages related to MPLS RSVP LSP interaction with other modules as virtual interfaces.

debug mpls rsvp event

Syntax: [no] debug mpls rsvp event

This command displays information about MPLS RSVP events, as shown in the following example.

```
Brocade# debug mpls rsvp event
RSVP: kill_tc DEL_IN_SEG Sess 0x0da02020 10.33.33.1(2)<-10.11.11.
RSVP: kill_session 0x0da02020 10.33.33.1(2)<-10.11.11.11
RSVP: make_session 0x0da022f0 10.4.1.2(1)<-10.11.11.11
RSVP: make_PSB 0x0da10fc8 Sess 0x0da022f0 10.4.1.2(1)<-10.11.11.1
RSVP: make_PSB 0x0da107f8 Sess 0x0da022f0 10.4.1.2(1)<-10.11.11.1
RSVP: new_tc QUERY_ROUTE Sess 0x0da022f0 10.4.1.2(1)<-10.11.11.11
RSVP: kill_tc QUERY_ROUTE Sess 0x0da022f0 10.4.1.2(1)<-10.11.11.1
rrr_query_route_rsp: inserting PSB 0x0da10fc8 in unrouted list
RSVP_FRR: rrr_frr_merge_point: Merging PSB not found.
RSVP_FRR: Path Tear on non-merging protected LSP, PSB 0da10fc8.
RSVP_FRR: Tear down PLR detour 0da107f8
RSVP: kill_PSB 0x0da10fc8 Sess 0x0da022f0 10.4.1.2(1)<-10.11.11.1
RSVP_FRR: rrr_frr_merge_point: Merging PSB not found.
RSVP: new_tc DEL_IN_SEG Sess 0x0da022f0 10.4.1.2(1)<-10.11.11.11
RSVP: kill_tc DEL_IN_SEG Sess 0x0da022f0 10.4.1.2(1)<-10.11.11.11
RSVP: kill_session 0x0da022f0 10.4.1.2(1)<-10.11.11.11
RSVP: new_tc TIMER_POP Sess 0x0da03ad8 10.1.1.2(4)<-10.11.11.11
RSVP: kill_tc TIMER_POP Sess 0x0da03ad8 10.1.1.2(4)<-10.11.11.11
RSVP: new_tc TIMER_POP Sess 0x0da02e30 10.3.3.2(5)<-10.11.11.11
RSVP: kill_tc TIMER_POP Sess 0x0da02e30 10.3.3.2(5)<-10.11.11.11
RSVP: new_tc TIMER_POP Sess 0x0da03ad8 10.1.1.2(4)<-10.11.11.11
RSVP: kill_tc TIMER_POP Sess 0x0da03ad8 10.1.1.2(4)<-10.11.11.11
RSVP: new_tc TIMER_POP Sess 0x0da041e0 10.22.22.1(3)<-10.11.11.11
RSVP: kill_tc TIMER_POP Sess 0x0da041e0 10.22.22.1(3)<-10.11.11.1
RCPF RE: fec type 2(1/0) dest 10.4.1.0 nexthop 10.1.1.2: prefix len 24
dest_inet_pl 4, rt_idx 0, event 2, fec_cb 0x0d902138
RCPF RE: fec cb: ing/egress =  (1/0), rt_index 0, lib_ret 0, state 1 pend_not 0
RCPF RE: fec type 2(1/0) dest 10.44.44.0 nexthop 10.1.1.2: prefix len 24
dest_inet_pl 4, rt_idx 0, event 2, fec_cb 0x0d90f5b8
```

debug mpls rsvp packets

Syntax: [**no**] **debug mpls rsvp packets** [**all** | **detail** | **count** *number*| **direction** [**send** | **receive**] | **pkt_type** [**ack** | **all** | **bundle** | **path** | **patherr** | **resv** | **resverr** | **resvtear** | **summary-fresh**] | **interface** | **sess_obj** *source_ip_address* [*destination_ip_address* | *p2mp-Id*] *tunnel_id*]

This command displays MPLS RSVP packets-related information, which is further filtered by direction, packet type, interface, and session object.

- **all** - Displays all messages related to MPLS RSVP packets.
- **detail** - Displays detailed information about MPLS RSVP packets.
- **count** *number* - Limits the display of MPLS RSVP packets to the specified number.
- **direction** - Displays information about MPLS RSVP packets for the specified direction.
- **send** - Displays information about sent MPLS RSVP packets.
- **receive** - Displays information about received MPLS RSVP packets.
- **pkt_type** - Displays information about MPLS RSVP packet types (similar to the debug rsvp packets detail command).
- **ack** - Displays information about MPLS RSVP reservation request acknowledgement messages.
- **all** - Turns on or off debugging of all MPLS RSVP packet types.
- **bundle** - Displays information about MPLS RSVP bundle messages.
- **path** - Displays information about MPLS RSVP path messages.
- **patherr** - Displays information about MPLS RSVP path error messages.
- **pathtear** - Displays information about MPLS RSVP path tear messages.
- **resv** - Displays information about MPLS RSVP reservation request messages.
- **resverr** - Displays information about MPLS RSVP reservation request error messages.
- **resvtear** - Displays information about MPLS RSVP reservation tear messages.
- **summary-refresh** - Displays information about MPLS RSVP summary refresh messages.
- **interface** - Displays RSVP packets transmitted or received on an interface.
- **sess_obj** *source_ip_address* [*destination_ip_address* | *p2mp-Id*] *tunnel_id* - Displays information about the MPLS RSVP packets for the specified RSVP session object or Point to Multipoint (P2MP) session object. RSVP session object includes source IP address, destination IP address, and tunnel ID. P2MP session object incudes source IP address, P2MP ID, and tunnel ID.

The **debug mpls rsvp packets** command output resembles the following example.

```
Brocade# debug mpls rsvp packets
Send Path message: src 10.20.20.20, dst 10.100.100.100 on port 1/1
Dest 10.100.100.100, tunnelId 1, ext tunnelId 10.20.20.20
Receive Resv message: src 10.1.2.1, dst 10.1.2.2 on port 1/2
Dest 10.100.100.100, tunnelId 2, ext tunnelId 10.20.20.20
```

debug mpls rsvp packets detail

Syntax: [**no**] **debug mpls rsvp packets detail**

This command displays detailed information about MPLS RSVP packets. Command output resembles the following example.

```
Brocade# debug mpls rsvp packets detail
Path message
Sep  5 04:15:45  Send Path message: src 10.66.66.20,dst 10.66.66.40 on port ¼
Sep  5 04:15:45  Type 1(Path) ver 1 flags 0x00 cksum 0xe9b8 ttl 254 len 196
Sep  5 04:15:45  Obj_class 1 (SESSION) ctype 7 length 16:
Sep  5 04:15:45   Dest 10.40.40.40, tunnelid 1, ext tunnelid (source) 10.10.10.10
Sep  5 04:15:45  Obj_class 3 (RSVP_HOP) ctype 1 length 12:
Sep  5 04:15:45   Address: 10.66.66.20 LIH: 00000004
Sep  5 04:15:45  Obj_class 5 (TIME) ctype 1 length 8:
Sep  5 04:15:45   Value: 30000
Sep  5 04:15:45  Obj_class 20 (EXPLICIT_ROUTE) ctype 1 length 12:
Sep  5 04:15:45   (#1) strict ipv4 prefix, 8 bytes, 10.66.66.40/32
Sep  5 04:15:45  Obj_class 19 (LABEL_REQ) ctype 1 length 8:
Sep  5 04:15:45   Label_request: 00000800
Sep  5 04:15:45  Obj_class 207 (SESSION_ATTR) ctype 7 length 16:
Sep  5 04:15:45   Setup_pri: 0 hold_pri: 0 flags: 00000044
Sep  5 04:15:45   Session name: sundeep
Sep  5 04:15:45  Obj_class 11 (SENDER_TEMPLATE) ctype 7 length 12:
Sep  5 04:15:45   Source 10.10.10.10, lsp_id 1
Sep  5 04:15:45  Obj_class 12 (SENDER_TSPEC) ctype 2 length 36:
Sep  5 04:15:45   Max rate: 1000, mean rate: 1000, max burst: 0

Path Error
Sep  5 04:28:28 Send PathErr message: src 10.11.11.20, dst 10.11.11.10 on port 1/5
Sep  5 04:28:28  Type 3(PathErr) ver 1 flags 0x00 cksum 0xa5c3 ttl 63 len 84
Sep  5 04:28:28  Obj_class 1 (SESSION) ctype 7 length 16:
Sep  5 04:28:28   Dest 10.40.40.40, tunnelid 2, ext tunnelid (source) 10.10.10.10
Sep  5 04:28:28  Obj_class 6 (ERROR) ctype 1 length 12:
Sep  5 04:28:28   Node: 10.11.11.20 Flags: 0x00 code: 34 value 1
Sep  5 04:28:28  Obj_class 11 (SENDER_TEMPLATE) ctype 7 length 12:
Sep  5 04:28:28   Source 10.10.10.10, lsp_id 1
Sep  5 04:28:28  Obj_class 12 (SENDER_TSPEC) ctype 2 length 36:
Sep  5 04:28:28   Max rate: 1000, mean rate: 1000, max burst: 0
```

debug mpls rsvp packets interface

Syntax: [**no**] **debug mpls rsvp packets interface** [**all** | **ethernet** *slot/port* | **pos** *slot/port* | **ve** *index*]

This command limits the display of MPLS RSVP packets to the specific interfaces.

- **all** - Displays all MPLS RSVP packets on all interfaces.
- **ethernet** *slot/port* - Limits the display of packets to specific Ethernet interfaces.
- **pos** *slot/port* - Limits the display of packets to specific POS interfaces.
- **ve** *index* - Limits the display of packets to specific virtual Ethernet interfaces.

Command output resembles the following example.

```
Brocade# debug mpls rsvp packets interface all
Dec 10 01:03:26 Send Path message: src 10.7.7.2, dst 10.7.7.1 on port 3/1
Dec 10 01:03:26  Dest 10.7.7.1, tunnelId 2974, ext tunnelId 10.7.7.2, lspId 1
Dec 10 01:03:26 Send Path message: src 10.7.7.2, dst 10.7.7.1 on port 2/1
Dec 10 01:03:26  Dest 10.7.7.1, tunnelId 3634, ext tunnelId 10.7.7.2, lspId 1
Dec 10 01:03:26 Send Path message: src 10.7.7.2, dst 10.7.7.1 on port 3/1
Dec 10 01:03:26  Dest 10.7.7.1, tunnelId 2791, ext tunnelId 10.7.7.2, lspId 1
Dec 10 01:03:26 Receive Resv message: src 10.1.17.1, dst 10.1.17.2 on port 3/1
Dec 10 01:03:26  Dest 10.7.7.1, tunnelId 1058, ext tunnelId 10.7.7.2, lspId 1
Dec 10 01:03:26 Send Path message: src 10.7.7.2, dst 10.7.7.1 on port 3/1
Dec 10 01:03:26  Dest 10.7.7.1, tunnelId 625, ext tunnelId 10.7.7.2, lspId 1
Dec 10 01:03:26 Send Path message: src 10.7.7.2, dst 10.7.7.1 on port 3/1
```

```
Dec 10 01:03:26  Dest 10.7.7.1, tunnelId 775, ext tunnelId 10.7.7.2, lspId 1
Dec 10 01:03:26 Receive Resv message: src 10.1.17.1, dst 10.1.17.2 on port 3/1
Dec 10 01:03:26  Dest 10.7.7.1, tunnelId 825, ext tunnelId 10.7.7.2, lspId 1
Dec 10 01:03:26 Receive Resv message: src 10.1.17.1, dst 10.1.17.2 on port 3/1
Dec 10 01:03:26  Dest 10.7.7.1, tunnelId 1554, ext tunnelId 10.7.7.2, lspId 1
Dec 10 01:03:26 Receive Resv message: src 10.1.17.1, dst 10.1.17.2 on port 3/1
Dec 10 01:03:26  Dest 10.7.7.1, tunnelId 939, ext tunnelId 10.7.7.2, lspId 1
Dec 10 01:03:26 Receive Resv message: src 10.1.17.1, dst 10.1.17.2 on port 3/1
Dec 10 01:03:26  Dest 10.7.7.1, tunnelId 2646, ext tunnelId 10.7.7.2, lspId 1
Dec 10 01:03:26 Receive Resv message: src 10.1.17.1, dst 10.1.17.2 on port 3/1
Dec 10 01:03:26  Dest 10.7.7.1, tunnelId 3183, ext tunnelId 10.7.7.2, lspId 1
Dec 10 01:03:26 Receive Resv message: src 10.1.17.1, dst 10.1.17.2 on port 3/1
Dec 10 01:03:26  Dest 10.7.7.1, tunnelId 3297, ext tunnelId 10.7.7.2, lspId 1
```

debug mpls rsvp packets sess_obj

Syntax: [**no**] **debug mpls rsvp packets sess_obj** *source_ip_address* [*destination_ip_address* | *p2mp-Id*] *tunnel_id*

- *source_ip_address* - Specifies the source IP address.
- *destination_ip_address* - Specifies the destination IP address.
- *p2mp-Id* - Specifies the P2MP ID in decimal or IP address format. This variable is applicable to P2MP LSPs only and can be used to filter the debug tracing based on P2MP session object.
- *tunnel_id* - Specifies the tunnel ID.

This command displays information about the MPLS RSVP packets for the specified RSVP session object or Point to Multipoint (P2MP) session object. RSVP session object includes source IP address, destination IP address, and tunnel ID. P2MP session object incudes source IP address, P2MP ID, and tunnel ID. A command output such as the following is displayed when a RSVP packet matching the session object filter is received.

```
Brocade# debug mpls rsvp packets sess_obj 10.0.0.1 10.1.1.1 1
Jun  1 10:30:36.008 Send Path message: src 10.0.0.1, dst 10.0.0.2 on port 4/12
Jun  1 10:30:36.008  Type 1(Path) ver 1 flags 0x00 cksum 0xb0c8 ttl 63 len 220
Jun  1 10:30:36.008  Obj_class 1 (SESSION) ctype 13 length 16:
Jun  1 10:30:36.008   p2mpId 10.1.1.1, tunnelid 1, ext tunnelid (source) 10.0.0.2
Jun  1 10:30:36.008  Obj_class 3 (RSVP_HOP) ctype 1 length 12:
Jun  1 10:30:36.008   Address: 10.0.0.1 LIH: 0000009c
Jun  1 10:30:36.008  Obj_class 5 (TIME) ctype 1 length 8:
Jun  1 10:30:36.008   Value: 30000
Jun  1 10:30:36.008  Obj_class 19 (LABEL_REQ) ctype 1 length 8:
Jun  1 10:30:36.008   Label_request: 00000800
Jun  1 10:30:36.008  Obj_class 207 (SESSION_ATTR) ctype 7 length 20:
Jun  1 10:30:36.008   Setup_pri: 7 hold_pri: 7 flags: 00000000
Jun  1 10:30:36.008   Session name: Sundeep-p2mp
Jun  1 10:30:36.008  Obj_class 11 (SENDER_TEMPLATE) ctype 12 length 20:
Jun  1 10:30:36.008   Source 20.0.0.2, lsp_id 1
Jun  1 10:30:36.008   subgrpOrigId 10.0.0.2, subgrpId 1
Jun  1 10:30:36.008  Obj_class 12 (SENDER_TSPEC) ctype 2 length 36:
Jun  1 10:30:36.008   Max rate: 0, mean rate: 800, max burst: 0
Jun  1 10:30:36.008     0x00 0x00 0x00 0x07
Jun  1 10:30:36.008     0x01 0x00 0x00 0x06
Jun  1 10:30:36.008     0x7f 0x00 0x00 0x05
Jun  1 10:30:36.008     0x47 0xc3 0x50 0x00
Jun  1 10:30:36.008     0x00 0x00 0x00 0x00
Jun  1 10:30:36.008     0x00 0x00 0x00 0x00
Jun  1 10:30:36.008     0x00 0x00 0x00 0x00
Jun  1 10:30:36.008     0x00 0x00 0x00 0x00
```

```
Jun  1 10:30:36.008  Obj_class 13 (ADSPEC) ctype 2 length 84
Jun  1 10:30:36.008     0x00 0x00 0x00 0x13
Jun  1 10:30:36.009     0x01 0x80 0x00 0x08
Jun  1 10:30:36.009     0x04 0x00 0x00 0x01
Jun  1 10:30:36.009     0x00 0x00 0x00 0x01
Jun  1 10:30:36.009     0x06 0x00 0x00 0x01
Jun  1 10:30:36.009     0x00 0x00 0x00 0x00
Jun  1 10:30:36.009     0x08 0x00 0x00 0x01
Jun  1 10:30:36.009     0xff 0xff 0xff 0xff
Jun  1 10:30:36.009     0x0a 0x00 0x00 0x01
Jun  1 10:30:36.009     0x00 0x00 0x05 0xdc
Jun  1 10:30:36.009     0x02 0x80 0x00 0x08
Jun  1 10:30:36.009     0x85 0x00 0x00 0x01
Jun  1 10:30:36.009     0x00 0x00 0x00 0x00
Jun  1 10:30:36.009     0x86 0x00 0x00 0x01
Jun  1 10:30:36.009     0x00 0x00 0x00 0x00
Jun  1 10:30:36.009     0x87 0x00 0x00 0x01
Jun  1 10:30:36.009     0x00 0x00 0x00 0x00
Jun  1 10:30:36.009     0x88 0x00 0x00 0x01
Jun  1 10:30:36.009     0x00 0x00 0x00 0x00
Jun  1 10:30:36.009     0x05 0x80 0x00 0x00
Jun  1 10:30:36.009  Obj_class 50 (S2L_SUB_LSP) ctype 1 length 8
Jun  1 10:30:36.009   S2L dest addr 10.0.0.2

Jun  1 10:30:36.713 Send Resv message: src 10.0.0.1, dst 10.0.0.2 on port 4/11
Jun  1 10:30:36.713  Type 2(Resv) ver 1 flags 0x00 cksum 0xcf8a ttl 63 len 144
Jun  1 10:30:36.713  Obj_class 1 (SESSION) ctype 13 length 16:
Jun  1 10:30:36.713   p2mpId 10.1.1.1, tunnelid 1, ext tunnelid (source) 10.0.0.2
Jun  1 10:30:36.714  Obj_class 3 (RSVP_HOP) ctype 1 length 12:
Jun  1 10:30:36.714   Address: 10.0.0.1 LIH: 00000000
Jun  1 10:30:36.714  Obj_class 5 (TIME) ctype 1 length 8:
Jun  1 10:30:36.714   Value: 30000
Jun  1 10:30:36.714  Obj_class 8 (STYLE) ctype 1 length 8:
Jun  1 10:30:36.714   Style: SE
Jun  1 10:30:36.714  Obj_class 9 (FLOWSPEC) ctype 2 length 36:
Jun  1 10:30:36.714   Max rate: 0, mean rate: 800, max burst: 0
Jun  1 10:30:36.714     0x00 0x00 0x00 0x07
Jun  1 10:30:36.714     0x05 0x00 0x00 0x06
Jun  1 10:30:36.714     0x7f 0x00 0x00 0x05
Jun  1 10:30:36.714     0x47 0xc3 0x50 0x00
Jun  1 10:30:36.714     0x00 0x00 0x00 0x00
Jun  1 10:30:36.714     0x00 0x00 0x00 0x00
Jun  1 10:30:36.714     0x00 0x00 0x00 0x00
Jun  1 10:30:36.714     0x00 0x00 0x00 0x00
Jun  1 10:30:36.714  Obj_class 10 (FILTER_SPEC) ctype 12 length 20:
Jun  1 10:30:36.714   Source 10.0.0.2, lsp_id 1
Jun  1 10:30:36.714   subgrpOrigId 10.0.0.2, subgrpId 1
Jun  1 10:30:36.714  Obj_class 16 (LABEL) ctype 1 length 8:
Jun  1 10:30:36.714   Label: 1024
Jun  1 10:30:36.714  Obj_class 21 (RECORDED_ROUTE) ctype 1 length 20:
Jun  1 10:30:36.714   (#1) ipv4 prefix, 8 bytes, 10.0.0.1/32, flags: 0x00000000
Jun  1 10:30:36.714   (#2) ipv4 prefix, 8 bytes, 10.0.0.2/32, flags: 0x00000000
Jun  1 10:30:36.714  Obj_class 50 (S2L_SUB_LSP) ctype 1 length 8
Jun  1 10:30:36.714   S2L dest addr 10.0.0.2

Jun 12 06:44:07.216 Send ResvTear message: src 10.0.0.1, dst 10.0.0.2 on port 4/11
Jun 12 06:44:07.216  Type 6(ResvTear) ver 1 flags 0x00 cksum 0xee22 ttl 63 len 80
Jun 12 06:44:07.216  Obj_class 1 (SESSION) ctype 13 length 16:
Jun 12 06:44:07.216   p2mpId 10.1.1.1, tunnelid 1, ext tunnelid (source) 10.0.0.2
Jun 12 06:44:07.217  Obj_class 3 (RSVP_HOP) ctype 1 length 12:
```

```
Jun 12 06:44:07.217   Address: 10.0.0.1 LIH: 00000000
Jun 12 06:44:07.217  Obj_class 8 (STYLE) ctype 1 length 8:
Jun 12 06:44:07.217   Style: SE
Jun 12 06:44:07.217  Obj_class 10 (FILTER_SPEC) ctype 12 length 20:
Jun 12 06:44:07.217   Source 10.0.0.2, lsp_id 1
Jun 12 06:44:07.217   subgrpOrigId 10.0.0.2, subgrpId 2
Jun 12 06:44:07.217  Obj_class 16 (LABEL) ctype 1 length 8:
Jun 12 06:44:07.217   Label: 1025
Jun 12 06:44:07.217  Obj_class 50 (S2L_SUB_LSP) ctype 1 length 8
Jun 12 06:44:07.217   S2L dest addr 10.0.1.2

Jun 12 05:44:34.330 Send PathTear message: src 10.0.2.1, dst 10.0.2.2 on port 4/12
Jun 12 05:44:34.330  Type 5(PathTear) ver 1 flags 0x00 cksum 0x7a5f ttl 63 len 100
Jun 12 05:44:34.330  Obj_class 1 (SESSION) ctype 13 length 16:
Jun 12 05:44:34.330   p2mpId 10.1.1.1, tunnelid 1, ext tunnelid (source) 10.0.0.2
Jun 12 05:44:34.330  Obj_class 3 (RSVP_HOP) ctype 1 length 12:
Jun 12 05:44:34.330   Address: 10.0.0.1 LIH: 0000009c
Jun 12 05:44:34.330  Obj_class 11 (SENDER_TEMPLATE) ctype 12 length 20:
Jun 12 05:44:34.330   Source 10.0.0.2, lsp_id 1
Jun 12 05:44:34.330   subgrpOrigId 10.0.0.2, subgrpId 3
Jun 12 05:44:34.330  Obj_class 12 (SENDER_TSPEC) ctype 2 length 36:
Jun 12 05:44:34.330   Max rate: 0, mean rate: 0, max burst: 0
Jun 12 05:44:34.330     0x00 0x00 0x00 0x07
Jun 12 05:44:34.330     0x01 0x00 0x00 0x06
Jun 12 05:44:34.330     0x7f 0x00 0x00 0x05
Jun 12 05:44:34.330     0x00 0x00 0x00 0x00
Jun 12 05:44:34.330     0x00 0x00 0x00 0x00
Jun 12 05:44:34.330     0x00 0x00 0x00 0x00
Jun 12 05:44:34.330     0x00 0x00 0x00 0x00
Jun 12 05:44:34.330     0x00 0x00 0x00 0x00
Jun 12 05:44:34.330  Obj_class 50 (S2L_SUB_LSP) ctype 1 length 8
Jun 12 05:44:34.330   S2L dest addr 10.0.2.2

Jun 12 06:23:01.462 Send PathErr message: src 10.0.0.1, dst 10.0.0.2 on port 4/11
Jun 12 06:23:01.462  Type 3(PathErr) ver 1 flags 0x00 cksum 0x24a0 ttl 63 len 184
Jun 12 06:23:01.462  Obj_class 1 (SESSION) ctype 13 length 16:
Jun 12 06:23:01.462   p2mpId 10.1.1.1, tunnelid 1, ext tunnelid (source) 10.0.0.2
Jun 12 06:23:01.462  Obj_class 6 (ERROR) ctype 1 length 12:
Jun 12 06:23:01.462   Node: 10.11.11.10 Flags: 0x00 code: 23 value 0
Jun 12 06:23:01.462  Obj_class 11 (SENDER_TEMPLATE) ctype 12 length 20:
Jun 12 06:23:01.462   Source 10.0.0.2, lsp_id 1
Jun 12 06:23:01.462   subgrpOrigId 10.0.0.2, subgrpId 2
Jun 12 06:23:01.462  Obj_class 12 (SENDER_TSPEC) ctype 2 length 36:
Jun 12 06:23:01.462   Max rate: 0, mean rate: 0, max burst: 0
Jun 12 06:23:01.462     0x00 0x00 0x00 0x07
Jun 12 06:23:01.462     0x01 0x00 0x00 0x06
Jun 12 06:23:01.462     0x7f 0x00 0x00 0x05
Jun 12 06:23:01.462     0x00 0x00 0x00 0x00
Jun 12 06:23:01.462     0x00 0x00 0x00 0x00
Jun 12 06:23:01.462     0x00 0x00 0x00 0x00
Jun 12 06:23:01.462     0x00 0x00 0x00 0x00
Jun 12 06:23:01.463     0x00 0x00 0x00 0x00
Jun 12 06:23:01.463  Obj_class 13 (ADSPEC) ctype 2 length 84
Jun 12 06:23:01.463     0x00 0x00 0x00 0x13
Jun 12 06:23:01.463     0x01 0x80 0x00 0x08
Jun 12 06:23:01.463     0x04 0x00 0x00 0x01
Jun 12 06:23:01.463     0x00 0x00 0x00 0x00
Jun 12 06:23:01.463     0x06 0x00 0x00 0x01
Jun 12 06:23:01.463     0x00 0x00 0x00 0x00
Jun 12 06:23:01.463     0x08 0x00 0x00 0x01
```

```
Jun 12 06:23:01.463     0x00 0x00 0x00 0x00
Jun 12 06:23:01.463     0x0a 0x00 0x00 0x01
Jun 12 06:23:01.463     0x00 0x00 0x05 0xdc
Jun 12 06:23:01.463     0x02 0x80 0x00 0x08
Jun 12 06:23:01.463     0x85 0x00 0x00 0x01
Jun 12 06:23:01.463     0x00 0x00 0x00 0x00
Jun 12 06:23:01.463     0x86 0x00 0x00 0x01
Jun 12 06:23:01.463     0x00 0x00 0x00 0x00
Jun 12 06:23:01.463     0x87 0x00 0x00 0x01
Jun 12 06:23:01.463     0x00 0x00 0x00 0x00
Jun 12 06:23:01.463     0x88 0x00 0x00 0x01
Jun 12 06:23:01.463     0x00 0x00 0x00 0x00
Jun 12 06:23:01.463     0x05 0x80 0x00 0x00
Jun 12 06:23:01.463  Obj_class 50 (S2L_SUB_LSP) ctype 1 length 8
Jun 12 06:23:01.463   S2L dest addr 10.0.1.2
```

debug mpls rsvp session

Syntax: [no] debug mpls rsvp session [all | detail | lsp]

This command displays MPLS RSVP session-related information.

- **all** - Displays all messages related to a MPLS RSVP session.
- **detail** - Displays messages related to a MPLS RSVP session in a detailed version.
- **lsp** - Limits the display of a MPLS RSVP session to specific LSPs.

Command output resembles the following example.

```
Brocade# debug mpls rsvp session
Processing input queue event "RSVP_pkt" for SESS(10.100.100.100
/1/10.20.20.20) Destp 0X142ED82C
Processing input queue event "Path_refresh_tmr_exp" for SESS(10
.100.100.100/1/10.20.20.20) Destp 0X142ED82C
Processing input queue event "Path_refresh_tmr_exp" for SESS(10
.100.100.100/2/10.20.20.20) Destp 0X142ECC80
Processing input queue event "Resv_refresh_tmr_exp" for SESS(10
.100.100.100/2/10.20.20.20) Destp 0X142ECC80
Processing input queue event "RSVP_pkt" for SESS(10.100.100.100
/1/10.20.20.20) Destp 0X142ED82C
Processing input queue event "Resv_refresh_tmr_exp" for SESS(10
.100.100.100/1/10.20.20.20) Destp 0X142ED82C
Processing input queue event "RSVP_pkt" for SESS(10.100.100.100
/2/10.20.20.20) Destp 0X142ECC80
Processing input queue event "Path_refresh_tmr_exp" for SESS(10
.100.100.100/1/10.20.20.20) Destp 0X142ED82C
Processing input queue event "Resv_refresh_tmr_exp" for SESS(10
.100.100.100/2/10.20.20.20) Destp 0X142ECC80
Processing input queue event "RSVP_pkt" for SESS(10.100.100.100
/1/10.20.20.20) Destp 0X142ED82C
Processing input queue event "Path_refresh_tmr_exp" for SESS(10
.100.100.100/2/10.20.20.20) Destp 0X142ECC80
Processing input queue event "Path_refresh_tmr_exp" for SESS(10
.100.100.100/1/10.20.20.20) Destp 0X142ED82C
Processing input queue event "Resv_refresh_tmr_exp" for SESS(10
.100.100.100/2/10.20.20.20) Destp 0X142ECC80
Processing input queue event "RSVP_pkt" for SESS(10.100.100.100
/1/10.20.20.20) Destp 0X142ED82C
Processing input queue event "Resv_refresh_tmr_exp" for SESS(10
.100.100.100/1/10.20.20.20) Destp 0X142ED82C
Processing input queue event "RSVP_pkt" for SESS(10.100.100.100
```

```
/2/10.20.20.20) Destp 0X142ECC80
Processing input queue event "Path_refresh_tmr_exp" for SESS(10
.100.100.100/2/10.20.20.20) Destp 0X142ECC80
```

debug mpls rsvp session lsp

Syntax: [**no**] **debug mpls rsvp session lsp** [**name** *name*| **sess_obj** *source_ip_address* [*destination_ip_address* | *p2mp-Id*] *tunnel_id*]

- **name** *name* - Displays RSVP session information for the specified LSP name.
- **sess_obj** - Displays RSVP session information for the specified RSVP session object or P2MP session object. RSVP session object includes source IP address, destination IP address, and tunnel ID. P2MP session object incudes source IP address, P2MP ID, and tunnel ID.
  - *source_ip_address* - Specifies the source IP address.
  - *destination_ip_address* - Specifies the destination IP address.
  - *p2mp-Id* - Specifies the P2MP ID in decimal or IP address format. This variable is applicable to P2MP LSPs only and can be used to filter the debug tracing based on P2MP session object.
  - *tunnel_id* - Specifies the tunnel ID.

This command displays RSVP session information for specific LSPs. Command output resembles the following example.

```
Brocade# debug mpls rsvp session lsp sess_obj 10.0.0.0 10.0.0.1 100
Dec 11 20:01:32.344 SESS(10.0.0.1/29433/10.0.0.1): Add  PSB(0x0X31183BC4)
Dec 11 20:01:32.344 SESS(10.0.0.1/29433/10.0.0.1): path for PSB(0x0X31183BC4)
received, new ttd: 597811550, ps_tc_flags: 0X0000001C
Dec 11 20:01:32.345 RSVP: New TC action QUERY_ROUTE for
Sess(10.0.0.1/29433/10.0.0.1), destp 0x0X2E779B14
Dec 11 20:01:32.345 SESS(10.0.0.1/29433/10.0.0.1): Merge not possible for
protected path (in-if 148)
Dec 11 20:01:32.345 SESS(10.0.0.1/29433/10.0.0.1): Found route to dest, nhop
31.31.31.16
Dec 11 20:01:32.345 SESS(10.0.0.1/29433/10.0.0.1): Merge not possible for
protected path (in-if 148)
Dec 11 20:01:32.345 SESS(10.0.0.1/29433/10.0.0.1): TC-action(QUERY_ROUTE)
finished
Dec 11 20:01:32.345 RSVP: Free TC action QUERY_ROUTE for
Sess(10.0.0.1/29433/10.0.0.1), destp 0x0X2E779B14
Dec 11 20:01:32.345 RSVP: New TC action LDB_RESERVE for
Sess(10.0.0.1/29433/10.0.0.1), destp 0x0X2E779B14
Dec 11 20:01:32.345 SESS(10.0.0.1/29433/10.0.0.1): refresh PSB(0x0X31183BC4)
Dec 11 20:01:32.345 SESS(10.0.0.1/29433/10.0.0.1): Tx PATH to 10.31.31.16
Dec 11 20:01:32.345 SESS(10.0.0.1/29433/10.0.0.1): RSVP->LMGR(Reserve) sent,
PSB(0x0X31183BC4)
Dec 11 20:01:32.345   Lsp/Grp 0x0X00000000(2)/0x0X00000000(2), Osg/Grp
0x0X00000000(0)/0x0X00000000(0)
Dec 11 20:01:32.345   Isg/Grp 0x0X00000000(2)/0x0X00000000(2), XC
0x0X00000000(0), TCSBP 0x0X00000000
Dec 11 20:01:32.345 SESS(10.0.0.1/29433/10.0.0.1): TC-action(LDB_RESERVE)
deferred
Dec 11 20:01:32.346 SESS(10.0.0.1/29433/10.0.0.1), LMGR->RSVP(Reserve) finished:
PSB 0x0X31183BC4
Dec 11 20:01:32.346   Lsp/Grp 0x0X00998001(2)/0x0X00990001(2), Osg/Grp
0x0X00000000(0)/0x0X00000000(0)
Dec 11 20:01:32.346   Isg/Grp 0x0X008B4001(2)/0x0X008AC001(2), XC
0x0X00000000(0), TCSBP 0x0X00000000
```

```
Dec 11 20:01:32.346 RSVP: Free TC action LDB_RESERVE for
Sess(10.0.0.1/29433/10.0.0.1), destp 0x0X2E779B14
Dec 11 20:01:32.346 Processing input queue event "RSVP_pkt" for
SESS(10.0.0.1/29433/10.0.0.1) Destp 0X2E779B14
Dec 11 20:01:32.347 SESS(10.0.0.1/29433/10.0.0.1): Add  PSB(0x0X311833C8)
Dec 11 20:01:32.347 SESS(10.0.0.1/29433/10.0.0.1): path for PSB(0x0X311833C8)
received, new ttd: 597811550, ps_tc_flags: 0X0000001C
Dec 11 20:01:32.347 RSVP: New TC action QUERY_ROUTE for
Sess(10.0.0.1/29433/10.0.0.1), destp 0x0X2E779B14
Dec 11 20:01:32.347 SESS(10.0.0.1/29433/10.0.0.1): Merge not possible for
protected path (in-if 148)
Dec 11 20:01:32.347 SESS(10.0.0.1/29433/10.0.0.1): Found route to dest, nhop
31.31.31.16
Dec 11 20:01:32.347 SESS(10.0.0.1/29433/10.0.0.1): Merge not possible for
protected path (in-if 148)
Dec 11 20:01:32.347 SESS(10.0.0.1/29433/10.0.0.1): TC-action(QUERY_ROUTE)
finished
Dec 11 20:01:32.347 RSVP: Free TC action QUERY_ROUTE for
Sess(10.0.0.1/29433/10.0.0.1), destp 0x0X2E779B14
Dec 11 20:01:32.347 RSVP: New TC action LDB_RESERVE for
Sess(10.0.0.1/29433/10.0.0.1), destp 0x0X2E779B14
Dec 11 20:01:32.347 SESS(10.0.0.1/29433/10.0.0.1): refresh PSB(0x0X311833C8)
Dec 11 20:01:32.347 SESS(10.0.0.1/29433/10.0.0.1): Tx PATH to 10.31.31.16
Dec 11 20:01:32.347 SESS(10.0.0.1/29433/10.0.0.1): RSVP->LMGR(Reserve) sent,
PSB(0x0X311833C8)
Dec 11 20:01:32.347   Lsp/Grp 0x0X00998001(1)/0x0X00990001(1), Osg/Grp
0x0X00000000(0)/0x0X00000000(0)
Dec 11 20:01:32.347   Isg/Grp 0x0X00000000(2)/0x0X00000000(2), XC
0x0X00000000(0), TCSBP 0x0X00000000
Dec 11 20:01:32.347 SESS(10.0.0.1/29433/10.0.0.1): TC-action(LDB_RESERVE)
deferred
Dec 11 20:01:32.348 SESS(10.0.0.1/29433/10.0.0.1), LMGR->RSVP(Reserve) finished:
PSB 0x0X311833C8
Dec 11 20:01:32.348   Lsp/Grp 0x0X00998001(1)/0x0X00990001(1), Osg/Grp
0x0X00000000(0)/0x0X00000000(0)
Dec 11 20:01:32.348   Isg/Grp 0x0X008B0001(2)/0x0X008AC001(1), XC
0x0X00000000(0), TCSBP 0x0X00000000
Dec 11 20:01:32.348 RSVP: Free TC action LDB_RESERVE for
Sess(10.0.0.1/29433/10.0.0.1), destp 0x0X2E779B14
Dec 11 20:01:32.348 Processing input queue event "RSVP_pkt" for
SESS(10.0.0.1/29433/10.0.0.1) Destp 0X2E779B14
Dec 11 20:01:32.348 SESS(10.0.0.1/29433/10.0.0.1): Add  PSB(0x0X31182BCC)
Dec 11 20:01:32.348 SESS(10.0.0.1/29433/10.0.0.1): path for PSB(0x0X31182BCC)
received, new ttd: 597811550, ps_tc_flags: 0X0000001C
Dec 11 20:01:32.348 RSVP: New TC action QUERY_ROUTE for
Sess(10.0.0.1/29433/10.0.0.1), destp 0x0X2E779B14
Dec 11 20:01:32.348 SESS(10.0.0.1/29433/10.0.0.1): Merge not possible for
protected path (in-if 148)
Dec 11 20:01:32.348 SESS(10.0.0.1/29433/10.0.0.1): Found route to dest, nhop
31.31.31.16
Dec 11 20:01:32.348 SESS(10.0.0.1/29433/10.0.0.1): Merge not possible for
protected path (in-if 148)
Dec 11 20:01:32.348 SESS(10.0.0.1/29433/10.0.0.1): TC-action(QUERY_ROUTE)
finished
Dec 11 20:01:32.348 RSVP: Free TC action QUERY_ROUTE for
Sess(10.0.0.1/29433/10.0.0.1), destp 0x0X2E779B14
Dec 11 20:01:32.348 RSVP: New TC action LDB_RESERVE for
Sess(10.0.0.1/29433/10.0.0.1), destp 0x0X2E779B14
Dec 11 20:01:32.349 SESS(10.0.0.1/29433/10.0.0.1): refresh PSB(0x0X31182BCC)
Dec 11 20:01:32.349 SESS(10.0.0.1/29433/10.0.0.1): Tx PATH to 10.31.31.16
```

```
Dec 11 20:01:32.349 SESS(10.0.0.1/29433/10.0.0.1): RSVP->LMGR(Reserve) sent,
PSB(0x0X31182BCC)
Dec 11 20:01:32.349   Lsp/Grp 0x0X00998001(1)/0x0X00990001(1), Osg/Grp
0x0X00000000(0)/0x0X00000000(0)
Dec 11 20:01:32.349   Isg/Grp 0x0X00000000(2)/0x0X00000000(2), XC
0x0X00000000(0), TCSBP 0x0X00000000
Dec 11 20:01:32.349 SESS(10.0.0.1/29433/10.0.0.1): TC-action(LDB_RESERVE)
deferred
Dec 11 20:01:32.349 SESS(10.0.0.1/29433/10.0.0.1), LMGR->RSVP(Reserve) finished:
PSB 0x0X31182BCC
Dec 11 20:01:32.349   Lsp/Grp 0x0X00998001(1)/0x0X00990001(1), Osg/Grp
0x0X00000000(0)/0x0X00000000(0)
Dec 11 20:01:32.349   Isg/Grp 0x0X008BC001(2)/0x0X008AC001(1), XC
0x0X00000000(0), TCSBP 0x0X00000000
Dec 11 20:01:32.349 RSVP: Free TC action LDB_RESERVE for
Sess(10.0.0.1/29433/10.0.0.1), destp 0x0X2E779B14
Dec 11 20:01:32.479 Processing input queue event "RSVP_pkt" for
SESS(10.0.0.1/29433/10.0.0.1) Destp 0X2E779B14
Dec 11 20:01:32.479 SESS(10.0.0.1/29433/10.0.0.1): Rx RESV, lbl 0X00000000
Dec 11 20:01:32.479 RSVP: New TC action LDB_CONNECT for
Sess(10.0.0.1/29433/10.0.0.1), destp 0x0X2E779B14
Dec 11 20:01:32.479 SESS(10.0.0.1/29433/10.0.0.1): RSVP->LMGR(Connect) sent,
PSB(0x0X31183BC4)
Dec 11 20:01:32.479   Lsp/Grp 0x0X00998001(1)/0x0X00990001(1), Osg/Grp
0x0X00000000(2)/0x0X00000000(2)
Dec 11 20:01:32.479   Isg/Grp 0x0X008B4001(1)/0x0X008AC001(1), XC
0x0X00000000(2), TCSBP 0x0X2E6CA908
Dec 11 20:01:32.479 SESS(10.0.0.1/29433/10.0.0.1): TC-action(LDB_CONNECT)
deferred
Dec 11 20:01:32.480 SESS(10.0.0.1/29433/10.0.0.1), LMGR->RSVP(Connect) finished:
PSB 0x0X31183BC4
Dec 11 20:01:32.480   Lsp/Grp 0x0X00998001(1)/0x0X00990001(1), Osg/Grp
0x0X00874001(2)/0x0X008B8001(2)
Dec 11 20:01:32.480   Isg/Grp 0x0X008B4001(1)/0x0X008AC001(1), XC
0x0X0087C001(2), TCSBP 0x0X2E6CA908
Dec 11 20:01:32.480 RSVP: Free TC action LDB_CONNECT for
Sess(10.0.0.1/29433/10.0.0.1), destp 0x0X2E779B14
Dec 11 20:01:32.480 SESS(10.0.0.1/29433/10.0.0.1): Tx RESV, out_if 149 lbl 2356
Dec 11 20:01:32.684 Processing input queue event "RSVP_pkt" for
SESS(10.0.0.1/29433/10.0.0.1) Destp 0X2E779B14
Dec 11 20:01:32.684 Processing input queue event "RSVP_pkt" for
SESS(10.0.0.1/29433/10.0.0.1) Destp 0X2E779B14
Dec 11 20:01:32.684 RSVP: New TC action LDB_CONNECT for
Sess(10.0.0.1/29433/10.0.0.1), destp 0x0X2E779B14
Dec 11 20:01:32.684 SESS(10.0.0.1/29433/10.0.0.1): RSVP->LMGR(Connect) sent,
PSB(0x0X31182BCC)
Dec 11 20:01:32.684   Lsp/Grp 0x0X00998001(1)/0x0X00990001(1), Osg/Grp
0x0X00000000(2)/0x0X008B8001(1)
Dec 11 20:01:32.684   Isg/Grp 0x0X008BC001(1)/0x0X008AC001(1), XC
0x0X00000000(2), TCSBP 0x0X2E6CA908
Dec 11 20:01:32.684 SESS(10.0.0.1/29433/10.0.0.1): TC-action(LDB_CONNECT)
deferred
Dec 11 20:01:32.685 SESS(10.0.0.1/29433/10.0.0.1), LMGR->RSVP(Connect) finished:
PSB 0x0X31182BCC
Dec 11 20:01:32.685   Lsp/Grp 0x0X00998001(1)/0x0X00990001(1), Osg/Grp
0x0X009EC001(2)/0x0X008B8001(1)
Dec 11 20:01:32.685   Isg/Grp 0x0X008BC001(1)/0x0X008AC001(1), XC
0x0X00A08001(2), TCSBP 0x0X2E6CA908
Dec 11 20:01:32.685 RSVP: Free TC action LDB_CONNECT for
Sess(10.0.0.1/29433/10.0.0.1), destp 0x0X2E779B14
```

```
Dec 11 20:01:32.685 SESS(10.0.0.1/29433/10.0.0.1): Tx RESV, out_if 149 lbl 2356
Dec 11 20:01:35.281 Processing input queue event "RSVP_pkt" for
SESS(10.0.0.1/29433/10.0.0.1) Destp 0X2E779B14
Dec 11 20:01:35.281 SESS(10.0.0.1/29433/10.0.0.1): path for PSB(0x0X31183BC4)
received, new ttd: 597814450, ps_tc_flags: 0X00000020
Dec 11 20:01:35.281 Processing input queue event "RSVP_pkt" for
SESS(10.0.0.1/29433/10.0.0.1) Destp 0X2E779B14
Dec 11 20:01:35.281 SESS(10.0.0.1/29433/10.0.0.1): path for PSB(0x0X311833C8)
received, new ttd: 597814450, ps_tc_flags: 0X00000020
Dec 11 20:01:35.282 Processing input queue event "RSVP_pkt" for
SESS(10.0.0.1/29433/10.0.0.1) Destp 0X2E779B14
Dec 11 20:01:35.282 SESS(10.0.0.1/29433/10.0.0.1): path for PSB(0x0X31182BCC)
received, new ttd: 597814450, ps_tc_flags: 0X00000020
Dec 11 20:01:36.483 SESS(10.0.0.1/29433/10.0.0.1): PSB(0x0X31183BC4), srefresh
send path
Dec 11 20:01:36.483 SESS(10.0.0.1/29433/10.0.0.1): PSB(0x0X311833C8), srefresh
send path
Dec 11 20:01:36.483 SESS(10.0.0.1/29433/10.0.0.1): PSB(0x0X31182BCC), srefresh
send path
Dec 11 20:01:36.483 Processing input queue event "Clean_path_state_tmr_exp" for
SESS(10.0.0.1/29433/10.0.0.1) Destp 0X2E779B14
Dec 11 20:01:36.483 Processing input queue event "Clean_path_state_tmr_exp" for
SESS(10.0.0.1/29433/10.0.0.1) Destp 0X2E779B14
Dec 11 20:01:36.483 Processing input queue event "Clean_path_state_tmr_exp" for
SESS(10.0.0.1/29433/10.0.0.1) Destp 0X2E779B14
Dec 11 20:01:38.278 SESS(10.0.0.1/29433/10.0.0.1): srefresh receive resv,
RSB(0x0X2E70D0C0), new ttd: 151, fsb_idx: 0
```

debug mpls rsvp tunnel

Syntax: [**no**] **debug mpls rsvp tunnel** [**all** | **detail** | **lsp**]

This command displays MPLS RSVP LSP tunnel interface-related information.

- **all** - Displays all messages related to a MPLS RSVP tunnel.
- **detail** - Displays detailed information about RSVP tunnel state transitions for all tunnels and their retries whenever the retry timer is expired.
- **lsp** - Displays RSVP tunnel information to specific LSPs.

Command output resembles the following example.

```
Brocade# debug mpls rsvp tunnel
MPLS: TNNL(test1): try signal LSP
MPLS: TNNL(test1): event = 2(ENABLE_CSPF_OK), change from state 3(PATH_SENT) to
2(ROUTE_FOUND)
MPLS: TNNL(test1): event = 31(SENT_PATH), change from state 2(ROUTE_FOUND) to
3(PATH_SENT)
RSVP_TNNL(test1): Update tunnel_vif_index 2
RSVP_TNNL(test1): Update tunnel oper old 0, new 1
MPLS: TNNL(test1): tnl 2 goes up
MPLS: TNNL(test1): primary(current instance), path path1 up
MPLS: TNNL(test1): activate primary
MPLS: TNNL(test1): tnl 2 added to mpls route table and indicated to application
MPLS: TNNL(test1): notify IP with vif 2 UP notification
```

debug mpls rsvp tunnel lsp

Syntax: [**no**] **debug mpls rsvp tunnel lsp** [**name** *name* | **sess_obj** *source_ip_address destination_ip_address tunnel_id*]

This command displays RSVP tunnel information for specific LSPs.

- **name** *name* - Limits the display of information to debug messages for the specified LSP name.
- **sess_obj** *source_ip_address destination_ip_address tunnel_id* - Limits the display of information to debug messages for the specified LSP session object, which includes source IP address, destination IP address, and tunnel ID.

Command output resembles the following example.

```
Brocade# debug mpls rsvp tunnel lsp name PE3
Brocade# debug mpls
Brocade# clear mpls lsp PE3
Disconnecting signaled LSP PE3
Dec 10 01:17:49 MPLS: TNNL(PE3): tnl 1 deleted from mpls route table and indicated
to application
Dec 10 01:17:49 MPLS: TNNL(PE3): possible delete LP with tunnel vif 1 using vif
index 1
Dec 10 01:17:49 MPLS: TNNL(PE3): FRR_FWD_FSM:
        old_st Pri(active), Sec(down), Det(up)
        evt FRR_FWD_EVT_PRI_DE_ACT
        new_st Pri(up), Sec(down), Det(up)
Dec 10 01:17:49 MPLS: TNNL(PE3): FRR_FWD_FSM: action_none
Dec 10 01:17:49 RSVP_TNNL(PE3): delete from LP with tunnel vif 1 using vif index 1
Dec 10 01:17:49 RSVP TNNL(PE3): Update tunnel_vif_index 1
Dec 10 01:17:49 RSVP_TNNL(PE3): Update tunnel oper old 1, new 0
Dec 10 01:17:49 MPLS: TNNL: tnl 1 goes down
Dec 10 01:17:49 MPLS: TNNL(PE3): primary down
Dec 10 01:17:49 MPLS: TNNL(PE3): event = 21(DOWN_REROUTE_OR_NO_CSPF), change from
state 3(PATH_SENT) to 2(ROUTE_FOUND)
Dec 10 01:17:49 MPLS: TNNL(PE3): tnl 1 deleted from mpls route table and indicated
to application
Dec 10 01:17:49 MPLS: TNNL(PE3): suppress vif 1 DOWN notification
Dec 10 01:17:49 MPLS: TNNL(PE3): FRR_FWD_FSM:
        old_st Pri(up), Sec(down), Det(up)
        evt FRR_FWD_EVT_FAULT_ON_FRR
        new_st Pri(down), Sec(down), Det(down)
Dec 10 01:17:49 MPLS: TNNL(PE3): FRR_FWD_FSM: action_none
Dec 10 01:17:49 MPLS: TNNL(PE3, detour): event = 41(RX_PATHERR), change from state
3(PATH_SENT) to 4(PATH_ERROR)
Dec 10 01:17:49 MPLS: TNNL(PE3, detour): change from state 3(ENABLE_CSPF_FAIL) to
0(INIT)
Dec 10 01:17:49 MPLS: TNNL(PE3): FRR_FWD_FSM:
        old_st Pri(down), Sec(down), Det(down)
        evt FRR_FWD_EVT_FAULT_DET
        new_st Pri(down), Sec(down), Det(down)
Dec 10 01:17:49 MPLS: TNNL(PE3): FRR_FWD_FSM: action_none
Dec 10 01:17:49 MPLS: TNNL(PE3, detour): change from state 0(UNKNOWN_EVT) to
0(INIT)
Dec 10 01:17:49 MPLS: TNNL(PE3): FRR_FWD_FSM:
        old_st Pri(down), Sec(down), Det(down)
        evt FRR_FWD_EVT_FAULT_DET
        new_st Pri(down), Sec(down), Det(down)
Dec 10 01:17:49 MPLS: TNNL(PE3): FRR_FWD_FSM: action_none
Connecting signaled LSP PE3
Dec 10 01:17:49 MPLS: TNNL(PE3): try signal LSP
Dec 10 01:17:49 MPLS: TNNL(PE3): event = 31(SENT_PATH), change from state
2(ROUTE_FOUND) to 3(PATH_SENT)
```

MPLS label manager debug commands

This section describes the debug commands that generate MPLS label manager information.

debug mpls lmgr

Syntax: [**no**] **debug mpls lmgr** [**all** | **error** | **ldp** | **rsvp**]

This command displays label manager-related information.

- **all** - Displays all the debug messages related to label manager.
- **error** - Displays label manager-related error messages.
- **ldp** - Displays label manager information for LDP.
- **rsvp** - Displays label manager information for RSVP.

Command output resembles the following example.

```
Brocade# debug mpls lmgr
LMGR 1 sent LSI_COMMON_LSP_XC SESSION (10.100.100.100, 0X0007, 10.20.20.20) LSP
(0X0001, 10.20.20.20), ref_flags (2,4,0,0,2,0,0), update_flags 0X00000011,
lspxc_flags 0X00000000, rvs_ref flags (0,0,0,0,0,0,0), rvs_update_flags
0X00000000, in_seg_index 0, xc_index 5, out_seg_index 0.
LMGR 1 rcvd LSI_COMMON_LSP_XC SESSION (10.100.100.100, 0X0007, 10.20.20.20) LSP
(0X0001, 10.20.20.20), ref_flags (2,4,0,0,2,0,0), update_flags 0X00000011,
lspxc_flags 0X00000000, rvs_ref flags (0,0,0,0,0,0,0), rvs_update_flags
0X00000000, rc 1, in_seg_index 0, xc_index 5, out_seg_index 0.
LMGR 1 sent LSI_COMMON_LSP_XC SESSION (10.100.100.100, 0X0007, 10.20.20.20) LSP
(0X0001, 10.20.20.20), ref_flags (1,4,2,2,1,0,2), update_flags 0X0000004C,
lspxc_flags 0X00000000, rvs_ref flags (0,0,0,0,0,0,0), rvs_update_flags
0X00000000, in_seg_index 0, xc_index 5, out_seg_index 4.
LMGR 1 rcvd LSI_COMMON_LSP_XC SESSION (10.100.100.100, 0X0007, 10.20.20.20) LSP
(0X0001, 10.20.20.20), ref_flags (1,4,2,2,1,0,2), update_flags 0X0000004C,
lspxc_flags 0X00000000, rvs_ref flags (0,0,0,0,0,0,0), rvs_update_flags
0X00000000, rc 1, in_seg_index 0, xc_index 5, out_seg_index 4.
```

debug mpls lmgr rsvp

Syntax: [**no**] **debug mpls lmgr rsvp** [**all** | **lsp** [**name** *name* | **sess_obj** *source_ip_address destination_ip_address tunnel_id*]]

This command displays label manager information for RSVP.

- **all** - Displays all debug messages related to label manager for RSVP.
- **lsp** - Displays label manager information for RSVP for specific LSPs.
- **name** *name* - Limits the display of information to debug messages for the specified LSP name.
- **sess_obj** *source_ip_address destination_ip_address tunnel_id* - Limits the display of information to debug messages for the specified LSP session object, which includes source IP address, destination IP address, and tunnel ID.

Command output resembles the following example.

```
Brocade# debug mpls lmgr rsvp all
Brocade# clear mpls lsp PE3
Disconnecting signaled LSP PE3
Connecting signaled LSP PE3
Dec 10 01:20:04 LMGR 1 sent LSI_COMMON_LSP_XC SESSION (10.7.7.3, 0X1E24, 10.7.7.2)
LSP (0X0001, 10.7.7.2),
ref_flags (2,4,0,0,2,0,0), update_flags 0X00000011, lspxc_flags 0X00000000,
```

```
rvs_ref flags (0,0,0,0,0,0,0), rvs_update_flags 0X00000000,
in_seg_index 0, xc_index 4251, out_seg_index 0.
Dec 10 01:20:04 LMGR 1 rcvd LSI_COMMON_LSP_XC SESSION (10.7.7.3, 0X1E24, 10.7.7.2)
LSP (0X0001, 10.7.7.2),
ref_flags (2,4,0,0,2,0,0), update_flags 0X00000011, lspxc_flags 0X00000000,
rvs_ref flags (0,0,0,0,0,0,0), rvs_update_flags 0X00000000, rc 1,
in_seg_index 0, xc_index 4251, out_seg_index 0.
Dec 10 01:20:04 LMGR 1 sent LSI_COMMON_LSP_XC SESSION (10.7.7.3, 0X1E24, 10.7.7.2)
LSP (0X0001, 10.7.7.2),
ref_flags (1,4,2,2,1,0,2), update_flags 0X0000004C, lspxc_flags 0X00000000,
rvs_ref flags (0,0,0,0,0,0,0), rvs_update_flags 0X00000000,
in_seg_index 0, xc_index 4251, out_seg_index 4071.
Dec 10 01:20:04 LMGR 1 rcvd LSI_COMMON_LSP_XC SESSION (10.7.7.3, 0X1E24, 10.7.7.2)
LSP (0X0001, 10.7.7.2),
ref_flags (1,4,2,2,1,0,2), update_flags 0X0000004C, lspxc_flags 0X00000000,
rvs_ref flags (0,0,0,0,0,0,0), rvs_update_flags 0X00000000, rc 1,
in_seg_index 0, xc_index 4251, out_seg_index 4071.
Dec 10 01:20:04 LMGR 1 sent LSI_COMMON_LSP_XC SESSION (10.7.7.3, 0X1E24, 10.7.7.2)
LSP (0X0001, 10.7.7.2),
ref_flags (1,4,2,2,1,0,2), update_flags 0X0000004C, lspxc_flags 0X0000000C,
rvs_ref flags (0,0,0,0,0,0,0), rvs_update_flags 0X00000000,
in_seg_index 0, xc_index 4251, out_seg_index 4072.
Dec 10 01:20:04 LMGR 1 rcvd LSI_COMMON_LSP_XC SESSION (10.7.7.3, 0X1E24, 10.7.7.2)
LSP (0X0001, 10.7.7.2),
ref_flags (1,4,2,2,1,0,2), update_flags 0X0000004C, lspxc_flags 0X0000000C,
rvs_ref flags (0,0,0,0,0,0,0), rvs_update_flags 0X00000000, rc 1,
in_seg_index 0, xc_index 4251, out_seg_index 4072.
```

VLL debug commands

This section describes the debug commands that generate VLL information.

debug vll

Syntax: [**no**] **debug vll** [**events** | **fsm** | **ipc**]

- **events** - Displays debugging information related to VLL events such as tunnel change, and bitmap change for syslogs.
- **fsm** - Displays debugging information related to all VLL Finite State Machines (FSMs).
- **ipc** - Displays debugging information related to Interprocess Communication (IPC) messages sent to the LP.

Command output resembles the following example.

```
Brocade# debug vll fsm
          MPLS:  VLL FSM(Finite State Machine) debugging is on
Brocade# debug vll events
          MPLS:  VLL Events debugging is on
Brocade# debug vll ipc
          MPLS:  VLL IPC debugging is on
Brocade(config)# router mpls
Brocade(config-mpls)# no vll mct-vll-1 1
Nov 21 02:42:51 VLL EVENTS: Unconfigured VLL 1 in Tagged-mode(default-mode)
Nov 21 02:42:51 VLL_MCT_SYNC_MSG: VC-ID: 1 Send sync message (Type: [VLL
deletion]) is Success (err = 0)
Nov 21 02:42:51 VLL EVENTS: Sending VC withdrawal for VLL 1, vll-index 0, pw-index
0x00000003
Nov 21 02:42:51 VLL EVENTS: Bitmap Event vll-index 0, min-index 0, max-index 0
```

```
Nov 21 02:42:51 VLL EVENTS: Bitmap event is DOWN, min-index 0, max-index 0
Nov 21 02:42:51 VLL EVENTS: Final Index min_index 0, max-index 0
Nov 21 02:42:51 VLL IPC: Sending IPC to LP as VLL 1 changed to  non-operational
with config-deletion
Nov 21 02:42:51 VLL DY-SYNC: Trunk Outbound - vll_name = mct-vll-1, vll_id = 1,
vll_port = 96, vlan_id = 401
Nov 21 02:42:51 VLL DY-SYNC: Peer 10.0.0.3, remote_vc = 798761, tunnel_label = 0,
mode TAGGED-MODE, action DELETE, PW_role Active, MCT_role Active, MCT_ID = 1,
MCT_Client_ID = 3
Nov 21 02:42:51 VLL DY-SYNC: vll_id = 1, vclabel = 798761, Peer 10.0.0.3, action
DELETE, PW_role Spoke, Forward_type PE_to_DROP, MCT_ID = 1, MCT_Client_ID = 3
Nov 21 02:42:51 VLL DY-SYNC: vll_id = 1, vclabel = 798773, Peer 10.0.0.1, action
ADD, PW_role Standby, Forward_type PE_to_DROP, MCT_ID = 1, MCT_Client_ID = 3
Nov 21 02:42:51 VLL DY-SYNC: vll_id = 1, vclabel = 798774, Peer 10.0.0.2, action
ADD, PW_role Active, Forward_type PE_to_EP, MCT_ID = 1, MCT_Client_ID = 3
Nov 21 02:42:51 VLL_Peer_FSM: VC-ID: 1, Peer: 10.0.0.3
[Operational] -> [Waiting for VC Withdraw] (Event: [Peer delete])
Nov 21 02:42:51 VLL EVENTS: Bitmap event is DOWN, min-index 0, max-index 0
Nov 21 02:42:51 VLL EVENTS: Final Index min_index 0, max-index 0
Nov 21 02:42:51 VLL_MCT_FSM: VC-ID: 1
[Active] -> [None] (Event: [None])
Nov 21 02:42:51 VLL EVENTS: Sending VC withdrawal for VLL 1, vll-index 0, pw-index
0x00000001
Nov 21 02:42:51 VLL EVENTS: Bitmap Event vll-index 0, min-index 0, max-index 0
Nov 21 02:42:51 VLL EVENTS: Bitmap event is DOWN, min-index 0, max-index 0
Nov 21 02:42:51 VLL EVENTS: Final Index min_index 0, max-index 0
Nov 21 02:42:51 VLL IPC: Sending IPC to LP as VLL 1 changed to  non-operational
state
Nov 21 02:42:51 VLL DY-SYNC: Trunk Outbound - vll_name = mct-vll-1, vll_id = 1,
vll_port = 96, vlan_id = 401
.
. (output edited for brevity)
.
Nov 21 02:42:51 VLL EVENTS:      VLL VC ID 5 for Peer 10.0.0.3:  current tnnl =
2160, new tnnl = 2157
Nov 21 02:42:51 VLL EVENTS:      VLL VC ID 4 for Peer 10.0.0.3:  current tnnl =
2160, new tnnl = 2157
Nov 21 02:42:51 VLL EVENTS:      VLL VC ID 20 for Peer 10.0.0.3:  current tnnl =
2160, new tnnl = 2157
Nov 21 02:42:51 VLL EVENTS:      VLL VC ID 19 for Peer 10.0.0.3:  current tnnl =
2157, new tnnl = 2157
Nov 21 02:42:51 VLL EVENTS : ** End peer 10.0.0.3

Brocade(config-mpls)# vll mct-vll-1 1
Nov 21 02:43:16 VLL EVENTS: VLL ID 1 has allocated vll-index 0
Nov 21 02:43:16 VLL EVENTS: Configured VLL 1 in Tagged-mode(default-mode)
Brocade(config-mpls-vll-mct-vll-1)# vll-peer 10.0.0.1 10.0.0.2
Brocade(config-mpls-vll-mct-vll-1)# vlan 401
Brocade(config-mpls-vll-mct-vll-1-vlan-401)# tagged e 3/1
Nov 21 02:43:27 VLL IPC: Sending IPC to LP for VLL 1 to program the drop cam
Nov 21 02:43:27 VLL DY-SYNC: Trunk Outbound - vll_name = mct-vll-1, vll_id = 1,
vll_port = 96, vlan_id = 401
Nov 21 02:43:27 VLL DY-SYNC: Peer 0.0.0.0, remote_vc = 0, tunnel_label = 0, mode
TAGGED-MODE, action UPDATE, PW_role Active, MCT_role None, MCT_ID = 0,
MCT_Client_ID = 0
.
. (output edited for brevity)
.
```

```
Nov 21 02:43:27 VLL_MCT_SYNC_MSG: VC-ID: 1 Received sync message (Type: [MCT
status])
Nov 21 02:43:27 VLL_MCT_SYNC_MSG: VC-ID: 1 Send sync message (Type: [MCT status])
is Success (err = 0)
Nov 21 02:43:27 VLL_MCT_SYNC_MSG: VC-ID: 5 Send sync message (Type: [MCT status])
is Success (err = 0)
Nov 21 02:43:27 VLL_MCT_SYNC_MSG: VC-ID: 5 Received sync message (Type: [MCT
status])
Nov 21 02:43:27 VLL_MCT_SYNC_MSG: VC-ID: 1 Received sync message (Type: [MCT
status])
Nov 21 02:43:27 VLL_MCT_SYNC_MSG: VC-ID: 5 Received sync message (Type: [MCT
status])
Nov 21 02:43:27 VLL_MCT_SYNC_MSG: VC-ID: 1 Received sync message (Type: [MCT
status])
```

debug vll-local

Syntax: [**no**] **debug vll-local**

This command displays information about activity in local Virtual Lease Line (VLL-Local) configurations. In the following example, VLL-local 10 is turned off, and then turned on again. With **debug vll-local** enabled, the activity around this event is displayed.

```
Brocade# debug vll-local
 DEBUG VLL LOCAL : MP_VLL_LOCAL_DY_SYNC: name = 10, vll_local_id = 10
                 :port1 = 65535, vlan_id1 = 1 cos1 = 0
                 :port2 = 23, vlan_id2 = 501, cos2 = 0 state = 0
 DEBUG VLL LOCAL : MP_VLL_LOCAL_DY_SYNC: 0x029a6862 1 102
 DEBUG VLL LOCAL : state change: name = 10, vll_local_id = 10, state = 0 oldState
= 1
 DEBUG VLL LOCAL : MP_VLL_LOCAL_DY_SYNC: name = 10, vll_local_id = 10
                 :port1 = 65535, vlan_id1 = 1 cos1 = 0
                 :port2 = 65535, vlan_id2 = 501, cos2 = 0 state = 0
 DEBUG VLL LOCAL : MP_VLL_LOCAL_DY_SYNC: 0x029a6862 2 184
 DEBUG VLL LOCAL : state change: name = 10, vll_local_id = 10, state = 0 oldState
= 0
 DEBUG VLL LOCAL : MP_VLL_LOCAL_DY_SYNC: name = 10, vll_local_id = 10
                 :port1 = 65535, vlan_id1 = 1 cos1 = 0
                 :port2 = 65535, vlan_id2 = 501, cos2 = 0 state = 0
 DEBUG VLL LOCAL : MP_VLL_LOCAL_DY_SYNC: 0x029a6862 3 266

Brocade(config-mpls)#vll-local 10
 DEBUG VLL LOCAL : MP_VLL_LOCAL_DY_SYNC: name = 10, vll_local_id = 1
                 :port1 = 65535, vlan_id1 = 1 cos1 = 0
                 :port2 = 65535, vlan_id2 = 1, cos2 = 0 state = 0
 DEBUG VLL LOCAL : MP_VLL_LOCAL_DY_SYNC: 0x0299f862 1 102
 DEBUG VLL LOCAL : state change: name = 10, vll_local_id = 1, state = 0 oldState =
0

Brocade(config-mpls-vll-lo-10)# vlan 501
Brocade(config-mpls-vll-lo-10-vlan)#tag e 1/4
DEBUG VLL LOCAL : MP_VLL_LOCAL_DY_SYNC: name = 10, vll_local_id = 1
                 :port1 = 3, vlan_id1 = 501 cos1 = 0
                 :port2 = 65535, vlan_id2 = 1, cos2 = 0 state = 0
DEBUG VLL LOCAL : MP_VLL_LOCAL_DY_SYNC: 0x0299f862 2 184
DEBUG VLL LOCAL : state change: name = 10, vll_local_id = 1, state = 0 oldState =
0
```

```
Brocade(config-mpls-vll-lo-10-if-e-1/4)#tag e 2/4
DEBUG VLL LOCAL : MP_VLL_LOCAL_DY_SYNC: name = 10, vll_local_id = 1
                  :port1 = 3, vlan_id1 = 501 cos1 = 0
:port2 = 23, vlan_id2 = 501, cos2 = 0 state = 0
DEBUG VLL LOCAL : MP_VLL_LOCAL_DY_SYNC: 0x0298d062 1 102
DEBUG VLL LOCAL : state change: name = 10, vll_local_id = 1, state = 1 oldState =
0
```

MPLS dynamic bypass LSP

The dynamic bypass Label Switched Path (LSP) behavior is used to provide Fast Reroute (FRR) link or node protection to a facility-protected LSP at the Point of Local Repair (PLR). PLR tunnels the traffic on the protected LSP to the bypass LSP in case of failure. Multiple protected LSPs can use the same bypass LSP in case of protected LSP link or node failures.

MPLS dynamic bypass show commands

This section describes the show commands that display information about the MPLS dynamic bypass LSPs.

show mpls debug dynamic-bypass

Syntax: **show mpls debug dynamic-bypass** [**lsp** | **reset**]

- **lsp** - Displays information about all the dynamic bypass LSPs.
- **reset** - Resets the dynamic bypass debug counters.

This command enables debugging of MPLS dynamic bypass LSPs. Command output resembles the following example.

```
Brocade# show mpls debug dynamic-bypass
Dynamic Bypass: Enabled
 Enable on all interfaces: No
 Reoptimization time: 0 seconds.
 Maximum bypasses per merge point: 1000
 Maximum allowed dynamic bypasses: 1000
 Current number of dynamic bypasses 998
 Debug Print enabled = 0
  cspf_get_dbyp_fail = 0
  cspf_get_telink_fail = 0
  cspf_get_router_id_fail = 0
  cspf_dbyp_add_check_fail = 0
  cspf_cspf_fail = 0
  cspf_gen_name_fail = 0
  cspf_dbyp_create = 998
  cspf_set_new_lsp_fail = 0
  bsearch_not_static = 47296
  bsearch_not_dynamic = 5322
  bsearch_area_fail = 0
  bsearch_rid_fail = 0
  bsearch_no_bypif = 0
  bsearch_rldf_cac_fail = 1182
  bsearch_cac_hprio_fail = 0
  bsearch_cac_sprio_fail = 1230771
  bsearch_use_bypass_fail = 1
```

```
  bsearch_found_dbyp = 1473
  bsearch_found_sbyp = 2
  bsearch_no_bypass = 3849
  tmr_down_wait = 0
  tmr_nobkp_wait = 0
  tmr_dec_bkp_wait = 4
  tmr_up_remove = 998
  tmr_expired = 0
  tmr_dind_wait = 0
  tmr_insert_list = 0
  tmr_exp_delete = 0
  reop_delete_async = 0
  reop_not_ropble = 0
  reop_imp_cmt = 0
  reop_start_reop = 0
  rrt_no_reroute = 0
  clsp_updt_new = 0
  clsp_gbl_dflt = 2
  clsp_dbyif_dflt = 1007
  clsp_bref_add = 1000
  clsp_bref_delete = 1473
  cadd_no_dbyp = 0
  cadd_max_lsp_fail = 0
  cadd_tot_byp_fail = 1
  cadd_max_byp_fail = 0
  cadd_max_dbyp_fail = 0
  cadd_if_nodbyp = 0
  cadd_reqbw_more = 0
  cadd_hpri_less = 0
  cadd_more_hop = 0
  cadd_max_mp = 0
  updt_mp_add = 998
  updt_mp_del = 0
  frr_info_alloc = 2471
  frr_info_free = 1473
  frr_info_copy = 2471
  frr_info_find = 2471
  frr_info_qrt_add = 2471
  frr_info_qrt_del = 1473
  frr_info_lsp_add = 2278
  frr_info_lsp_inc = 193
  frr_info_lsp_del = 1280
```

The **show mpls debug dynamic-bypass lsp** command displays information about all the dynamic bypass LSPs. Command output resembles the following example.

```
Brocade# show mpls debug dynamic-bypass lsp
LSP dbyp-10.1.1.1-1  Backup: 1/0, Delete Wait: 0 Reop: 0/0
LSP dbyp-10.1.1.1-10  Backup: 1/0, Delete Wait: 0 Reop: 0/0
LSP dbyp-10.1.1.1-100  Backup: 1/0, Delete Wait: 0 Reop: 0/0
LSP dbyp-10.1.1.1-101  Backup: 1/0, Delete Wait: 0 Reop: 0/0
LSP dbyp-10.1.1.1-102  Backup: 1/0, Delete Wait: 0 Reop: 0/0
LSP dbyp-10.1.1.1-103  Backup: 1/0, Delete Wait: 0 Reop: 0/0
LSP dbyp-10.1.1.1-104  Backup: 1/0, Delete Wait: 0 Reop: 0/0
LSP dbyp-10.1.1.1-105  Backup: 1/0, Delete Wait: 0 Reop: 0/0
LSP dbyp-10.1.1.1-106  Backup: 1/0, Delete Wait: 0 Reop: 0/0
LSP dbyp-10.1.1.1-107  Backup: 1/0, Delete Wait: 0 Reop: 0/0
LSP dbyp-10.1.1.1-108  Backup: 1/0, Delete Wait: 0 Reop: 0/0
LSP dbyp-10.1.1.1-109  Backup: 1/0, Delete Wait: 0 Reop: 0/0
LSP dbyp-10.1.1.1-11  Backup: 1/0, Delete Wait: 0 Reop: 0/0
LSP dbyp-10.1.1.1-110  Backup: 1/0, Delete Wait: 0 Reop: 0/0
```

```
LSP dbyp-10.1.1.1-111  Backup: 1/0, Delete Wait: 0 Reop: 0/0
LSP dbyp-10.1.1.1-112  Backup: 1/0, Delete Wait: 0 Reop: 0/0
LSP dbyp-10.1.1.1-113  Backup: 1/0, Delete Wait: 0 Reop: 0/0
LSP dbyp-10.1.1.1-114  Backup: 1/0, Delete Wait: 0 Reop: 0/0
LSP dbyp-10.1.1.1-115  Backup: 1/0, Delete Wait: 0 Reop: 0/0
LSP dbyp-10.1.1.1-116  Backup: 1/0, Delete Wait: 0 Reop: 0/0
LSP dbyp-10.1.1.1-117  Backup: 1/0, Delete Wait: 0 Reop: 0/0
LSP dbyp-10.1.1.1-118  Backup: 1/0, Delete Wait: 0 Reop: 0/0
LSP dbyp-10.1.1.1-119  Backup: 1/0, Delete Wait: 0 Reop: 0/0
LSP dbyp-10.1.1.1-12  Backup: 1/0, Delete Wait: 0 Reop: 0/0
LSP dbyp-10.1.1.1-120  Backup: 1/0, Delete Wait: 0 Reop: 0/0
LSP dbyp-10.1.1.1-121  Backup: 1/0, Delete Wait: 0 Reop: 0/0
```

The **show mpls debug dynamic-bypass reset** command resets the dynamic bypass debug counters. Command output resembles the following example.

```
Brocade# show mpls debug dynamic-bypass reset
Dynamic Bypass: Enabled
 Enable on all interfaces: No
 Reoptimization time: 0 seconds.
 Maximum bypasses per merge point: 1000
 Maximum allowed dynamic bypasses: 1000
 Current number of dynamic bypasses 998
 Debug Print enabled = 0
  cspf_get_dbyp_fail = 0
  cspf_get_telink_fail = 0
  cspf_get_router_id_fail = 0
  cspf_dbyp_add_check_fail = 0
  cspf_cspf_fail = 0
  cspf_gen_name_fail = 0
  cspf_dbyp_create = 998
  cspf_set_new_lsp_fail = 0
  bsearch_not_static = 47296
  bsearch_not_dynamic = 5322
  bsearch_area_fail = 0
  bsearch_rid_fail = 0
  bsearch_no_bypif = 0
  bsearch_rldf_cac_fail = 1182
  bsearch_cac_hprio_fail = 0
  bsearch_cac_sprio_fail = 1230771
  bsearch_use_bypass_fail = 1
  bsearch_found_dbyp = 1473
  bsearch_found_sbyp = 2
  bsearch_no_bypass = 3849
  tmr_down_wait = 0
  tmr_nobkp_wait = 0
  tmr_dec_bkp_wait = 4
  tmr_up_remove = 998
  tmr_expired = 0
  tmr_dind_wait = 0
  tmr_insert_list = 0
  tmr_exp_delete = 0
  reop_delete_async = 0
  reop_not_ropble = 0
  reop_imp_cmt = 0
  reop_start_reop = 0
  rrt_no_reroute = 0
  clsp_updt_new = 0
  clsp_gbl_dflt = 2
  clsp_dbyif_dflt = 1007
  clsp_bref_add = 1000
```

```
  clsp_bref_delete = 1473
  cadd_no_dbyp = 0
  cadd_max_lsp_fail = 0
  cadd_tot_byp_fail = 1
  cadd_max_byp_fail = 0
  cadd_max_dbyp_fail = 0
  cadd_if_nodbyp = 0
  cadd_reqbw_more = 0
  cadd_hpri_less = 0
  cadd_more_hop = 0
  cadd_max_mp = 0
  updt_mp_add = 998
  updt_mp_del = 0
  frr_info_alloc = 2471
  frr_info_free = 1473
  frr_info_copy = 2471
  frr_info_find = 2471
  frr_info_qrt_add = 2471
  frr_info_qrt_del = 1473
  frr_info_lsp_add = 2278
  frr_info_lsp_inc = 193
  frr_info_lsp_del = 1280
Resetting Mpls dynamic bypass debug counters
```

MPLS dynamic bypass debug commands

This section describes the debug command that enables dynamic bypass console logs.

debug mpls dynamic-bypass all

Syntax: [**no**] **debug mpls dynamic-bypass all**

This command enables and displays the dynamic bypass-related logs on the console. Command output resembles the following example.

```
Brocade# debug mpls dynamic-bypass all
MPLS: dynamic-bypass debugging is on
```

NOTE
The **debug mpls all** or **debug all** command do not enable dynamic bypass debugs; however, the **no debug mpls all** or **no debug all** commands disable the dynamic bypass debugs if dynamic bypass debugging is already enabled.

MPLS LDP

Brocade devices support the Label Distribution Protocol (LDP) for setting up non-traffic-engineered tunnel LSPs in an MPLS network.

When used to create tunnel LSPs, LDP allows a set of destination IP prefixes (known as a Forwarding Equivalence Class (FEC)) to be associated with an LSP. Each label switch router (LSR) establishes a peer relationship with its neighboring LDP-enabled routers and exchanges label mapping information, which is stored in an LDP database.

The result of an LDP configuration is a full mesh of LSPs in an MPLS network, with each LDP-enabled router a potential ingress, transit, or egress LSR, depending on the destination.

MPLS LDP show commands

You can display the following information about LDP:

- The LDP version number, as well as the LSR's LDP identifier and loopback number
- Information about active LDP-created LSPs on the device
- Information about LDP-created tunnel LSPs for which this device is the ingress LER
- The contents of the LDP database
- Information about the LDP session between this LSR and its LDP peers
- Information about the connection between this LSR and its LDP peers
- Information about LDP-enabled interfaces on the LSR

show mpls ldp

Syntax: show mpls ldp

To display the LDP version number, the LSR's LDP identifier and loopback number, and the LDP hello interval and hold time, enter the **show mpls ldp** command.

```
Brocade# show mpls ldp
Label Distribution Protocol version 1
  LSR ID: 10.15.1.15, using Loopback 1 (deleting it will stop LDP)
  Hello interval: Link 5 sec, Targeted 15 sec
  Hold time value sent in Hellos: Link 15 sec, Targeted 45 sec
  Keepalive interval: 6 sec, Hold time multiple: 6 intervals
  Keepalive timeout: 36 sec
  Load sharing: 1
  Tunnel metric: 0
  FEC used for auto discovered peers: current 129, configured 129
  Label Withdrawal Delay: 30s
```

show mpls ldp database

Syntax: show mpls ldp database

This command displays the LSR LDP Label Information Base, which contains all the labels that have been learned from each LSR peer, as well as all of the labels that has been sent to its LDP peers. Command output resembles the following example.

```
Brocade# show mpls ldp database
Session 10.1.1.1:0 - 10.2.2.2:0
 Downstream label database:
   Label     Prefix              State
   3         10.2.2.2/32          Installed
   1104      10.3.3.3/32          Retained
   1106      10.14.14.14/32      Retained
   1107      10.44.44.44/32      Retained
   800005    VC-FEC              Installed
 Upstream label database:
   Label     Prefix
   3         10.1.1.1/32
   1024      10.2.2.2/32
   1026      10.3.3.3/32
   1028      10.14.14.14/32
   1029      10.44.44.44/32
   800005    VC-FEC
   800006    VC-FEC
```

show mpls ldp fec prefix

Syntax: **show mpls ldp fec prefix** [*ip_address*]

This command displays the host address and prefix FECs from the LDP FEC database. Command output resembles the following example.

```
Brocade# show mpls ldp fec prefix
Total number of prefix FECs: 31
Total number of prefix FECs with LWD timer running: 10
Destination       State       Out-intf         Next-hop        Ingress Egress
10.11.11.11/32    current     --               --              No      No
10.42.42.29/32    retained    --               --              No      No
```

The following is the sample output from the **show mpls ldp fec prefix** command if IP address is specified.

```
Brocade# show mpls ldp fec prefix 10.16.16.16
FEC_CB: 0x2d9029d4, idx: 6, type: 2, pend_notif: None
 State: current, Ingr: Yes, Egr: No, UM Dist. done: No
 Prefix: 10.16.16.16/32
 Label Withdrawal Delay: Active: Time Remaining: 10s
 next_hop: 10.56.1.16, out_if: e1/1

 Downstream mappings:
(*) indicates delayed processing for this DM
 Local LDP ID        Peer LDP ID         Label State             CB
 10.15.1.15:0        10.16.16.16:0       3     Installed*       0x2d8ba878

 Upstream mappings:
 Local LDP ID        Peer LDP ID         Label      CB
 10.15.1.15:0        10.14.14.14:0       100        0x2d8ba878
```

show mpls ldp fec summary

Syntax: **show mpls ldp fec summary**

This command displays LDP FEC summary information. Command output resembles the following example.

```
Brocade# show mpls ldp fec summary
LDP FEC summary:
  Total number of prefix FECs: 1
  Total number of prefix FECs installed: 0
  Total number of prefix FECs filtered: 0
  Total number of VC-FEC type 128: 0
  Total number of VC-FEC type 129: 0
  Total number of VC FECs installed: 0

LDP error statistics:
  Total number of route update processing errors: 0
  Total number of VC FEC processing errors: 0
```

show mpls ldp fec vc

Syntax: **show mpls ldp fec vc** *vc-id*

The *vc-id* variable specifies the ID of the Virtual Circuit (VC).

This command displays a list of VC FECs from the LDP FEC database. Command output resembles the following example.

```
Brocade# show mpls ldp fec vc
Total number of VC FECs: 2
Total number of VC FECs Installed: 2
Peer LDP ID       State      VC-ID  VC-Type  FEC-Type  Ingress   Egress
10.125.125.1:0    current    100    4        128       Yes       Yes
10.125.125.1:0    current    1000   5        128       Yes       Yes
```

To display detailed information about a specific VC, enter the **show mpls ldp fec vc** command with a VC ID, as shown in the following example.

```
Brocade# show mpls ldp fec vc 100
FEC_CB: 0x29391510, idx: 6, type: 128, pend_notif: None
State: current, Ingr: Yes, Egr: Yes, UM Dist. done: Yes
VC-Id: 100, vc-type: 4, grp-id: 0
Local-mtu: 1500, remote-mtu: 1500, MTU enforcement: enabled

Downstream mappings:
Local LDP ID       Peer LDP ID       Label     State      CB
10.128.128.28:0    10.125.125.1:0    800000    Installed  0x29391328(-1)

Upstream mappings:
Local LDP ID       Peer LDP ID       Label     CB
10.128.128.28:0    10.125.125.1:0    800003    0x2939141c(-1)
```

show mpls ldp interface

Syntax: show mpls ldp interface

This command displays information about LDP-enabled interfaces on the LSR. Command output resembles the following example.

```
Brocade# show mpls ldp interface
              Label-space   Nbr    Hello      Next
Interface     ID            Count  Interval   Hello
e1/1          0             1      5          0 sec
gre_tnl200    0             1      5          2 sec
(targeted)    0             0      0           --
```

show mpls ldp neighbor

Syntax: show mpls ldp neighbor [**detail** | *peer ip-addr* [*label space id*]]

- **detail** - Displays detailed information about the connection between the LSR and its LDP-enabled neighbors.
- *peer ip-addr* - Specifies the LDP ID of the neighbor.
- *label space id* - Specifies the label space ID of the peer.

This command displays information about the connection between the LSR and its LDP-enabled neighbors. Command output resembles the following example.

```
Brocade# show mpls ldp neighbor
Nbr Transport     Interface     Nbr LDP ID    Max Hold   Time Left
10.1.1.1          e1/1          10.1.1.1:0    15         12
10.3.3.3          gre-tnl200    10.3.3.3:0    15         12
```

To display detailed information about the connection between the LSR and its LDP-enabled neighbors, enter the **show mpls ldp neighbor detail** command.

```
Brocade# show mpls ldp neighbor detail
Number of link neighbors: 2
 Number of targeted neighbors: 1
Nbr Transport Addr: 10.42.42.42, Interface: e1/5, Nbr LDP ID: 10.42.42.42:0
 MaxHold: 13 sec, Time Left: 9 sec, Up Time: 9 hr 22 min 50 sec
 Configured Hold time: 13 sec, Neighbor Proposed Hold Time: 15 sec
```

show mpls ldp path

Syntax: **show mpls ldp path**

Use the **show mpls ldp path** command to display information about active LDP-created LSPs for which the device is an ingress, transit, or egress LSR. Command output resembles the following example.

```
Brocade# show mpls ldp path
Destination route    Upstr-session(label)     Downstr-session(label, intf)
10.2.2.2/32          10.1.1.1:0(3)
10.1.1.1/32                                   10.1.1.1:0(3, e1/1)
                                              10.1.1.1:0(3, e1/2)
10.3.3.3/32          10.3.3.3:0(5050)         10.4.4.4:0(2057,gre-tnl200)
```

NOTE
In this context, “upstream” and “downstream” refer to the direction that data traffic flows in an LSP. This is the opposite of the direction that labels are distributed using LDP.

show mpls ldp peer

Syntax: **show mpls ldp peer** [**detail**]

This command displays LDP peer information. Command output resembles the following example.

```
Brocade# show mpls ldp peer
Peer LDP ID        State
10.2.2.2:0          Operational
10.3.3.3:0          Operational
10.8.8.8:0          Operational
10.9.9.9:0          Unknown
10.14.14.14:0      Operational
```

To display more detailed information about the LDP peers, enter the **show mpls ldp peer detail** command, as shown in the following example.

```
Brocade# show mpls ldp peer detail
Peer LDP ID: 10.2.2.2:0, Local LDP ID: 10.1.1.1:0, State: Operational
Session Status UP, Entity Idx: 4, Targeted: No, Target Adj Added: Yes
Num VLL: 2, Num VPLS: 0
Rcvd VC-FECs:
 From 10.2.2.2: Label: 800001, VC Id: 120, Grp_Id: 0, VC Type: 4
Peer LDP ID: 10.8.8.8:0, Local LDP ID: 10.1.1.1:0, State: Operational
Session Status UP, Entity Idx: 2, Targeted: Yes, Target Adj Added: Yes
Num VLL: 2, Num VPLS: 0
Rcvd VC-FECs:
 From 10.8.8.8: Label: 16, VC Id: 19, Grp_Id: 0, VC Type: 32773
 From 10.8.8.8: Label: 18, VC Id: 18, Grp_Id: 0, VC Type: 32772
```

show mpls ldp session

Syntax: **show mpls ldp session** [**detail** | *a.b.c.d* **filtered** [**in** | **out**]]

- **detail** - Displays detailed information about the LDP sessions.
- **filtered** - Displays a list of filtered upstream and downstream mappings for the session. You can use the **in** and **out** options with the **filtered** option to display only the filtered downstream or filtered upstream mappings, respectively.

To display detailed information about the LDP sessions, enter the **show mpls ldp session detail** command.

```
Brocade# show mpls ldp session detail
Number of link LDP sessions: 1
 Number of Operational link LDP sessions: 1
 Number of targeted LDP sessions: 0
 Number of Operational targeted LDP sessions: 0

Peer LDP ID: 10.12.12.12:0, Local LDP ID: 10.14.14.14:0, State: Operational
  Adj: Link, Role: Active, Next keepalive: 4 sec, Hold time left: 39 sec
  Keepalive interval: 5 sec, Max hold time: 40 sec
  Configured keepalive timeout: 40 sec
  Peer proposed keepalive timeout: 49 sec
  Up time: 17 hr 21 min 23 sec
  Neighboring interfaces: e2/1
  TCP connection: 10.14.14.14:9005--10.12.12.12:646, State: ESTABLISHED
  Number of FECs installed from peer: 7
  Number of FECs filtered  for peer (in/out): 0/0
  Next-hop addresses received from the peer:
    10.12.12.12  10.21.21.12  10.23.23.12  10.29.29.12  192.168.1.1
    192.168.1.2  192.168.1.3  192.168.1.4  192.168.1.5
  IGP Sync:
    Unrecognized Notification Capability: Local: Off, Remote: Off
    Local State: In-sync, RemoteState: -
    Receive label silence time: 1000 ms, Timer not running
  Graceful restart: disabled
  Number of FECs Received from peer: 7
  Number of FECs installed from peer: 7
  Number of FECs filtered for peer(in/out): 0/3
 Outbound FEC filtering prefix-list: list-out
```

The following is the sample output from the **show mpls ldp session** *a.b.c.d* **filtered** command.

```
Brocade# show mpls ldp session 10.12.12.12  filtered
 Number of link LDP sessions: 1
 Number of Operational link LDP sessions: 1
 Number of targeted LDP sessions: 0
 Number of Operational targeted LDP sessions: 0

Peer LDP ID: 10.12.12.12:0, Local LDP ID: 10.14.14.14:0, State: Operational
  Adj: Link, Role: Active, Next keepalive: 4 sec, Hold time left: 39 sec
  Keepalive interval: 5 sec, Max hold time: 40 sec
  Configured keepalive timeout: 40 sec
  Peer proposed keepalive timeout: 49 sec
  Up time: 17 hr 21 min 23 sec
  Neighboring interfaces: e2/1
  TCP connection: 10.14.14.14:9005--10.12.12.12:646, State: ESTABLISHED
  Next-hop addresses received from the peer:
    10.12.12.12  10.21.21.12  10.23.23.12  10.29.29.12  192.168.1.1
    192.168.1.2  192.168.1.3  192.168.1.4  192.168.1.5
  IGP Sync:
    Unrecognized Notification Capability: Local: Off, Remote: Off
    Local State: In-sync, RemoteState: -
    Receive label silence time: 1000 ms, Timer not running
```

```
  Graceful restart: disabled
  Number of FECs Received from peer: 7
  Number of FECs installed from peer: 2
  Number of FECs filtered for peer(in/out): 5/3
 Outbound FEC filtering prefix-list: list-out
FECs received from peer and filtered inbound
   3           192.168.1.6/32
   3           192.168.1.7/32
   3           192.168.1.8/32
   3           192.168.1.9/32
   3           192.168.1.10/32
FECs not advertised to peer because of outbound filter
192.168.1.3/32
192.168.1.4/32
192.168.1.5/32
```

show mpls ldp tunnel

Syntax: **show mpls ldp tunnel** [**detail** | [**out-interface** *interface-name*]]

- **detail** - Displays detailed information about LDP-created LSPs.
- **out-interface** *interface-name* - Filters and displays the output to include only the LDP LSP tunnels whose out-interfaces match the out-interface specified using the *interface-name* variable.

This command filters and displays information about LDP-created LSPs based on the out-interface. Command output resembles the following example.

```
Brocade# show mpls ldp tunnel out-interface ve 15
Total LDP tunnels on interface: 3
                Oper      Tunnel    Outbound
To              State     Intf      Intf
10.130.130.3   UP        tnl7      ve15
10.130.130.4   UP        tnl11     ve15
10.130.130.5   UP        tnl12     ve15
```

MPLS LDP debug commands

This section describes the debug commands that generate MPLS LDP information.

debug mpls ldp

Syntax: [**no**] **debug mpls ldp** [**all** | **adjacency** | **error** | **event** | **fec** | **gr** | **packets** | **socket** | **state** | **tcpdump** | **tunnel**]

- **all** - Displays all messages related to a MPLS LDP module.
- **adjacency** - Displays debug messages related to MPLS LDP adjacency messages.
- **error** - Displays debug messages related to MPLS LDP errors.
- **event** - Displays debug messages related to MPLS LDP events.
- **fec** - Displays MPLS LDP FEC information.
- **gr** - Displays debug messages related to MPLS LDP Graceful Restart (GR).
- **packets** - Displays debug messages related to MPLS LDP packets.
- **socket** - Displays debug messages related to MPLS LDP sockets.
- **state** - Displays debug messages related to MPLS LDP states.

- **tcpdump** - Displays MPLS LDP packets as raw data.
- **tunnel** - Displays debug messages related to MPLS LDP LSP interaction with other modules as virtual interfaces.

This command displays MPLS LDP-related information, as shown in the following example.

```
Brocade# debug mpls ldp all
Dec 10 01:26:34 LDP_PKT: send targeted Hello to <10.7.7.3, 646>
Dec 10 01:26:34 LDP_PKT: send targeted Hello to <10.7.7.1, 646>
Warning: LDP session 10.7.7.1 doesn't exist!
telnet@PE2#Dec 10 01:26:36 LDP_PKT: receive link Hello from 10.1.1.1 on e2/1
Dec 10 01:26:37 LDP_TCPDUMP: rx from 10.7.7.3, TCP length 18
 0001000e 07070703 00000201 00040000 142e
Dec 10 01:26:37 LDP_PKT: receive targeted Hello from 10.7.7.1 on e2/1
Dec 10 01:26:37 LDP_ADJ: Targeted adajacency to 10.7.7.1:0, entity idx 4 is added
Dec 10 01:26:37 LDP_EVT: LDP session to 10.7.7.1 is initiated, targeted adjacency
Yes
Dec 10 01:26:37 LDP SC 1 Initialization FSM, Session CB 159/13, input 0, old state
0, new state 1, action 1.
Dec 10 01:26:37 LDP SC 1 SM Socket FSM, CB 124/9, input 0, old state 0, new state
1, action 1, Session CB 159/13,
local LDP ID 10.7.7.2:0, peer LDP ID 10.7.7.1:0.
Dec 10 01:26:37 LDP_SCK: receive TCP socket request
Dec 10 01:26:37 LDP_SCK: try connecting 10.7.7.1, local port 9003
Dec 10 01:26:37 LDP_SCK: start connecting 10.7.7.1 local port 9003, TCB 0X117EF03E
Dec 10 01:26:37 LDP_SCK: connection accepted by peer 10.7.7.1, ready to send TCB
0X117EF03E
Dec 10 01:26:37 LDP SC 1 SM Socket FSM, CB 124/9, input 1, old state 1, new state
2, action 2, Session CB 159/13,
local LDP ID 10.7.7.2:0, peer LDP ID 10.7.7.1:0.
Dec 10 01:26:37 LDP SC 1 Initialization FSM, Session CB 159/13, input 1, old state
1, new state 2, action 2.
Dec 10 01:26:37 LDP SC 1 Initialization FSM, Session CB 159/13, input 4, old state
2, new state 3, action 4.
Dec 10 01:26:37 LDP_GR: PM add session to 10.7.7.1:2, gr 0, reconnect 0, recovery
0
Dec 10 01:26:37 LDP SC 1 Initialization FSM, Session CB 159/13, input 5, old state
3, new state 4, action 5.
Dec 10 01:26:37 LDP SC 1 Initialization FSM, Session CB 159/13, input 7, old state
4, new state 5, action 6.
Dec 10 01:26:37 LDP_PKT: Rcvd PDU <- 10.7.7.1, ver 1, pdu len 32, id 10.7.7.2:0
(tcb a00014a9)
Dec 10 01:26:37  Msg: Initialize(0x0200) len 22, id 0x00000001
Dec 10 01:26:37 LDP_TCPDUMP: tx to 10.7.7.1, TCP length 36
 00010020 07070702 00000200 00160000 00010500 000e0001 00240000 10000707
 07010000
Dec 10 01:26:37 LDP_TCPDUMP: rx from 10.7.7.1, TCP length 36
 00010020 07070701 00000200 00160000 00010500 000e0001 00240000 05a00707
 07020000
Dec 10 01:26:37 LDP SC 1 Initialization FSM, Session CB 159/13, input 9, old state
5, new state 6, action 8.
Dec 10 01:26:37 LDP SC 1 Initialization FSM, Session CB 159/13, input 10, old
state 6, new state 8, action 9.
Dec 10 01:26:37 LDP_PKT: Rcvd PDU <- 10.7.7.1, ver 1, pdu len 14, id 10.7.7.2:0
(tcb a00014a9)
Dec 10 01:26:37  Msg: Keepalive(0x0201) len 4, id 0x00000003
Dec 10 01:26:37 LDP_TCPDUMP: tx to 10.7.7.1, TCP length 18
 0001000e 07070702 00000201 00040000 0003
Dec 10 01:26:37 LDP_TCPDUMP: rx from 10.7.7.1, TCP length 18
 0001000e 07070701 00000201 00040000 0002
```

```
Dec 10 01:26:37 LDP SC 1 Initialization FSM, Session CB 159/13, input 11, old
state 8, new state 9, action 11.
Dec 10 01:26:37 LDP_EVT: LDP session to 10.7.7.1, entity idx 4, type targeted goes
UP, use TCB 0XA00014A9:0X117EF03E
Dec 10 01:26:37 LDP_PKT: Rcvd PDU <- 10.7.7.1, ver 1, pdu len 40, id 10.7.7.2:0
(tcb a00014a9)
Dec 10 01:26:37  Msg: Address(0x0300) len 30, id 0x80000000
Dec 10 01:26:37 LDP_TCPDUMP: tx to 10.7.7.1, TCP length 44
 00010028 07070702 00000300 001e8000 00000101 00160001 07070702 0e010203
 11010101 11011102 16010102
Dec 10 01:26:37 LDP SC 1 Initialization FSM, Session CB 159/13, input 16, old
state 9, new state 10, action 24.
Dec 10 01:26:37 LDP SC 1 Adjacency FSM, CB 1/41, input 1, old state 1, new state
2, action 1, Session CB 159/13.
Dec 10 01:26:37 LDP SC 1 Adjacency FSM, CB 1/41, input 3, old state 2, new state
4, action 0, Session CB 159/13.
Dec 10 01:26:37 LDP_TCPDUMP: rx from 10.7.7.1, TCP length 56
 00010034 07070701 00000300 002a8000 00000101 00220001 01020304 03030b02
 03060702 03080302 07070701 100c0301 11011101 c0a81502
Dec 10 01:26:37 LDP PM 1 Egress FSM, CB 52/5, input 0, old state 0, new state 0,
action 1, FEC CB 159/1,
FEC Prefix: 10.3.11.0, len 24.
Dec 10 01:26:37 LDP PM 1 Egress FSM, CB 52/5, input 3, old state 0, new state 0,
action 0, FEC CB 159/1,
FEC Prefix: 10.3.11.0, len 24.
Dec 10 01:26:37 LDP PM 1 Egress FSM, CB 139/17, input 0, old state 0, new state 0,
action 1, FEC CB 175/9,
FEC Prefix: 10.6.7.0, len 24.
Dec 10 01:26:37 LDP PM 1 Egress FSM, CB 139/17, input 3, old state 0, new state 0,
action 0, FEC CB 175/9,
FEC Prefix: 10.6.7.0, len 24.
Dec 10 01:26:37 LDP PM 1 Egress FSM, CB 133/5, input 0, old state 0, new state 0,
action 1, FEC CB 183/13,
FEC Prefix: 10.8.3.0, len 24.
Dec 10 01:26:37 LDP PM 1 Egress FSM, CB 133/5, input 3, old state 0, new state 0,
action 0, FEC CB 183/13,
FEC Prefix: 10.8.3.0, len 24.
Dec 10 01:26:37 LDP PM 1 Egress FSM, CB 126/41, input 0, old state 0, new state 0,
action 1, FEC CB 207/25,
FEC Prefix: 10.7.7.1, len 32.
Dec 10 01:26:37 LDP PM 1 Egress FSM, CB 126/41, input 3, old state 0, new state 0,
action 0, FEC CB 207/25,
FEC Prefix: 10.7.7.1, len 32.
Dec 10 01:26:37 LDP PM 1 Egress FSM, CB 114/17, input 0, old state 0, new state 0,
action 1, FEC CB 215/29,
FEC Prefix: 10.12.3.0, len 24.
Dec 10 01:26:37 LDP PM 1 Egress FSM, CB 114/17, input 3, old state 0, new state 0,
action 0, FEC CB 215/29,
FEC Prefix: 10.12.3.0, len 24.
Dec 10 01:26:37 LDP_PKT: Rcvd PDU <- 10.7.7.3, ver 1, pdu len 14, id 10.7.7.2:0
(tcb a0000028)
Dec 10 01:26:37  Msg: Keepalive(0x0201) len 4, id 0x0000142d
Dec 10 01:26:37 LDP_TCPDUMP: tx to 10.7.7.3, TCP length 18
 0001000e 07070702 00000201 00040000 142d
Dec 10 01:26:37 LDP_TCPDUMP: rx from 10.7.7.6, TCP length 18
 0001000e 07070706 00000201 00040000 142f
```

debug mpls ldp adjacency

Syntax: [no] debug mpls ldp adjacency

This command displays information about MPLS LDP adjacencies.

```
Brocade# debug mpls ldp adjacency
LDP_ADJ: Link adjacency to 10.140.140.4:0 on interface e1/2 is deleted, reason 4
LDP_ADJ: Link adjacency to 10.140.140.4:0 on interface e1/2 is added
```

debug mpls ldp error

Syntax: [no] debug mpls ldp error

This command displays the errors detected by LDP components such as LDP session keepalive timer expiration, hello adjacency timeout, and so on.

```
Brocade# debug mpls ldp error
LDP_ERR: Session KeepAlive timer to peer 10.120.120.2 has expired
```

debug mpls ldp event

Syntax: [no] debug mpls ldp event

This command displays session up and down events, similar to the following example.

```
Brocade# debug mpls ldp event
LDP_EVT: LDP session to 10.140.140.4, entity idx 1, type non-targeted goes DOWN,
use TCB 0XA0000010:0X12BF2B70
LDP_EVT: remove session to 10.140.140.4. Session is deleted
LDP_EVT: initiate LDP session to peer 10.140.140.4. Session is not found
LDP_EVT: initiate session block 0X344BF140, entity idx 1 and insert to tree
LDP_EVT: LDP session to 10.140.140.4 is initiated, targeted adjacency No
LDP_EVT: LDP session to 10.140.140.4, entity idx 1, type non-targeted goes UP, use
TCB 0XA0000012:0X12BF244A
```

debug mpls ldp fec

Syntax: [**no**] **debug mpls ldp fec** [**all** | **lsr-id** *ip_address label_space* | **key** [**prefix** *ip-address prefix-length* | **vc** *vc_id*]]

This command displays MPLS LDP FEC information.

- **all** - Displays all MPLS LDP FEC-related information.
- **lsr-id** *ip_address label_space* - Limits the MPLS LDP FEC information displayed to a specific LSR ID.
- **key** - Limits the information displayed to a specific FEC key value.
- **prefix** *ip-address prefix-length* - Limits the display of MPLS LDP FEC-related information to specific prefixes.
- **vc** *vc_id* - Displays MPLS LDP FEC-related information for a specific VC ID.

Command output resembles the following example.

```
Brocade# debug mpls ldp fec
LDP PM 1 Ingress FSM, CB 171/33, input 0, old state 0, new state 0, action 1, FEC
CB 169/33,
FEC Prefix: 10.100.100.100, len 32.
LDP PM 1 Egress FSM, CB 205/45, input 0, old state 0, new state
0, action 1, FEC CB 184/45,
```

```
FEC Prefix: 10.20.20.20, len 32.
LDP PM 1 UM FSM, CB 164/29, input 0, old state 0, new state 1, a
ction 1, UT CB 101/5, session CB 117/25, FEC CB 184/45.
FEC Prefix: 10.20.20.20, len 32.
LDP PM 1 UM FSM, CB 164/29, input 3, old state 1, new state 2, a
ction 23, UT CB 101/5, session CB 117/25, FEC CB 184/45.
FEC Prefix: 10.20.20.20, len 32.
LDP PM 1 UM FSM, CB 164/29, input 1, old state 2, new state 2, a
ction 2, UT CB 101/5, session CB 117/25, FEC CB 184/45.
FEC Prefix: 10.20.20.20, len 32.
```

debug mpls ldp gr

Syntax: [**no**] **debug mpls ldp gr**

This command displays debug messages related to MPLS LDP Graceful Restart (GR), including the internal GR FSM transitions.

Command output resembles the following example.

```
Brocade# debug mpls ldp gr
Oct 25 14:06:15 LDP_GR: wait for peer 10.210.210.21:0 to restart
Oct 25 14:06:30 LDP_GR: PM add session to 10.210.210.21:0, gr 1, reconnect 120000,
recovery 120000
Oct 25 14:06:30 LDP_GR: GR begins for peer 10.210.210.21:0
Oct 25 14:08:30 LDP_GR: GR completes successfully for peer 10.210.210.21:0
```

debug mpls ldp packets

Syntax: [**no**] **debug mpls ldp packets** [**all** | **detail** | **direction** | **lsr_id** | **pkt_type**]

This command displays MPLS LDP packet information, which is further filtered by direction, packet types, and LSR ID.

- **all** - Displays all MPLS LDP packets.
- **detail** - Displays messages about MPLS LDP packets in a detailed version.
- **direction** - Limits the display of MPLS LDP packets to specific directions (send and receive).
- **lsr_id** - Limits the display of MPLS LDP packets to a specific LSR ID.
- **pkt_type** - Limits the display of MPLS LDP packets to specific packet types.

Command output resembles the following example.

```
Brocade# debug mpls ldp packets
Msg: Keepalive(0x0201) len 4, id 0x00000131
LDP_PKT: send link Hello to e1/1
LDP_PKT: receive link Hello from 10.1.1.2 on e1/1
LDP_PKT: send link Hello to e1/1
LDP_PKT: Rcvd PDU <- 10.20.20.20, ver 1, pdu len 14, id 10.100.100.100:0 (tcb
0dfa0328)
Msg: Keepalive(0x0201) len 4, id 0x00000132
LDP_PKT: receive link Hello from 10.1.1.2 on e1/1
LDP_PKT: send link Hello to e1/1
LDP_PKT: receive link Hello from 10.1.1.2 on e1/1
LDP_PKT: Rcvd PDU <- 10.20.20.20, ver 1, pdu len 14, id 10.100.100.100:0 (tcb
0dfa0328)
Msg: Keepalive(0x0201) len 4, id 0x00000133
LDP_PKT: receive UDP packet for invalid destination address 10.1.1.1
```

debug mpls ldp packets pkt_type

Syntax: [**no**] **debug mpls ldp packets pkt_type** [**all** | **address** | **initialization** | **label** | **notification** | **hello** | **keepalive**]

This command displays MPLS LDP packets with specific LDP packet types.

- **all** - Displays MPLS LDP packets of all types.
- **address** - Displays information about MPLS LDP addresses, including address withdraw messages.
- **initialization** - Displays LDP Initialization messages.
- **label** - Displays LDP label mapping, withdraw, request, release, and abort messages.
- **notification** - Displays LDP notification information.
- **hello** - Displays information about the periodic link Hello messages sent and received.
- **keepalive** - Displays LDP Keepalive messages after the session comes up.

Command output resembles the following example.

```
Brocade# debug mpls ldp packets pkt_type all
Dec 10 01:28:01 LDP_PKT: receive link Hello from 10.1.1.1 on e2/1
Dec 10 01:28:01 LDP_PKT: Rcvd PDU <- 10.7.7.1, ver 1, pdu len 14, id 10.7.7.2:0
(tcb a00014a9)
Dec 10 01:28:01  Msg: Keepalive(0x0201) len 4, id 0x00000011
Dec 10 01:28:01 LDP_PKT: Rcvd PDU <- 10.7.7.3, ver 1, pdu len 14, id 10.7.7.2:0
(tcb a0000028)
Dec 10 01:28:01  Msg: Keepalive(0x0201) len 4, id 0x0000143b
Dec 10 01:28:03 LDP_PKT: Rcvd PDU <- 10.7.7.6, ver 1, pdu len 14, id 10.7.7.2:0
(tcb a0000025)
Dec 10 01:28:03  Msg: Keepalive(0x0201) len 4, id 0x0000143c
Dec 10 01:28:04 LDP_PKT: send link Hello to e2/1
Dec 10 01:28:04 LDP_PKT: send targeted Hello to <10.7.7.3, 646>
Dec 10 01:28:04 LDP_PKT: send targeted Hello to <10.7.7.1, 646>
Dec 10 01:28:06 LDP_PKT: receive link Hello from 10.1.1.1 on e2/1
Dec 10 01:28:07 LDP_PKT: Rcvd PDU <- 10.7.7.1, ver 1, pdu len 14, id 10.7.7.2:0
(tcb a00014a9)
Dec 10 01:28:07  Msg: Keepalive(0x0201) len 4, id 0x00000012
Dec 10 01:28:07 LDP_PKT: receive targeted Hello from 10.7.7.1 on e2/1
Dec 10 01:28:07 LDP_PKT: Rcvd PDU <- 10.7.7.3, ver 1, pdu len 14, id 10.7.7.2:0
(tcb a0000028)
Dec 10 01:28:07  Msg: Keepalive(0x0201) len 4, id 0x0000143c
Dec 10 01:28:09 LDP_PKT: send link Hello to e2/1
Dec 10 01:28:09 LDP_PKT: Rcvd PDU <- 10.7.7.6, ver 1, pdu len 14, id 10.7.7.2:0
(tcb a0000025)
Dec 10 01:28:09  Msg: Keepalive(0x0201) len 4, id 0x0000143d
no Dec 10 01:28:10 LDP_PKT: receive targeted Hello from 10.7.7.6 on e2/2
debug Dec 10 01:28:11 LDP_PKT: receive link Hello from 10.1.1.1 on e2/1
```

debug mpls ldp packets pkt_type address

Syntax: [**no**] **debug mpls ldp packets pkt_type address**

This command displays information about MPLS LDP addresses, including address withdraw messages.

```
Brocade# debug mpls ldp pkt_type address
LDP um 0d9098c8 fec 10.11.11.0 ses 10.22.22.1: inp 0, st 0 -> 1, act A, gen 0
LDP um 0d9098c8 fec 10.11.11.0 ses 10.22.22.1: inp 3, st 1 -> 2, act W, gen 0
LDP um 0d9098c8 fec 10.11.11.0 ses 10.22.22.1: inp 1, st 2 -> 2, act B, gen 0
```

```
LDP um 0d9098c8 fec 10.11.11.0 ses 10.22.22.1: inp 13, st 2 -> 5, act G, gen 0
LDP um 0d9098c8 fec 10.11.11.0 ses 10.22.22.1: inp 1, st 5 -> 6, act I, gen 0
LDP: Send PDU -> 10.22.22.1, ver 1, pdu len 32, id 10.11.11.1:0 (tcb 0000002f)
Msg: Address(0x0300) len 22, id 0x80000000
Tlv: Addr_lst(0x0101) len 14, fam 256
Addrs: 10.1.1.1 10.5.1.1 10.11.11.1
```

debug mpls ldp packets pkt_type initialization

Syntax: [no] **debug mpls ldp packets pkt_type initialization**

This command displays LDP Initialization messages, as shown in the following example.

```
Brocade# debug mpls ldp packets pkt_type initialization
Msg: Initialize(0x0200) len 22, id 0x00000001
LDP: Send PDU -> 10.5.1.1, ver 1, pdu len 32, id 10.4.1.1:0 (tcb f0020000)
Msg: Keepalive(0x0201) len 4, id 0x00000002
LDP: Send PDU -> 10.5.1.1, ver 1, pdu len 32, id 10.4.1.1.0 (tcbf00s0000)
Msg: Address(0x0300)len 22, id 0x80000000
LDP: Send PDU -> 10.5.1.1, ver 1, pdu len 33, id 10.4.1.1:0 (tcbf0020000)
Msg: LabelMap(0x0400) len 23, id 0x80000001
LDP: Send PDU -> 10.5.1.1, ver 1, pdu len 33, id 10.4.1.1:0 (tcbf0020000)
Msg: LabelMap(0x0400) len 23, id 0x80000004
```

debug mpls ldp packets pkt_type notification

Syntax: [no] **debug mpls ldp packets pkt_type notification**

This command displays LDP notification information, which is generated when an unexpected event occurs. In the following example, an MPLS interface has gone down and come back up.

```
Brocade# debug mpls ldp packets pkt_type notification
Msg: Notification(0x0001) len 18, id 0x00000056
LDP: Send PDU -> 10.5.1.1, ver 1, pdu len 28, id 10.4.1.1:0 (tcb f0030000)
Msg: Notification(0x0001) len 18, id 0x00000001
```

debug mpls ldp packets pkt_type hello

Syntax: [no] **debug mpls ldp packets pkt_type hello**

This command displays information about the periodic link Hello messages sent and received.

```
Brocade# debug mpls ldp packets pkt_type hello
LDP: Send link Hello to e1/1
LDP: Send link Hello to e1/2
LDP: Rcvd link Hello from 10.1.1.2 on e1/1
LDP: Rcvd link Hello from 10.5.1.2 on e1/2
```

debug mpls ldp packets direction

Syntax: [no] **debug mpls ldp packets direction** [**send** | **receive**]

- **send** - Displays information about the sent LDP packets.
- **receive** - Displays information about the received LDP packets.

This command limits the display of MPLS LDP packets to the specified direction.

debug mpls ldp packets lsr_id

Syntax: [no] **debug mpls ldp packets lsr-id** *ip_address label_space_id*

- *ip_address* - Specifies the IP address.
- *label_space_id* - Specifies the label ID.

This command displays MPLS LDP packets for the specified LDP LSR ID.

debug mpls ldp socket

Syntax: [no] debug mpls ldp socket

This command displays information about LDP sockets. Command output resembles the following example.

```
Brocade# debug mpls ldp socket
LDP_SCK: close UDP link tx socket on interface e1/2
LDP_SCK: close UDP link rx socket on e1/2, ucb 0XFF010000#1, appl_sock 0X34B8A280
LDP_SCK: start closing TCP session<10.130.130.3,646-10.140.140.4,9006>, TCB
0X12BF244A
LDP_SCK: start closing TCP listen socket TCB 0X230015D5
LDP_SCK: TCP start listen for <10.130.130.3, 646>, TCB 0X12BF2726
LDP_SCK: open link hello tx socket on interface e1/2
LDP_SCK: complete link hello tx socket open ucb 0XFF010000, count1, appl_sock
0X34B8A380
LDP_SCK: open link hello rx socket on interface e1/2
LDP_SCK: complete link hello tx socket open ucb 0XFF010000, count1, appl_sock
0X34B8A380
LDP_SCK: open link hello rx socket on interface e1/2
LDP_SCK: receive incoming connection from <10.140.140.4, 9007>, listen TCB
0X12BF2726, TCB 0X12BF2B70
LDP_SCK: accept incoming connection from <10.140.140.4, 9007>, TCP handle
0X12BF2B70, pending_data No
LDP_SCK: connection established with <10.140.140.4, 9007>, TCB 0X12BF2B70
```

debug mpls ldp state

Syntax: [no] debug mpls ldp state

This command displays the MPLS LDP states. The following states are possible:

- 0 - Unknown
- 1 - No route
- 2 - Route found
- 3 - Path sent
- 4 - Path error

Command output resembles the following example.

```
Brocade# debug mpls ldp state
LDP SC 1 Initialization FSM, Session CB 207/33, input 3, old state 0, new state 2,
action 3.
LDP SC 1 Initialization FSM, Session CB 207/33, input 4, old state 2, new state 3,
action 4.
LDP SC 1 Initialization FSM, Session CB 207/33, input 5, old state 3, new state 4,
action 5.
LDP SC 1 Initialization FSM, Session CB 207/33, input 8, old state 4, new state 8,
action 7.
LDP SC 1 Initialization FSM, Session CB 207/33, input 11, old state 8, new state
9, action 11.
LDP SC 1 Initialization FSM, Session CB 207/33, input 16, old state 9, new state
10, action 24.
```

debug mpls ldp tcpdump

Syntax: [**no**] **debug mpls ldp tcpdump**

This command displays information about LDP dumps. Command output resembles the following example.

```
Brocade# debug mpls ldp tcpdump
LDP_TCPDUMP: tx to 10.140.140.4, TCP length 18 0001000e 82828203 00000201 00040000
000a
LDP_TCPDUMP: rx from 10.140.140.4, TCP length 18 0001000e 8c8c8c04 00000201
00040000 000b
```

debug mpls ldp tunnel

Syntax: [**no**] **debug mpls ldp tunnel** [**all** | **prefix** *ip-address prefix-length*]

This command displays MPLS LDP tunnel-related information.

- **all** - Displays all MPLS LDP tunnel up and down events.
- **prefix** *ip-address prefix-length* - Limits the display of information to specific prefixes.

Command output resembles the following example.

```
Brocade# debug mpls ldp tunnel
LDP_TUNNEL: tunnel(5) to 10.100.100.100 is up
```

MPLS VPLS

Virtual Private LAN Service (VPLS) is a method for carrying Layer 2 frames between Customer Edge (CE) devices across an MPLS domain. The Brocade implementation supports VPLS as described in the IETF Internet Draft, “draft-ietf-l2vpn-vpls-ldp-05.txt”.

VPLS CPU protection shields the management module CPU from being overwhelmed by VPLS traffic that requires CPU processing, such as source-MAC learning, or forwarding of unknown-unicast or broadcast traffic.

MPLS VPLS show commands

This section describes the show commands that display MPLS VPLS information.

show mpls vpls

Syntax: **show mpls vpls**

This command displays information about VPLS instances configured on the device.

```
Brocade# show mpls vpls
                      Num      Num      Ports      Num      Peers      Num
Name         Id       Vlans    Ports    Up         Peers    Up         VC-label
test         100      2        3        0          2        0          32
```

show mpls vpls summary

Syntax: **show mpls vpls summary**

This command displays a summary of VPLS statistics, including the number of VPLS instances, number of VPLS peers, label range size, and maximum size of the VPLS MAC database.

```
Brocade# show mpls vpls summary
Total VPLS configured: 3, maximum number of VPLS allowed: 4096
Total VPLS peers configured: 1, total peers operational: 1
Maximum VPLS macentries allowed: 8192, currently installed: 3
VPLS global raw mode VC-Type is Ethernet (0x5)
VPLS global MTU is 1500, MTU enforcement is OFF
Global CPU protection: OFF
MVIDs in use: 1 of 1 total allocated
```

show mpls vpls detail

Syntax: show mpls vpls detail

This command displays more detailed information about each VPLS instance.

```
Brocade# show mpls vpls detail
VPLS 1, Id 1, Max mac entries: 8192
 PBB
 Total vlans: 2, Tagged ports: 2 (2 Up), Untagged ports 0 (0 Up)
 IFL-ID: 4096
  Vlan 300
   Tagged: ethe 1/5
  Vlan 3000 isid 40000
   Tagged: ethe 4/1
CPU-Protection: ON,  MVID: 0x001,  VPLS FID: 0x0000a00c
 Local Switching: Enabled
 Extended Counter: ON
```

show mpls vpls id

Syntax: show mpls vpls id *vpls id*

The *vpls id* variable specifies the ID of the VPLS instance for which you want to display debugging information.

This command shows the tunnel LSPs being used to forward VPLS traffic from a device to a peer. If VPLS traffic to the peer is being load balanced across multiple tunnel LSPs, then the command lists the tunnel LSPs used for load balancing, as shown in the following example.

```
Brocade# show mpls vpls id 2
VPLS b, Id 2, Max mac entries: 100000
 Routing Interface Id 3
 Total vlans: 1, Tagged ports: 1 (1 Up), Untagged ports 0 (0 Up)
 IFL-ID: n/a
  Vlan 444
   Tagged: ethe 4/5 ethe 4/5
 VC-Mode: Raw
 CPU-Protection: ON,  MVID: 0x005,  VPLS FIDs: 0x0000a00a, 0x0000a00a
 Local Switching: Enabled
 Extended Counter: ON
 Mutlicast Snooping: Disabled
```

show mpls vpls down

Syntax: show mpls vpls down

This command displays information about VPLS instances that are not fully operational, as shown in the following example.

```
Brocade# show mpls vpls down
The following VPLS'es are not completely operational:
                        Num   Num    Ports  Num    Peers   CPU
Name          Id        Vlans Ports  Up     Peers  Up      Prot
vpls1         1003      1     1      1      1      0       ON
vpls2         1004      0     0      0      1      0       ON
```

show mac vpls

Syntax: **show mac vpls** *vpls id mac address*

- *vpls id* - Specifies the VPLS ID.
- *mac address* - Specifies the MAC address.

The VPLS MAC database stores entries associating remote MAC addresses with the VC LSPs, and local MAC addresses with the CE devices. Each VPLS instance has a separate VPLS MAC database. Command output resembles the following example.

```
Brocade# show mac vpls
Total VPLS mac entries in the table: 10 (Local: 5, Remote: 5)
VPLS  MAC Address    L/R  Port Vlan/Peer       Age
====  ===========    ===  ==== =========       ===
1     0000.0000.1601 R    5/1  10.3.3.3         0
1     0000.0000.1003 L    5/3  2                0
1     0000.0000.1603 R    5/1  10.3.3.3         0
1     0000.0000.1005 L    5/3  2                0
1     0000.0000.1002 L    5/3  2                0
1     0000.0000.1605 R    5/1  10.3.3.3         0
1     0000.0000.1602 R    5/1  10.3.3.3         0
1     0000.0000.1004 L    5/3  2                0
1     0000.0000.1001 L    5/3  2                0
1     0000.0000.1604 R    5/1  10.3.3.3         0
```

This command displays the VPLS MAC database on the management processor for a VPLS instance specified by its VPLS ID.

```
Brocade# show mac vpls 1
Total MAC entries for VPLS 1: 10 (Local: 5, Remote: 5)
VPLS  MAC Address    L/R Port   Vlan/Peer      Age
====  ===========    === ====   =========      ===
1     0000.0000.1601 R   5/1    10.3.3.3        0
1     0000.0000.1003 L   5/3    2               0
1     0000.0000.1603 R   5/1    10.3.3.3        0
1     0000.0000.1005 L   5/3    2               0
1     0000.0000.1002 L   5/3    2               0
1     0000.0000.1605 R   5/1    10.3.3.3        0
1     0000.0000.1602 R   5/1    10.3.3.3        0
1     0000.0000.1004 L   5/3    2               0
1     0000.0000.1001 L   5/3    2               0
1     0000.0000.1604 R   5/1    10.3.3.3        0
```

This command displays a specific entry in the MAC database on the management processor.

```
Brocade# show mac vpls 1 0000.0000.1601
VPLS: 1            MAC: 0000.0000.1601        Age: 0
Remote MAC         Port: ethe 5/1             Peer: 10.3.3.3
Trunk slot mask: 00000000
```

show mpls statistics vpls

Syntax: show mpls statistics vpls [*vpls-name* | *vpls-id*]

- *vpls-name* - Indicates the configured name for a VPLS instance.
- *vpls-id* - Indicates the ID of a VPLS instance.

This command displays traffic statistics for all VPLS instances, or a specific instance. The following example shows statistics for all VPLS traffic.

```
Brocade# show mpls statistics vpls
VPLS-Name       In-Port(s)    Endpt-Out-Pkts      Tnl-Out-Pkts
---------       ----------    --------------      ------------
test2           e1/1          0                   0
                e1/2          0                   0
                e1/3          0                   0
                e1/4          0                   0
test2           e2/1 - e2/10  0                   0
                e2/11 - e2/20 0                   0
                e2/21 - e2/30 0                   0
                e2/31 - e2/40 0                   0
test3           e1/1          0                   0
                e1/2          0                   0
                e1/3          0                   0
                e1/4          0                   0
test3           e2/1 - e2/10  0                   0
                e2/11 - e2/20 0                   0
                e2/21 - e2/30 0                   0
                e2/31 - e2/40 0                   0
test4           e1/1          0                   0
                e1/2          0                   0
                e1/3          0                   0
                e1/4          0                   0
test4           e2/1 - e2/10  0                   0
                e2/11 - e2/20 0                   0
                e2/21 - e2/30 0                   0
                e2/31 - e2/40 0                   0
test4           e5/1          10354120822         0
                e5/2          0                   0
                e5/3          0                   2992416134
                e5/4          0                   0
```

NOTE
The VPLS name is repeated for each module from which the statistics are collected, to be displayed on the Management console.

show mpls debug vpls

Syntax: show mpls debug vpls *vpls id*

This command displays generic VPLS debug information. Enter a VPLS ID to display information about a specific VPLS instance, as shown in the following example.

```
Brocade# show mpls debug vpls 1
ID:       1      Name:     test 1
CPU-Prot: OFF    MVID:     INVD     FID:    0x00002002
MAC Info:
     Total MACs: 2  Local: 2  Remote: 0
     Max Exceed: 0 Table Full: 0
```

show mpls debug vpls local

Syntax: show mpls debug vpls local *num*

The *num* variable specifies the ID of the VPLS instance for which you want to display the debugging information.

This command displays information about all the VPLS local entries. Command output resembles the following example.

```
Brocade# show mpls debug vpls local
VPLS 1:

    VLAN  In-Tag      Port   Valid  Blocked  Pending    CCEP  R-CCEP-ST  VVPORT
    ====  ======      ====   =====  =======  =======    ====  =========  ======
    500   n/a         4/1    1      No       0         No    DOWN        1

    Local Broadcast Fids:
    =====================
    Vlan 500                     --  Fid: 00008007, Ports: 1
      Port 4/1                   --  Fid: 00000000

VPLS 45:


    VLAN  In-Tag      Port   Valid  Blocked  Pending    CCEP  R-CCEP-ST  VVPORT
    ====  ======      ====   =====  =======  =======    ====  =========  ======
    600   n/a        4/1    1      No       0         No    DOWN       n/a

    Local Broadcast Fids:
    =====================
    Vlan 600                     --  Fid: 00008007, Ports: 1
      Port 4/1                   --  Fid: 00000000
```

show mpls debug vpls remote

Syntax: show mpls debug vpls remote *vpls id*

The *vpls id* variable specifies the ID of the VPLS instance for which you want to display debugging information.

This command displays the state of all the configured VPLS peers for the specified VPLS ID. Command output resembles the following example.

```
Brocade# show mpls debug vpls remote 1
VPLS 1:
    num_valid_remote_entries:1

 * - Single Hop LSP
    Peer:    10.8.8.8           Valid:   Yes           Pending Delete: 0
    Label:   983040             Tagged:  No            Load Balance:   No
    MCT-SPK: No                 F-Port:  2/1           Num LSP Tnnls:  1
    VVPort: 2
```

```
       VC      Tunnel Byp/Det NHT         Use  Tunnel
Port  Label   Label  Label   Index   COS  COS  VIF Idx
====  ======  ====== ======  =====   ===  ===  =======
2/1   983040  1    * --      0       0    0    0
```

Internally, a maximum of four LSP tunnels is maintained to reach the peer. If load balancing is disabled, information for only one tunnel is displayed in the output.

show mpls debug vpls fsm-trace

Syntax: **show mpls debug vpls fsm-trace** *vpls id*

This command displays the VPLS peer FSM history trace that includes FSM state, the events that it received, and also the time stamp as to when the transition happened.

```
Brocade# show mpls debug vpls fsm-trace 1
Time               FSM State      Rcvd Event       Action
Oct 16 02:07:34   WAIT_PT        PORT_UP           A
Oct 16 02:17:34   WAIT_TNNL      TNNL_UP           B
Oct 16 02:17:34   WAIT_PW_UP     PW_UP             E
Oct 16 03:04:24   OPER           PW_DN             I
Oct 16 03:05:34   WAIT_PW_UP     PW_UP             E
```

Clearing VPLS traffic statistics

To clear the entries stored for all VPLS statistics, enter the following command.

clear mpls statistics vpls

Syntax: **clear mpls statistics vpls** [*vpls-name* | *vpls-id*]

To clear entries for a specific VPLS instance, enter the VPLS name or ID number.

- *vpls-name* - The configured name for a VPLS instance.
- *vpls-id* - The ID of a VPLS instance.

MPLS VPLS debug commands

This section describes the debug commands that generate MPLS VPLS information.

debug vpls

Syntax: [**no**] **debug vpls** [**cam** | **count** | **dy-sync** | **events** | **failover** | **filter** | **forwarding** | **generic** | **mac** | **statistics** | **topology**]

- **cam** - Displays information about VPLS CAM or PRAM programming.
- **count** - Displays information about the VPLS debug print counter.
- **dy-sync** - Displays information about VPLS table synchronization between management and CPU cards.
- **events** - Displays information about VPLS control-plane events.
- **failover** - Displays information about VPLS failover events.
- **filter** - Displays VPLS filtering options.
- **forwarding** - Displays information about VPLS CPU packet forwarding.

- **generic** - Enables generic VPLS debugging.
- **mac** - Displays VPLS MAC learning, aging, deletion, and movement.
- **statistics** - Displays information about VPLS statistics.
- **topology** - Displays information about VPLS topology group events.

debug vpls cam

Syntax: [**no**] **debug vpls cam** [**additions** | **deletions** | **updates**]

This command displays information about VPLS CAM or PRAM programming.

NOTE
This command is available only on the Line Processor.

- **additions** - Displays information about VPLS CAM or PRAM additions.
- **deletions** - Displays information about VPLS CAM or PRAM deletions.
- **updates** - Displays information about VPLS CAM or PRAM updates.

```
Brocade# debug vpls cam
        VPLS CAM:  all debugging is on
VPLS CAM-DEL: lp_cam_del_vpls_mac_cam_all() - Delete all CAMs for MAC
0000.0004.0000
VPLS CAM-DEL: lp_cam_del_vpls_mac_cam_single() - MAC 0000.0004.0000
(VPLS_SA_VC_ENTRY): deleted single CAM 0001800a and PRAM.
deleting cam-index 98314 for PPCR 1:1, CAM_TYPE:13
        cam-index chain:
VPLS CAM-ADD: lp_cam_add_vpls_mac_egress_one_ppcr() - Add CAM entry for MAC
0000.0004.0000, port 1/1, vc-label 000f0000, type VPLS_SA_VC_ENTRY.
VPLS CAM-ADD: lp_cam_add_vpls_mac_egress_one_ppcr() - SA-VC CAM add success. CAM
0001800a, new PRAM 000000c7.
adding cam-index 98314 for PPCR 1:1 CAM_TYPE:13
```

debug vpls cam additions

Syntax: [**no**] **debug vpls cam additions**

This command generates debug output when a VPLS CAM or PRAM addition operation is performed by VPLS due to the addition of MAC CAM or PRAM entry to the hardware.

debug vpls cam deletions

Syntax: [**no**] **debug vpls cam deletions**

This command generates debug output when a VPLS CAM or PRAM deletion operation is performed by VPLS due to the deletion of MAC CAM or PRAM entry from the hardware.

debug vpls cam updates

Syntax: [**no**] **debug vpls cam updates**

This command generates debug output when a VPLS CAM or PRAM entry is being updated due to a change in the PRAM information in regard to a forwarding decision.

debug vpls count

Syntax: [**no**] **debug vpls count**

This command specifies the number of debug prints generated by the **debug vpls** command. Use this command to limit high-volume displays such as MAC learning activity and certain dy-sync activity.

debug vpls dy-sync

Syntax: [**no**] **debug vpls dy-sync** [**local** | **mac** | **remote** | **tlv**]

- **local** - Displays information about VPLS local entry dy-sync.
- **mac** - Displays information about VPLS MAC table dy-sync.
- **remote** - Displays information about VPLS remote entry dy-sync.
- **tlv** - Displays information about VPLS TLV dy-sync.

This command monitors all VPLS dy-sync activity. Dy-sync is a method of synchronizing internal VPLS data between management and CPU cards. This includes VPLS configuration (TLV), local entry or endpoint status (local), remote entry or VPLS peer status (remote), and VPLS MAC table activity. This data is used generally for internal debugging.

debug vpls dy-sync local

Syntax: [**no**] **debug vpls dy-sync local**

This command generates information about local VPLS dy-sync activity. Command output resembles the following example for a specified VLAN (VLAN ID 2).

```
Brocade# debug vpls dy-sync local
Brocade(config-mpls-vpls-test1)# vlan 2
Brocade(config-mpls-vpls-test1-vlan-2)# no tag e 2/19
VPLS MAC-GROUP: vpls_mac_delete_local_entry() - VPLS 1, port 2/19, vlan 2: HW CAMs
flushed 620.
VPLS DY-SYNC-LOC: mpls_vpls_pack_one_vpls_local_entry() - VPLS 1, action 2, vlan
2, port 2/19,
replace-vlan 0.
VPLS DY-SYNC-TLV: mpls_vpls_pack_one_ipc_tlv_vpls_vlan_port() - VPLS 1, vlan 2,
port 2/19.
VPLS DY-SYNC-TLV: mpls_vpls_pack_one_ipc_tlv_port_config() - VPLS 1, port 2/19,
vlan 2, mode 1.
VPLS DY-SYNC-TLV: mpls_vpls_sync_timer() - Flush TLV packet
VPLS DY-SYNC-LOC: mpls_vpls_sync_timer() - Flush LOCAL packet
R4(config-mpls-vpls-test1-vlan-2)#
R4(config-mpls-vpls-test1-vlan-2)#tag e 2/19
VPLS DY-SYNC-TLV: mpls_vpls_pack_one_ipc_tlv_port_config() - VPLS 1, port 2/19,
vlan 2, mode 1.
VPLS DY-SYNC-LOC: vpls_mac_add_local_entry() - Add: VPLS 1, vlan 2, port 2/19.
VPLS DY-SYNC-LOC: mpls_vpls_pack_one_vpls_local_entry() - VPLS 1, action 1, vlan
2, port 2/19,
replace-vlan 1.
VPLS DY-SYNC-TLV: mpls_vpls_pack_one_ipc_tlv_vpls_vlan_port() - VPLS 1, vlan 2,
port 2/19.
VPLS DY-SYNC-TLV: mpls_vpls_sync_timer() - Flush TLV packet
VPLS DY-SYNC-LOC: mpls_vpls_sync_timer() - Flush LOCAL packet
```

debug vpls dy-sync mac

Syntax: [**no**] **debug vpls dy-sync mac**

The following output was generated with **debug vpls dy-sync mac** enabled and indicates that an existing VPLS MAC has been deleted and then reinstalled.

```
Brocade# debug vpls dy-sync mac
Brocade# clear mac vpls e 2/19
VPLS MAC: mpls_vpls_delete_mac_entry_from_table() - VPLS 1, MAC 0000.0000.0601
1 mac entries flushed
VPLS MAC-LOCAL: mpls_vpls_mac_sync_itc_callback() - VPLS_MAC_SYNC_LOCAL_ENTRY:
MAC
0000.0000.0601, port 2/19, vlan 2
VPLS MAC-LOCAL: vpls_local_sa_learning() - MAC 0000.0000.0601, port 2/19, vlan 2.
VPLS MAC: mpls_vpls_insert_mac_entry_in_table() - VPLS 1, MAC 0000.0000.0601
VPLS DY-SYNC-MAC: mpls_vpls_mp_send_create_delete_one_mac_entry() - VPLS 1,
action 1, mac 0000.0000.0601
VPLS DY-SYNC-MAC: mpls_vpls_pack_mac_table_change() - VPLS 1, action 1, mac
0000.0000.0601, local 1.
VPLS MAC-LOCAL: mpls_vpls_mac_sync_itc_callback() - VPLS_MAC_SYNC_LOCAL_ENTRY:
MAC
0000.0000.0601, port 2/19, vlan 2
VPLS MAC-LOCAL: vpls_local_sa_learning() - MAC 0000.0000.0601, port 2/19, vlan 2.
VPLS MAC: mpls_vpls_insert_mac_entry_in_table() - VPLS 1, MAC 0000.0000.0601
VPLS MAC-LOCAL: vpls_local_sa_learning() - Existing entry.
VPLS DY-SYNC-MAC: mpls_vpls_mp_send_create_delete_one_mac_entry() - VPLS 1,
action 1, mac 0000.0000.0601
VPLS DY-SYNC-MAC: mpls_vpls_pack_mac_table_change() - VPLS 1, action 1, mac
0000.0000.0601, local 1.
VPLS DY-SYNC-MAC: mpls_vpls_mp_send_create_delete_one_mac_entry() - VPLS 1,
action 3, mac 0000.0000.0601
VPLS DY-SYNC-MAC: mpls_vpls_pack_mac_table_change() - VPLS 1, action 3, mac
0000.0000.0601, local 1.
VPLS DY-SYNC-MAC: mpls_vpls_mp_send_create_delete_one_mac_entry() - VPLS 1,
action 3, mac 0000.0000.0601
VPLS DY-SYNC-MAC: mpls_vpls_pack_mac_table_change() - VPLS 1, action 3, mac
0000.0000.0601, local 1.
VPLS DY-SYNC-MAC: mpls_vpls_mp_send_create_delete_one_mac_entry() - VPLS 1,
action 3, mac 0000.0000.0601
VPLS DY-SYNC-MAC: mpls_vpls_pack_mac_table_change() - VPLS 1, action 3, mac
```

debug vpls dy-sync remote

Syntax: [**no**] **debug vpls dy-sync remote**

This command generates information about VPLS remote entry dy-sync activity. The following output was generated with this command enabled, and indicates that an existing peer was deleted and then reinstalled.

```
Brocade# debug vpls dy-sync remote
Brocade(config-mpls-vpls-test1-vlan2)#no vpls-peer 10.5.5.5
VPLS MAC-GROUP: vpls_mac_delete_remote_entry () - VPLS 1, VC-label 000f0000: HW
CAMs flushed 4.
VPLS DY-SYNC-REM: mpls_vpls_send_remote_entry_info () - called
VPLS DY-SYNC-REM: mpls_vpls_sync_timer () - Flush REMOTE packet
(config-mpls-vpls-test1-vlan-2)# vpls-peer 10.5.5.5
VPLS DY-SYNC-REM: mpls_vpls_pack_one_vpls_remote_entry_tnns () - called
VPLS DY-SYNC-REM: mpls_vpls_pack_one_vpls_remote_entry_tnnl () - called
VPLS DY-SYNC-REM: mpls_vpls_pack_one_vpls_remote_entry () - called
VPLS DY-SYNC-REM: mpls_vpls_send_remote_entry_info () - called
VPLS DY-SYNC-REM: mpls_vpls_sync_timer () - Flush REMOTE packet
```

debug vpls dy-sync tlv

Syntax: [no] debug vpls dy-sync tlv

This command generates information about the VPLS configuration that is being synchronized (dy-sync) between management and CPU cards. The following output results when a VPLS instance is deleted and then reinstalled.

```
Brocade# debug vpls dy-sync tlv
Brocade(config-mpls)# no vpls 1 1
VPLS DY-SYNC-TLV: mpls_vpls_pack_one_ipc_tlv_vpls_vlan_port() - VPLS 1, vlan 2, 
port 1/2.
VPLS DY-SYNC-TLV: mpls_vpls_pack_one_ipc_tlv_port_config() - VPLS 1, port 1/2, 
vlan 2, mode 0.
VPLS DY-SYNC-TLV: mpls_vpls_pack_one_ipc_tlv_vpls_vlan() - VPLS 1, vlan 2, 
vlan-fid 65535.
VPLS DY-SYNC-TLV: mpls_vpls_send_ipc_delete_vpls_table() - VPLS 1.

Brocade(config-mpls)# vpls 1 1
Brocade(config-mpls-vpls-1)# vpls-peer 10.200.200.1
Brocade(config-mpls-vpls-1)# vlan 2
Brocade(config-mpls-vpls-1-vlan-2)# untagged ethe 1/2
VPLS DY-SYNC-TLV: mpls_vpls_pack_one_ipc_tlv_vpls_name() - VPLS 1, action 1, name 
1.
VPLS 1, action 1, value 0.
VPLS DY-SYNC-TLV: mpls_vpls_pack_one_ipc_tlv_vpls_fid() - VPLS 1, action 1, Fid 
0x0000a002.
VPLS DY-SYNC-TLV: mpls_vpls_pack_one_ipc_tlv_vpls_mvid() - VPLS 1, action 1, 
value 2048.
VPLS DY-SYNC-TLV: mpls_vpls_pack_one_ipc_tlv_vlan_id() - VPLS 1, vlan 2.
VPLS DY-SYNC-TLV: mpls_vpls_pack_one_ipc_tlv_vpls_vlan() - VPLS 1, vlan 2, 
vlan-fid 32772.
```

debug vpls events

Syntax: [no] debug vpls events

This command displays information about VPLS label-range allocation, LDP session status (up or down), LSP tunnel switchovers, VPLS endpoint port transitions, VPLS peer state machine transitions, label exchanges, VC bindings, and VC withdrawals. The following example shows an MPLS uplink being disabled and then re-enabled.

```
Brocade# debug vpls events
Brocade(config-if-e1000-2/9)# disable
VPLS EVENT-LSP: mpls_vpls_process_tnnl_down()- VPLS test1: Peer 10.5.5.5, tunnel 
1 is down.
VPLS EVENT-LSP: mpls_vpls_del_vpls_peer_in_tnnl_list()- VPLS test1: Delete peer 
10.5.5.5 from tunnel-list of 1.
VPLS EVENT-FSM: mpls_vpls_peer_fsm_step()- VPLS fsm: vpls test1 peer 10.5.5.5 
event
TNNL_DN st OPER->WAIT_TNNL act F.
VPLS EVENT-FSM: mpls_vpls_peer_fsm_step()- VPLS: vpls test1 peer 10.5.5.5 is now 
DOWN, num up peer->0.
VPLS MAC-GROUP: vpls_mac_delete_remote_entry()- VPLS 1, VC-label 000f0000: HW 
CAMs flush 4.
VPLS EVENT-PEER: mpls_vpls_send_vc_withdrawal()- VPLS 1, group 0, peer 10.5.5.5, 
Send label withdraw for 983040.
(config-if-e1000-2/9)#enable
VPLS EVENT-LDP: mpls_ldp_peer_session_ind()- VPLS test1, peer 10.5.5.5 LDP 
session is down.
```

```
VPLS EVENT-FSM: mpls_vpls_peer_fsm_ste ()- VPLS fsm: vpls test1 peer 10.5.5.5
event LDP_DN
st WAIT_TNNL->WAIT_TNNL act -.
VPLS EVENT-LSP: mpls_vpl_process_tnnl_up()- VPLS test1: Peer 10.5.5.5, tunnel 1 is
up.
VPLS EVENT-LSP: mplc_vpls_add_vpls_peer_in_tnnl_list() = VPLS test1: Add peer
10.5.5.5 to tunnel-list of 1.
VPLS EVENT-FSM: mpls_vpls_peer_fsm_step()- VPLS fsm: vpls test1 peer 10.5.5.5,
send local-label 983040.
VPLS EVENT-FSM: mpls_vpls_peer_fsm_step()- VPLSfsm: vpls test 1 peer 10.5.5.5
event
RCV_LBL st WAIT_VC->OPER act D.
VPLS EVENT-FSM: mpls_vpls_peer_fsm_step()- VPLS: vpls test1 peer 10.5.5.5 is now
UP, num up peer->1.
```

debug vpls failover

Syntax: [**no**] **debug vpls failover**

This command enables debugging of VPLS failover-related events, as shown in the following example.

```
Brocade# debug vpls failover
        VPLS Failover:  debugging is on
Oct  8 18:36:36 VPLS DY-SYNC-TLV: mpls_vpls_pack_one_tlv_tsl_to_stdby() -
Oct  8 18:36:36 VPLS FAILOVER: mpls_vpls_send_to_stdby_tlv_peer_tsl() - called
SYSLOG: <14>Oct  8 18:36:37 FWD2(XMR12) System: Interface ethernet 1/19, state
down - link down
SYSLOG: <14>Oct  8 18:36:53 FWD2(XMR12) System: Interface ethernet 1/20, state
down - link down
Oct  8 18:36:53 VPLS DY-SYNC-TLV: mpls_vpls_pack_one_tlv_tsl_to_stdby() -
Oct  8 18:36:53 VPLS FAILOVER: mpls_vpls_send_to_stdby_tlv_peer_tsl() - called
SYSLOG: <13>Oct  8 18:36:53 FWD2(XMR12) MPLS: VPLS 1 peer 10.6.6.6 (VC ID 1) is
down
SYSLOG: <14>Oct  8 18:37:10 FWD2(XMR12) System: Interface ethernet 1/20, state up
rldf_complete_lsi_rsv_conn_xc: update sync node
rldf_complete_lsi_rsv_conn_xc: update sync node for VC FEC
rldf_complete_lsi_rsv_conn_xc: update sync node for VC FEC
Oct  8 18:37:23 VPLS DY-SYNC-TLV: mpls_vpls_pack_one_tlv_tsl_to_stdby() -
Oct  8 18:37:23 VPLS FAILOVER: mpls_vpls_send_to_stdby_tlv_peer_tsl() - called
SYSLOG: <13>Oct  8 18:37:23 FWD2(XMR12) MPLS: VPLS 1 peer 10.6.6.6 (VC ID 1) is up
```

debug vpls filter

Syntax: [**no**] **debug vpls filter** [**b-src-mac** | **b-dst-mac** | **dst-mac-address** | **id** | **inner-tag** | **outer-vlan** | **src-mac-address** | **src-port** | **topo-id** | **topo-hw-idx** | **vc-label** | **vpls-peer-ip-address**]

This command displays VPLS filtering options.

- **b-src-mac** - Filter to include outer source MAC address.
- **b-dst-mac** - Filter to include outer destination MAC address.
- **dst-mac-address** - Filter to include destination MAC address (LP only).
- **id** - Filter to include VPLS ID.
- **inner-tag** - Filter to include inner-tag ID (VLAN or ISID).
- **outer-vlan** - Filter to include VPLS outer VLAN.
- **src-mac-address** - Filter to include source MAC address.

- **src-port** - Filter to include source port.
- **topo-id** - Filter to include topology group ID (MP only).
- **topo-hw-idx** - Filter to include topology group hardware index (LP only).
- **vc-label** - Filter to include local VC label.
- **vpls-peer-ip-address** - Filter to include this VPLS peer IP address.

NOTE
To clear a specific filter, enter the **no debug vpls filter id** *num* command from the MP or LP console. To clear all VPLS filters, enter the **no debug vpls filter** command from the MP or LP console.

---

debug vpls forwarding

Syntax: [**no**] **debug vpls forwarding**

This command is used to generate some CPU packet forwarding in regard to VPLS traffic. On the LP, it generates how the packet was received and how it was software forwarded. On the MP, it generates the packet handling of those applications such as Multicast or 802.1ag to send out control packets through VPLS to a remote peer.

```
Brocade# debug vpls forwarding
        VPLS Forwarding:  debugging is on
VPLS EGRESS FWD: lp_l2_vpls_outbound_process_packet() - RX pkt from MPLS uplink
src-port:1/1. VC label:983040
VPLS EGRESS FWD: lp_l2_vpls_outbound_process_packet() - DA missed in CAM.
VPLS EGRESS FWD: lp_l2_vpls_outbound_forward_packet() - Payload Tag 0x81000008
exisits.
VPLS EGRESS FWD: lp_l2_vpls_outbound_forward_packet() - Unknown VPLS MAC
0000.0003.0000, VPLS 1.
VPLS EGRESS FWD: lp_l2_vpls_outbound_forward_packet() - UNKNOWN: send pkt to all
end-points.
VPLS EGRESS FWD: lp_l2_vpls_broadcast_local() - Bcast pkt to VPLS VLAN 300,
inner_vid_valid:0 inner_vlan_id 0x00000000, FID 0x800b, Egress:1,
priority:0x00000000, inner_vlan_priority:0x00000000
VPLS EGRESS FWD: lp_l2_vpls_broadcast_local() - Bcast pkt to VPLS VLAN 3000,
inner_vid_valid:0 inner_vlan_id 0x00000000, FID 0x800c, Egress:1,
priority:0x00000000, inner_vlan_priority:0x00000000
```

debug vpls generic

Syntax: [**no**] **debug vpls generic**

This command generates generic information about VPLS events such as VPLS MAC table allocation failures, and VPLS MAC table deletions.

debug vpls mac

Syntax: [**no**] **debug vpls mac** [**errors** | **group** | **local** | **remote**]

- **errors** - Displays information about VPLS MAC errors.
- **group** - Displays information about a VPLS MAC group.
- **local** - Displays information about VPLS local MAC learning.
- **remote** - Displays information about VPLS remote MAC learning.

This command generates information about VPLS MAC learning, aging, deletions, and topology changes. This command can be enabled for either local or remote MAC monitoring (or both).

NOTE
Use of this command can generate a large number of debug prints. To limit debug prints, use the **debug vpls count** command.

---

debug vpls mac local

Syntax: [**no**] **debug vpls mac local**

This command generates information about local MAC activity, as shown in the following example.

```
Brocade# debug vpls mac local
VPLS MAC: mpls_vpls_delete_mac_entry_from_table() - VPLS 1, MAC 0000.0000.0601
VPLS MAC-GROUP: mac_group_delete_mac_entry_from_list()- 26d9f025: pt/lbl 58, vpls
1,
update_head 0, p 00000000, n 00000000.
VPLS MAC-GROUP: mp_vpls_mac_send_group_flush_msg()- Send IPC to LPs: Flush MACs for
VPLS
4294967295, port/label 0000003a,  is-vlan 0, hw-only 0.
VPLS MAC-GROUP: mac_flush_by_group() - VPLS 4294967295, port/label 0000003a,
hw-only 0:
Entries processed 1.
1 mac entries flushed
```

debug vpls statistics

Syntax: [**no**] **debug vpls statistics**

This command can be used to monitor statistics collection activity for VPLS instances. The following output was generated with **debug vpls statistics** enabled, and as a result of the **show mpls statistics vpls** command.

```
Brocade# debug vpls statistics
debug vpls statistics is enabled
Brocade# show mpls statistics vpls
VPLS STATS: get_mpls_vpls_stat_from_lp () - VPLS 1.
VPLS STATS: get_mpls_vpls_stat_from_lp () - VPLS 1
VPLS-Name   In-Port(s)   Endpt-Out-Pkts   Tnl-Out-Pkts
---------   ----------   --------------   --------------
test1       e2/1 - e2/20 1252              0
test1       e4/1 - e4/2  1260              0
test1       e4/3 - e4/4  0                 0
```

debug vpls topology

Syntax: [**no**] **debug vpls topology**

This command is used to debug the topology group-related handling in the VPLS area. If there are any VPLS VLANs configured in a topology group, this command allows you to see what is taking place when topology group events occurred, such as topology group master VLAN changes, forwarding state changes, and so on.

NOTE
This command is available only on the Management Processor.

```
Brocade# debug vpls topology
VPLS TOPO: mpls_vpls_topo_itc_membership_update() - Send ITC update: Topo 2 VPLS
VLANs exist=1.
```

```
VPLS TOPO: mpls_vpls_topo_inherit_control_state() - Inherit control state for
Topo 2.
VPLS TOPO: mpls_vpls_topo_add_vpls_vlan() - Add VPLS 1 VLAN 300:0xffffffff to topo
(2) member list. Member count 1
VPLS TOPO: mpls_vpls_topo_update_end_points() - Topo 1 VLAN 300:0xffffffff old
topo_hw_index 0x0000ffff new topo_hw_index 0x00000000
VPLS TOPO: mpls_vpls_topo_update_end_points() - Update VPLS end-point state: VPLS
1 VLAN 300:0xffffffff Port 1/5 Block 0. topo_hw_index:0x00000000
VPLS TOPO: mpls_vpls_topo_add_member_vlan() - Set VPLS 1 VLAN 300:0xffffffff as
member of topo ID 2. with topo_hw_index 0x00000000
```

debug vpls fsm-trace

Syntax: [**no**] **debug vpls fsm-trace** *vpls_id vpls_peer_ IP_ address*

This command shows VPLS peer FSM trace history. Command output resembles the following example.

```
Brocade# debug vpls fsm-trace 1 10.11.11.11
Time                 FSM State            Rcvd Event           Action
==============       ==================== ==================== ======
Nov 16 22:52:51      0 WAIT_PT            PARAM_UPDT           -
Nov 16 22:53:24      0 WAIT_PT            PORT_UP              A
Nov 16 22:53:38      1 WAIT_TNNL          TNNL_UP              B
Nov 16 22:56:41      2 WAIT_PW_UP         PW_UP                E
Nov 16 22:57:10      3 OPER               PW_DN                I
Nov 16 22:57:22      2 WAIT_PW_UP         PW_UP                E
Nov 16 22:58:02      3 OPER               PORT_DN              G
Nov 16 22:58:02      0 WAIT_PT            WITHD_SENT           -
Nov 16 22:58:02      7 WWD(WAIT_PT)       WITHD_DONE           -
Nov 16 22:58:43      0 WAIT_PT            PORT_UP              A
Nov 16 22:58:43      1 WAIT_TNNL          TNNL_UP              B
Nov 16 22:58:43      2 WAIT_PW_UP         PW_UP                E
Nov 16 22:59:18      3 OPER               TNNL_DN              G
Nov 16 22:59:18      1 WAIT_TNNL          WITHD_SENT           -
Nov 16 22:59:18      5 WWD(TNNL_DWN)      WITHD_DONE           -
```

Common diagnostic scenarios

- You cannot configure MPLS on a port that is part of a dynamic link aggregation.

  Where MPLS is enabled globally on the device, a port that is configured in a trunk can be enabled as an MPLS interface port to create an MPLS trunk. You can either include a primary trunk port that has already been MPLS-enabled in a new trunk or MPLS-enable a primary trunk port of an already configured trunk. This feature was introduced in version 03.5.00 of the Multi-Service IronWare software. The following considerations must be considered when configuring MPLS on a trunk:

  - Only static server trunks and per-packet server trunks are supported.
  - Switch and LACP trunks are not supported.
  - MPLS is enabled on the primary port of the trunk and this enables MPLS on the entire trunk. Secondary ports of the trunk cannot be individually configured for MPLS.
- MPLS interfaces are not forwarding traffic.

This may be caused because the customer is running software codes that do not match. It is recommended that customers always update software so that all interface modules are running the same code. If you have questions about your software version, contact Brocade Technical Support for assistance.

- For a VPLS VLAN that is configured as multicast active, VPLS does not accept a receiver report.

  This issue is resolved by upgrading the software version to include the latest patches.

- MPLS traffic is lost.

  This issue is resolved by upgrading the software version to include the latest patches.

Chapter

Layer 3 Protocol Diagnostics 6

In this chapter

This chapter describes the diagnostic commands for Brocade NetIron XMR and Brocade MLX series Layer 3 protocol environments.

BFD

Bidirectional Forwarding Detection (BFD) quickly detects the failure of a forwarding path by confirming that the next-hop router is alive. Without BFD, failure detection can take from 3 to 30 seconds and may cause unacceptable levels of packet loss.

BFD show commands

This section describes the show commands that display BFD information.

show bfd

Syntax: show bfd

This command displays information about BFD activity, as shown in the following example.

```
Brocade# show bfd
 BFD State: ENABLED Version: 1 Use PBIF Assist: Y
 Current Registered Protocols: bgp/1 ospf/0  ospf6/0 bgp/0
 All Sessions: Current: 4 Maximum Allowed: 100 Maximum Exceeded Count: 0
 LP Sessions: Maximum Allowed on LP: 40  Maximum Exceeded Count for LPs: 0
   LP Sessions LP Sessions LP Sessions LP Sessions
   1  0        2  2        3  2        4  0
   5  0        6  0        7  0        8  0
   9  0        10 0        11 0        12 0
   13 0        14 0        15 0        16 0
```

```
BFD Enabled ports count: 2
 Port       MinTx       MinRx       Mult Sessions
 eth 2/1    100         100         3    2
```

show bfd application

Syntax: show bfd application

This command displays information about BFD applications, as shown in the following example.

```
Brocade# show bfd application
Registered Protocols Count: 3
Protocol  VRFID  Parameter
ospf       0     1
ospf6      0     0
isis_task  0     0
bgp        0     0
```

show bfd neighbor

Syntax: show bfd neighbor [**interface ethernet** *slotnum/port-num* | **interface pos** *slotnum/port-num* | **interface ve** *port-num*]

- **interface ethernet** *slotnum/port-num* - Displays BFD neighbor information for the specified Ethernet interface.
- **interface pos** *slotnum/port-num* - Displays POS information for a specific interface, slot, and port.
- **interface ve** *port-num* - Displays BFD neighbor information for the specified virtual interface.

When the **show bfd application** command is enabled, the **show bfd neighbor** command displays output similar to the following example.

```
Brocade# show bfd neighbor
Total Entries:7 R:RXRemote(Y:Yes/N:No)H:Hop(S:Single/M:Multi)
NeighborAddress   state  Interface  Holddown   Interval   R/H
10.3.0.1          UP     ve 30      600000     200000     Y/S
10.31.31.14       UP     ve 30      25000000   5000000    Y/M
10.4.0.1          UP     ve 40      600000     200000     Y/S
10.1.0.1          UP     ve 10      300000     100000     Y/S
10.11.11.14       UP     ve 10      25000000   5000000    Y/M
```

show bfd neighbor detail

Syntax: show bfd neighbor detail [*IP-address* | *IPv6-address*]

- *IP-address* - Displays BFD neighbor information for the specified IPv4 address.
- *IPv6-address* - Displays BFD neighbor information for the specified IPv6 address.

This command displays BFD neighbor information in the detailed format.

```
Brocade# show bfd neighbor detail
Total number of Neighbor entries: 2
NeighborAddress                         State   Interface Holddown  Interval  RH
10.1.1.1                                UP      eth 4/1   1800000   600000    1
  Registered Protocols(Protocol/VRFID): isis_task/0 ospf/0
  Local: Disc: 1, Diag: 0, Demand: 0 Poll: 0
         MinTxInterval: 500000, MinRxInterval: 500000, Multiplier: 3
  Remote: Disc: 684, Diag: 0, Demand: 0 Poll: 0
```

```
         MinTxInterval: 600000, MinRxInterval: 600000, Multiplier: 3
  Stats: RX: 4617 TX: 5189 SessionUpCount: 1 at SysUpTime: 0:0:41:49.325
  Session Uptime: 0:0:40:54.400, LastSessionDownTimestamp: 0:0:0:0.0
NeighborAddress                        State   Interface Holddown  Interval  RH
fe80::202:17ff:fe6e:c41d               DOWN    eth 4/1   0         1000000   0
  Registered Protocols(Protocol/VRFID): ospf6/0
  Local: Disc: 2, Diag: 0, Demand: 0 Poll: 0
        MinTxInterval: 500000, MinRxInterval: 500000, Multiplier: 3
  Remote: Disc: 0, Diag: 0, Demand: 0 Poll: 0
         MinTxInterval: 0, MinRxInterval: 0, Multiplier: 0
  Stats: RX: 0 TX: 3278 SessionUpCount: 0 at SysUpTime: 0:0:41:49.325
  Session Uptime: 0:0:0:0.0, LastSessionDownTimestamp: 0:0:0:0.0
Using PBIF Assist: Y
```

Clearing BFD neighbor sessions

You can clear all BFD neighbor sessions or a specified BFD neighbor session using the following command.

clear bfd neighbor

Syntax: **clear bfd neighbor** [*IP-address* | *IPv6-address*]

- *IP-address* - Specifies the IPv4 address of a neighbor whose BFD session you want to clear.
- *IPv6-address* - Specifies the IPv6 address of a neighbor whose BFD session you want to clear.

Executing this command without specifying an IPv4 or IPv6 address clears the sessions of all BFD neighbors.

BFD debug commands

This section describes how to use diagnostic debug commands to monitor BFD environments.

debug bfd

Syntax: [**no**] **debug bfd** [**application** | **hitless-upgrade** | **ipc-error** | **ipc-event** | **itc** | **pbif** | **timer**]

- **application** - Displays information about BFD applications.
- **hitless-upgrade** - Displays information about hitless upgrade events.
- **ipc-error** - Displays information about interprocess communication (IPC) errors.
- **ipc-event** - Displays information about IPC events.
- **itc** - Displays information about BFD inter-task communication (ITC) activity.
- **pbif** - Enables debugging of BFD use of Peripheral Bus Interface FPGA (PBIF) Assist.
- **timer** - Enables debugging of BFD timer operation on LP.

debug bfd application

Syntax: [**no**] **debug bfd application**

This command generates information about BFD application activity. When this command is enabled, output resembles the following example, which shows that the OSPF application was disabled, and then enabled.

```
Brocade# debug bfd application
        application:  debugging is on
BFD/APP: Application: AppId: ospf App Subid: 0 DeRegistered with BFD
BFD/APP: Application: AppId: ospf App Subid: 0 Registered with BFD
```

debug bfd hitless-upgrade

Syntax: [**no**] **debug bfd hitless-upgrade**

This command displays information about BFD hitless upgrade events, as shown in the following example.

```
Brocade# debug bfd hitless-upgrade
MP switchover done, clearing all session without graceful restart app.
bfd_mp_delete_all_app_sessions_with_no_graceful_restart() called
BFD: LP Hitless Upgrade started

Resetting LP 1...
Resetting LP 2...
Resetting LP 3...
Resetting LP 4...
BFD/IPC: saving of IPC message during LP Upgrade started
BFD/IPC: Saving of IPC message during LP Upgrade finished.
BFD/IPC: sending saved ipc to LP
bfd_mp_add_all_app_sessions_with_no_graceful_restart called
BFD: LP Hitless Upgrade finished
```

debug bfd ipc-error

Syntax: [**no**] **debug bfd ipc-error**

This command generates information about BFD interprocess communication (IPC) errors. The following example shows output from a session with **debug ip ospf bfd**, **debug bfd ipc-event**, and **debug bfd ipc-error** enabled.

```
Brocade1# debug ip ospf bfd
         OSPF:  BFD events debugging is on
Brocade# debug bfd ipc-error
        ipc-error:  debugging is on
Brocade# debug bfd ipc-event
        ipc-event:  debugging is on
Brocade# show bfd neighbor
Dec 19 14:51:37 BFD/IPC:Sending MP Request for all sessions information to LP 2.
Dec 19 14:51:37 BFD/IPC:Received LP Response for all sessions information from LP
2.
Total number of Neighbor entries: 2
NeighborAddress                        State   Interface Holddown  Interval  RH
10.2.2.2                               UP      eth 2/2   300000    100000    1
fe80::20c:dbff:fee2:b529               UP      eth 2/2   300000    100000    1
SYSLOG: Dec 19 14:51:58:<13>R1, OSPF: nbr state changed, rid 10.1.1.1, nbr addr
10.2.2.2, nbr rid 10.2.2.2, state down
SYSLOG: Dec 19 14:51:58:<13>R1, OSPF: interface state changed, rid 10.1.1.1, intf
addr 10.2.2.1, state designated router
BFD/ITC: Received Delete Session Request from App:ospf for Port:eth
2/2 Neighbor:10.2.2.2
BFD: ipc set session admin down(0->2), for session 22
BFD/IPC: Received session parameter change for session=22
BFD: ipc delete session (0->2), for session 22
BFD/ITC: Received Create Session Request from App:ospf for Port:eth 2/2
Neighbor:10.2.2.2
```

```
BFD: ipc create session (0->2), for session 76
SYSLOG: Dec 19 14:52:01:<13>R1, OSPF: interface state changed, rid 10.1.1.1, intf
addr 10.2.2.1, state backup designated router
BFD/IPC: Received session parameter change for session=76
BFD/IPC: Received session state change notification for session=76
OSPF: ITC Session State Change Notification rxd for nbr 10.2.2.2
SYSLOG: Dec 19 14:52:01:<13>R1, OSPF: nbr state changed, rid 10.1.1.1, nbr addr
10.2.2.2, nbr rid 10.2.2.2, state full
Brocade#show bfd neighbor
BFD/IPC: Sending MP Request for all sessions information to LP 2.
BFD/IPC: Received LP Response for all sessions information from LP 2.
Total number of Neighbor entries: 2
NeighborAddress                       State   Interface Holddown  Interval  RH
fe80::20c:dbff:fee2:b529              UP      eth 2/2   300000    100000    1
10.2.2.2                              UP      eth 2/2   300000    100000    1
```

debug bfd ipc-event

Syntax: [no] debug bfd ipc-event

These commands generate information about IPC and ITC errors. The previous example shows output from a session with these two commands enabled.

debug bfd itc

Syntax: [no] debug bfd itc

This command displays information about BFD ITC activity. Command output resembles the following example.

```
Brocade# debug bfd itc
        itc:  debugging is on
Brocade# show bfd neighbor
BFD/ITC: Received Delete Session Request from App:ospf for Port:eth 2/2
Neighbor:10.2.2.2
BFD/ITC: Received Create Session Request from App:ospf for Port:eth 2/2
Neighbor:10.2.2.2
OSPF: ITC Session State Change Notification rxd for nbr 10.2.2.2
```

debug bfd pbif

Syntax: [no] debug bfd pbif

This command enables debugging of PBIF Assist feature that is used in the transmission of BFD packets. The following example shows output from a session with **debug bfd pbif** and **debug bfd ipc-event** enabled.

```
Brocade# debug bfd pbif
          BFD:  pbif debugging is on
Brocade# debug bfd ipc-event
          BFD:  ipc-event debugging is on
Feb 11  07:49:28 BFD/IPC: Received set use pbif assist to FALSE
Feb 11  07:49:28 BFD: Use of PBIF TX Assist Disabled, Disabling Use of PBIF TX A
ssist for all sessions
Feb 11  07:49:28 BFD: Tx PBIF Assist disabled for session 1
Feb 11  07:49:28 BFD: send IPC to MP for Use PBIF Assist change  to N  for sessi
on 1
Feb 11  07:49:56 BFD/IPC: Received set use pbif assist to TRUE
Feb 11  07:49:56 BFD: Use of PBIF TX Assist enabled,  Enabling PBIF TX Assist fo
r all qualified sessions
```

```
Feb 11  07:49:56 BFD: PBIF TX Assist Request successful for session 1 with inter
val 925 ms handle 1a480140
Feb 11  07:49:56 BFD: send IPC to MP for Use PBIF Assist change  to Y  for sessi
on 1
```

debug bfd timer

Syntax: [**no**] **debug bfd timer**

This command enables debugging of BFD timer operation.

The following example shows the output from a session with **debug bfd timer**, **debug bfd ipc-event**, and **debug bfd pbif** enabled.

```
Brocade# debug bfd timer
          BFD:  timer debugging is on
Brocade# debug bfd pbif
          BFD:  pbif debugging is on
Brocade# debug bfd ipc-event
          BFD:  ipc-event debugging is on
Feb 11  07:52:41 BFD: Timer called at interval of 46 ms
Feb 11  07:52:41 BFD: Timer called at interval of 17 ms
Feb 11  07:52:41 BFD: Negotiated TX Interval 1000000 microsec,  Jittered Transmi
t Interval 822 ms, random num 1572
Feb 11  07:52:41 BFD: Tx PBIF Assist disabled for session 1
Feb 11  07:52:41 BFD: send IPC to MP for Use PBIF Assist change  to N  for sessi
on 1
Feb 11  07:52:41 BFD: ipc session state change to MP, for session 1
Feb 11  07:52:41 BFD: ipc session parameters change to MP, for session 1
Feb 11  07:52:42 BFD: Timer called at interval of 56 ms
Feb 11  07:52:42 BFD/IPC: Received set session admin down for session=1
Feb 11  07:52:42 BFD: ipc session parameters change to MP, for session 1
Feb 11  07:52:42 BFD/IPC: Received delete session for session=1
Feb 11  07:52:42 BFD: Timer called at interval of 19 ms
Feb 11  07:52:50 BFD/IPC: Received create session for session=4
Feb 11  07:52:50 BFD: Negotiated TX Interval 1000000 microsec,  Jittered Transmi
t Interval 827 ms, random num 3827
Feb 11  07:52:50 BFD: ipc session parameters change to MP, for session 4
Feb 11  07:52:50 BFD: ipc session state change to MP, for session 4
Feb 11  07:52:50 BFD: ipc session parameters change to MP, for session 4
Feb 11  07:52:50 BFD: PBIF TX Assist Request successful for session 4 with inter
val 20675 ms handle 1a480140
Feb 11  07:52:50 BFD: send IPC to MP for Use PBIF Assist change  to Y  for sessi
on 4
Feb 11  07:52:50 BFD: Negotiated TX Interval 50000 microsec,  Jittered Transmit
Interval 40 ms, random num 10731
Feb 11  07:52:50 BFD: Tx PBIF Assist disabled for session 4
Feb 11  07:52:50 BFD: send IPC to MP for Use PBIF Assist change  to N  for sessi
on 4
Feb 11  07:52:50 BFD: PBIF TX Assist Request successful for session 4 with inter
val 1000 ms handle 1a480140
Feb 11  07:52:50 BFD: send IPC to MP for Use PBIF Assist change  to Y  for sessi
on 4
Feb 11  07:52:50 BFD: ipc session parameters change to MP, for session 4
```

Configuration notes

- BFD session establishment on an interface does not start until 90 seconds after the interface comes up. The reason for this delay is to ensure that the link is not affected by unstable link conditions which could cause BFD to flap. This delay time is not user-configurable.
- BFD supports multi-slot trunks in cases where all BFD packets are transmitted only on a single path, which does not change unless the trunk active membership changes. BFD is not supported on multi-slot trunks where per-packet switching is used (where the path taken by the BFD packets varies).
- The BFD Control Message is a UDP message with destination port 3784.
- When you configure BFD, you must set timing and interval parameters.

Common diagnostic scenarios

- It takes a few minutes for BFD to come up after a partner link comes up over Layer 2.

  This is an expected behavior. The Brocade device waits for 90 seconds after a port state changes from down to up before sending the BFD packet out.
- When configuring BFD, one end detects a session but the other end does not.

  The **ip ospf bfd** command must be enabled at the interface level or the **bfd all-interfaces** command must be enabled at the router OSPF level.

BGP

Border Gateway Protocol version 4 (BGP4) is the standard Exterior Gateway Protocol (EGP) used on the Internet to route traffic between Autonomous Systems. BGP maintains a routing table (separate from the main router routing table) of the accessible routes and addresses of Autonomous System neighbors.

BGP show commands

This section describes the show commands that display BGP information.

show debug

Syntax: **show debug**

This command can be used at any time to display debug settings for a device. The following example shows debug settings for a device that has only BGP debug settings enabled.

```
Brocade# show debug
Debug message destination: console
IP Routing
       BGP:bgp debugging is on
       BGP:events debugging is on
       BGP:keepalives debugging is on
       BGP(RED):bgp debugging is on
       BGP(RED):events debugging is on
       BGP(RED):keepalives debugging is on
```

show ip bgp debug

Syntax: show ip bgp debug [**memory** | **network** | **profiling** | **reset-profiling** | **route-table** | **variables** |**out-policy**]

This command displays information about BGP network configurations, including next hops, available memory, profile data, and internal variables.

- **memory** - Displays BGP memory pool information.
- **network** - Displays information about BGP networks.
- **profiling** - Displays profile data on BGP functions.
- **reset-profiling** - Resets the profile data collection setting.
- **route-table** - Displays routing table information.
- **variables** - Displays BGP internal variables.
- **out-policy** - Displays outbound peer policies.

Command output from the **show ip bgp debug** command resembles the following example.

```
Brocade# show ip bgp debug
BGP Debug Information
Pid Size  Address   Total   Used    Free    NoMem  Errors  #_pools p_unit
0   8     021da799  0       0       0       0      0       0       2000
1   16    021da7c5  0       0       0       0      0       0       2000
2   24    021da7f1  0       0       0       0      0       0       2000
3   32    021da81d  910     2       908     0      0       1       800
4   48    021da849  630     1       629     0      0       1       400
5   64    021da875  0       0       0       0      0       0       200
6   96    021da8a1  0       0       0       0      0       0       80
7   128   021da8cd  0       0       0       0      0       0       40
8   256   021da8f9  31      2       29      0      0       1       20
9   48    021da925  5041    5       5036    0      0       1       4000
10  44    021da951  10666   5       10661   0      0       1       8000
11  86    021da97d  5688    7       5681    0      0       1       4000
12  92    021da9a9  5333    4       5329    0      0       1       4000
Total Memory Use for Route and Attributes Tables   : 1871568
  Memory Block Not Available Count   : 0
  Bad Memory Pool ID Count           : 0
  TCP buffers                        : 2048 0 0 0
  BGP Tx Parameters                 : 5 30 3
  BGP route update time counter     : sched: N 0 (100)
  BGP route update count            : 0 (0) last:
                  event             : (1:1) 0.0.0.0/0
  BGP io semaphore take 5, yield 1, 0
  Max timer process: l-0 s-0 (0), io: 0 0 us
  io_rx_yield_time 0x021d9480, 2
  1 sec timer value: 0, 0 TB
  MP active: 1, standby up 0
  Graceful_restart: enable 0, restart time 120, stale-routes 360, purge 600
    Restarted 0, fwd 0, restart_up_time_count[0] 0

BGP inbound/outbound policy caching Enabled

BGP internal debug trace flags:
bgp class data structure is clean
```

show ip bgp debug memory

Syntax: show ip bgp debug memory [**check-as-path** | **check-free** | **dump-used**]

- **check-as-path** - Checks AS paths.
- **check-free** - Checks free pool entry.
- **dump-used** - Dumps used pool entry.

This command displays information about the internal sizes of various BGP entities. It shows a summary of how many chunks were allocated and freed for each pool, as the example that follows illustrates.

```
Brocade# show ip bgp debug memory
BGP_CLASS_VRF: 26367, BGP_PEER_CLASS: 179124, BGP_CONFIGURATION_CLASS: 4135
BGP_AS_PATH_ENTRY: 94, BGP_IPV6_AS_PATH_ENTRY: 106, BGP_AS_PATH_SEGMENTS: 18
BGP_NLRI_ENTRY: 86, BGP_PATRICIA_KEY_ADDRESS: 16, BGP_PATRICIA_NODE: 48
BGP_RIB_OUT_NLRI_ENTRY: 44, BGP_RIB_OUT_HOLDER: 28, BGP_WITHDRAWN_ROUTE_ENTRY: 42
BGP_NEXTHOP_ADDRESS: 16, BGP_NEXTHOP_ENTRY: 199, BGP_DAMPING_NLRI_ENTRY: 36
BGP_ROUTE_DAMPING_REUSE_LIST: 2052, BGP_ROUTE_DAMPING_BLOCK: 3522,
BGP_DAMPENING:37374
BGP pool 0 allocated 1 chunk freed 0 chunk
BGP pool 1 allocated 0 chunk freed 0 chunk
BGP pool 2 allocated 0 chunk freed 0 chunk
BGP pool 3 allocated 1 chunk freed 0 chunk
BGP pool 4 allocated 1 chunk freed 0 chunk
BGP pool 5 allocated 1 chunk freed 0 chunk
BGP pool 6 allocated 0 chunk freed 0 chunk
BGP pool 7 allocated 0 chunk freed 0 chunk
BGP pool 8 allocated 0 chunk freed 0 chunk
BGP pool 9 allocated 1 chunk freed 0 chunk
BGP pool 10 allocated 1 chunk freed 0 chunk
BGP pool 11 allocated 1 chunk freed 0 chunk
BGP pool 12 allocated 1 chunk freed 0 chunk
BGP pool 13 allocated 0 chunk freed 0 chunk
BGP pool 14 allocated 0 chunk freed 0 chunk
BGP pool 15 allocated 1 chunk freed 0 chunk
BGP pool 16 allocated 0 chunk freed 0 chunk
BGP pool 17 allocated 0 chunk freed 0 chunk
```

show ip bgp debug memory check-free

Syntax: show ip bgp debug memory check-free *pool-id*

This command shows the number of free entries in the memory pool. Command output is similar to the following example (in this case, memory pool 1).

```
Brocade# show ip bgp debug memory check-free 1
Number of free entry in the pool(1) = 0, expected 0
```

show ip bgp debug memory dump-used

Syntax: show ip bgp debug memory dump-used *pool-id*

This command displays (in page mode) all memory chunks that are held by a pool (either fully used or still with free entries), as the following example illustrates.

```
Brocade# show ip bgp debug memory dump-used 10
     Pool 10: memory chunk that had free entry:
chunk addr: 2827c000, total entry 2520, free 2405, used 115
     Pool 10: memory chunk that had all entries in use:
```

show ip bgp debug network

Syntax: show ip bgp debug network [*X:X::X:X* | *A.B.C.D or A.B.C.D/L*]

- *X:X::X:X* - IPv6 network address.
- *A.B.C.D or A.B.C.D/L* - IP network address.

This command displays internal BGP information about network command-related data structures. Command output resembles the following example (shown for an IP network address).

```
Brocade# show ip bgp debug network 10.2.2.2/32
BGP: network 10.2.2.2/32 found
(x05236ebc, 00000000, 00000000) 10.2.2.2/32
   weight:32768 back_door:0 imported:0
   route-map:<>  sptr:x00000000
   next_hop:0.0.0.0 med:0 type:0
```

show ip bgp debug profiling

Syntax: show ip bgp debug profiling

This command displays BGP profiling information, as shown in the following example.

```
Brocade# show ip bgp debug profiling
BGP Profiling Data:
Send                            TOTAL=0                 FREQ=0          AVG=0
SendRibOut                      TOTAL=0                 FREQ=0          AVG=0
SendRibOutAddAS                 TOTAL=0                 FREQ=0          AVG=0
SendRibOutAddNLRI               TOTAL=0                 FREQ=0          AVG=0
Recv                            TOTAL=0                 FREQ=0          AVG=0
RecvASAlloc                     TOTAL=0                 FREQ=0          AVG=0
RecvUpd                         TOTAL=0                 FREQ=0          AVG=0
RecvAschk                       TOTAL=0                 FREQ=0          AVG=0
RecvAsloop                      TOTAL=0                 FREQ=0          AVG=0
RecvAsadd1                      TOTAL=0                 FREQ=0          AVG=0
RecvAsadd2                      TOTAL=0                 FREQ=0          AVG=0
RecvUpdnlri                     TOTAL=0                 FREQ=0          AVG=0
RecvUpdvpnv4nlri                TOTAL=0                 FREQ=0          AVG=0
RecvVpnv4copy                   TOTAL=0                 FREQ=0          AVG=0
RecvVpnv4alloc                  TOTAL=0                 FREQ=0          AVG=0
RecvVpnv4ribin                  TOTAL=0                 FREQ=0          AVG=0
RecvVpnv4process                TOTAL=0                 FREQ=0          AVG=0
RecvVpnv4processImp             TOTAL=0                 FREQ=0          AVG=0
RecvVpnv4processnewas           TOTAL=0                 FREQ=0          AVG=0
RecvVpnv4processasadd           TOTAL=0                 FREQ=0          AVG=0
RecvRibinTree2                  TOTAL=0                 FREQ=0          AVG=0
RecvRibinTree3                  TOTAL=0                 FREQ=0          AVG=0
RecvRibinTree4                  TOTAL=0                 FREQ=0          AVG=0
AllocAPool1                     TOTAL=0                 FREQ=0          AVG=0
AllocAPool2                     TOTAL=0                 FREQ=0          AVG=0
AllocPoolDymalloc               TOTAL=219               FREQ=3          AVG=73
AllocAPoolChain                 TOTAL=5592              FREQ=3          AVG=1864
vpnv4_ribout_add_nlri            TOTAL=0                FREQ=0          AVG=0
vpnv4_ribout_allocate_label      TOTAL=0                FREQ=0          AVG=0
vpnv4_ribout_run_policy          TOTAL=0                FREQ=0          AVG=0
```

```
vpnv4_ribout_rem_withd_rt        TOTAL=0                 FREQ=0          AVG=0
rp_match_total                  TOTAL=0                 FREQ=0          AVG=0
rp_set_total                    TOTAL=0                 FREQ=0          AVG=0
rp_match_access                 TOTAL=0                 FREQ=0          AVG=0
bgp_check_update                TOTAL=0                 FREQ=0          AVG=0
Update_Chk_Nexthop              TOTAL=0                 FREQ=0          AVG=0
Update_Add_Routes               TOTAL=0                 FREQ=0          AVG=0
Update_Tail                     TOTAL=0                 FREQ=0          AVG=0
Chk_NextHop_Change_Def          TOTAL=0                 FREQ=0          AVG=0
Chk_NextHop_Change_Hash         TOTAL=0                 FREQ=0          AVG=0
Chk_NextHop_Change_Hash         TOTAL=0                 FREQ=0          AVG=0
Chk_NextHop_Change_Lookup       TOTAL=0                 FREQ=0          AVG=0
Chk_NextHop_Change_Process      TOTAL=0                 FREQ=0          AVG=0
Revert_Idle_State_Delete_All    TOTAL=0                 FREQ=0          AVG=0
Revert_Idle_State_Tail          TOTAL=0                 FREQ=0          AVG=0
RIB_in_delete_all_nlris_from_p   TOTAL=0                 FREQ=0          AVG=0
Timer_Add_Routes                TOTAL=0                 FREQ=0          AVG=0
Timer_Delete_All_Nlri           TOTAL=0                 FREQ=0          AVG=0
Add_routes                      TOTAL=0                 FREQ=0          AVG=0
Add_routes_cbk_nlri_list        TOTAL=0                 FREQ=0          AVG=0
Add_routes_cbk_update_ip        TOTAL=0                 FREQ=0          AVG=0
check_and_update_bgp_route      TOTAL=0                 FREQ=0          AVG=0
```

show ip bgp debug route-table

Syntax: **show ip bgp debug route-table**

This command displays the Network Layer Reachability Information (NLRI) count in the BGP route table, as shown in the following example.

```
Brocade# show ip bgp debug route-table
There are 7 NLRIs in BGP Route Table, time 0 ms
```

show ip bgp debug variables

Syntax: **show ip bgp debug variables**

This command displays detailed BGP information about internal flags, statistics, and states. Command output resembles the following example.

```
Brocade# show ip bgp debug variables
safi:0, &bgp:04d2dfdc, enabled:1, operational:1, dbg_mem=&021da72e/0, curr_afi:0
io_process_running:0, io_process_next_peer_number=1
in_long_loops 0, clear_all 0, timer 00000000, count 0
timer_enabled:1, timer_next_peer_number:0, 1s timer 1, short timer 1
scheduler id:1:1, ip:0.0.0.0/0, time=16977
bgp_tcb:021cae08 (0x0001001f, 0), tick_cnt=17, seconds=691
bgp_tcb6:0001002f (0x00000000, 0x04d2e0c4)
*peer:021d9546, *peer_group:021da4f2, RIB_in_root_node:0bb373b4
Maximum Peer Index Number:2, check_nexthops:0 0
router_id:10.3.3.3, configured:0, cluster_id:0.0.0.0, configured:0
route_is_router_reflector:0, client_to_client_reflection:1
networks:x021cdbe9, aggregate:x021cdc0d
default_metric:4294967294, local_preference:100, keep_alive:60, hold_time:180
originate_default:0, originated:0
distance:20 200 200, fast_external_fallover=0
nexthop recur0, en_def:0, readvertise:1, auto_sum:0, synch:0
always_compare_med:0, compare_med_with_empty_aspath: 0, redistribute_ibgp:0,
local_network_check_time_count:1
```

```
nexthop_cache_hit_count:6, nexthop_cache_miss_count:2
system memory:536870912, total_allocated:1964468, bgp_defined_quota:2147483648
import map:"", export map:""
nexthop_lb_interface:<no-such-port>, nexthop_lb_addr:0.0.0.0
```

show ip bgp debug out-policy

Syntax: show ip bgp debug out-policy

This command displays outbound peer policies. Command output resembles the following example.

```
Brocade# show ip bgp debug out-policy
BGP(vrf 0/safi 0) outbound policy entries: 13
Outbound Policy Group: 0x34731a00 (Hash 0), ID: 1, Drop 1, Use Count: 0, Staring:
0, Update: 0
Ribout Group: 0x34831000, ID: 1, Type: -1, Peer Count: 0, Mask: 0x00000000 (0),
ribout: 0, withdrawn: 0
Outbound Policy Group: 0x347a5600 (Hash 0), ID: 9, Drop 0, Use Count: 1, Staring:
42, Update: 0
Ribout Group: 0x35563000, ID: 2, Type: 1, Peer Count: 1, Mask: 0x00000004 (2),
ribout: 2102, withdrawn: 0
Outbound Policy Group: 0x355e9e00 (Hash 31), ID: 5, Drop 0, Use Count: 6, Staring:
18, Update: 0
Ribout Group: 0x35615000, ID: 6, Type: 2, Peer Count: 6, Mask: 0x0000003f (5),
ribout: 175020, withdrawn: 0
routemap: map10
Outbound Policy Group: 0x355e9200 (Hash 110), ID: 2, Drop 0, Use Count: 6,
Staring: 0, Update: 0
Ribout Group: 0x355f1000, ID: 3, Type: 2, Peer Count: 6, Mask: 0x0000003f (5),
ribout: 175020, withdrawn: 0
routemap: map0
Outbound Policy Group: 0x347a5400 (Hash 111), ID: 8, Drop 0, Use Count: 6,
Staring: 36, Update: 0
Ribout Group: 0x35627000, ID: 9, Type: 2, Peer Count: 6, Mask: 0x0000003f (5),
ribout: 175020, withdrawn: 0
routemap: map1
```

show ip bgp debug out-policy peer-list

Syntax: show ip bgp debug out-policy peer-list

This command displays the peer grouping. The Brocade device groups BGP peers together based on their outbound policies. To reduce RIB-out memory usage, the device then groups the peers within an outbound policy group according to their RIB-out routes. Peers in a group have the same Policy-ID and Group-ID.

RIB-out peer grouping is not shared among different VRFs or address families, and is not supported for VPNv4 or Layer 2 VPN peers.

```
Brocade# show ip bgp debug out-policy peer-list
BGP sorted peers on outbound policy (safi=0)
Index ->   PeerIdx    Peer Address  Policy-ID   Group-ID Vrf-idx
0          0          10.0.100.2    2           3        0
1          1          10.0.101.2    2           3        0
2          2          10.0.102.2    2           3        0
3          3          10.0.103.2    2           3        0
4          4          10.0.104.2    2           3        0
5          5          10.0.105.2    2           3        0
6          18         10.0.100.2    3           4        0
```

```
7          19         10.0.101.2    3          4         0
8          20         10.0.102.2    3          4         0
```

show ip bgp nexthop

Syntax: **show ip bgp nexthop** [*ip-address* | **unreachable**]

- *ip-address* - Displays the next hop with the specified IP address.
- **unreachable** - Displays only the unreachable next hops.

This command displays information about IPv4 next hops. The information includes next hops, route type, cost, resolve schema, next hop router IP address, and outgoing interface. Command output resembles the following example.

```
Brocade# show ip bgp nexthop
safi : BGP UNICAST SAFI
Type Codes - B:BGP D:Connected I:ISIS O:OSPF R:RIP S:Static
OSPF Codes - i:Internal 1:External Type 1 2:External Type 2

Next Hop          RouteType  Cost   ResolveSchema     Router IP     Port
10.122.122.122     Oi        2      IPv4/6            10.32.32.2    eth 1/3
                                                      10.42.42.2    eth 1/4
10.132.10.10      Oi         2      IPv4/6            10.22.22.10   eth 1/2
```

show ipv6 bgp nexthop

Syntax: **show ipv6 bgp nexthop** [*ipv6-address* | **unreachable**]

- *ipv6-address* - Displays the next hop with the specified IPv6 address.
- **unreachable** - Displays only the unreachable next hops.

This command displays information about IPv6 next hops. The information includes next hops, route type, cost, resolve schema, next hop router IPv6 address, and outgoing interface. Command output resembles the following example.

```
Brocade# show ipv6 bgp nexthop
safi : BGP IPV6 UNICAST SAFI
Type Codes - B:BGP D:Connected I:ISIS O:OSPF R:RIP S:Static
OSPF Codes - i:Internal 1:External Type 1 2:External Type 2

Next Hop     RouteType Cost  ResolveSchema     Router IP               Port
2001:DB8::9  Oi       2     IPv4/6            fe80::224:38ff:fe43:6d41 eth 1/1
                                               fe80::224:38ff:fe43:6d42 eth 1/2
2001:DB8::20 Oi       2     IPv4/6            fe80::224:38ff:fe43:6d41 eth 1/1
                                               fe80::224:38ff:fe43:6d42 eth 1/2
```

show ip rtm

Syntax: **show ip rtm** [**vrf** *vrf name*] | **include bgp**

- **vrf** *vrf name* - Specifies that you want to display the RTM information for the specified VRF.
- **include bgp** - Includes the matching lines from BGP output.

This command displays information about the Routing Table Manager (RTM) BGP rib count, as shown in the following example.

```
Brocade# show ip rtm vrf vpn1 | include bgp
client bgp (0x105b1104):
bgp route limit 0, current 0
```

BGP debug commands

This section describes the debug commands used for monitoring the BGP environment.

debug ip bgp

Syntax: [**no**] **debug ip bgp** [**all-vrfs** | *vrf-name*] [**dampening** | **events** | **general** | **graceful-restart** | **keepalives** | **updates** [**rx** | **tx**] | **route-selection**]

- **all-vrfs** - Displays information for all virtual routing and forwarding events.
- *vrf-name* - Displays information for a specific VRF event.
- **dampening** - Displays BGP dampening activity.
- **events** - Displays BGP event information.
- **general** - Displays BGP common debugs.
- **graceful-restart** - Displays information about graceful-restart events.
- **keepalives** - Displays keepalive activity.
- **updates** [**rx** | **tx**] - Displays BGP receive, transmit, or receive and transmit update messages about debug processing.
- **route-selection** - Displays BGP route selection debug information.

This command enables common BGP debugs to be displayed for all VRFs or for a specific VRF. Command output resembles the following example.

```
Brocade# debug ip bgp
/**** local-as of peer has changed ****/
BGP: 10.1.1.2 Rcv TCP connection closed remotely. handle 00000005:0a0132d4, key 0
BGP: 10.1.1.2 remote peer closed TCP connection
BGP: 10.1.1.2 rcv notification: CEASE Message
BGP: 10.1.1.2 BGP connection closed

BGP: 10.1.1.2 start peer
BGP: 10.1.1.2 Init TCP Connection to peer, local IP 10.1.1.1
BGP: 10.1.1.2 Rcv TCP connection closed remotely. handle 0000000 6:0a0132d4, key 0
BGP: 10.1.1.2 TCP connection failed
BGP: Rcv incoming TCP connection check. handle 00000007:0a0123d4, key 0
BGP: Incoming TCP connection. peer 10.1.1.2 OKed
BGP: Rcv incoming TCP connection UP. handle 00000007:0a0123d4, key 0
BGP: 10.1.1.2 New incoming TCP connection is open, local IP 10.1.1.1
BGP: 10.1.1.2 sending MultiProcotol cap, afi/safi=1/1, length 4
BGP: 10.1.1.2 sending IPEN, holdTime=180 route_refresh=1 cooperative= 1, restart
0/0
BGP: 10.1.1.2 rcv OPEN w/Option parameter length 16, as 2. hold_time 180
BGP: 10.1.1.2 rcv capability 2, len 0
BGP: 10.1.1.2 rcv capability 128, len 0
```

debug ip bgp route-selection

Syntax: [**no**] **debug ip bgp** [**all-vrfs** | *vrf-name*] **route-selection**

This command enables debugging of BGP route selection for all VRFs or for a specified VRF. Command output resembles the following example.

```
Brocade# debug ip bgp route-selection
```

```
BGP: Clearing install flags for 2001:DB8:0:61::/64
BGP: 2001:DB8::20 Move path to front for 2001:DB8:0:51::/64
BGP: select best route 2001:DB8:0:61::/64 load_share (ibgp 1, ip 1), (ebgp 1, ip
1)
BGP: eligible route 1
BGP: 2001:DB8::20 Best path up 2001:DB8:0:61::/64, install
BGP: 2001:DB8::20 Move path to front for 2001:DB8:0:61::/64
BGP: add bgp routes to IPv6 table 2001:DB8:0:61::/64
BGP: Adding 2001:DB8:0:61::/64 to ipv6 route table, next_hop=2001:DB8::20
BGP: Clearing install flags for 2001:DB8:0:61::/64
BGP: 2001:DB8::20 Move path to front for 2001:DB8:0:51::/64
BGP: select best route 2001:DB8:0:61::/64 load_share (ibgp 1, ip 1), (ebgp 1, ip
1)
BGP: eligible route 1
BGP: 2001:DB8::20 Best path up 2001:DB8:0:61::/64, install
BGP: 2001:DB8::20 Move path to front for 2001:DB8:0:61::/64
BGP: add bgp routes to IPv6 table 2001:DB8:0:61::/64
BGP: Adding 2001:DB8:0:61::/64 to ipv6 route table, next_hop=2001:DB8::20
```

debug ip bgp dampening

Syntax: [**no**] **debug ip bgp** [**all-vrfs** | *vrf-name*] **dampening**

This command displays information about dampening process configurations, route penalties, durations, restraint, and release. Command output resembles the following example.

```
Brocade# debug ip bgp dampening
BGP: dampening debugging is on
BGP: 2001:DB8::20 dampening 2001:DB8::/64 down, penalty=9154 flaps=572
BGP: 2001:DB8::20 dampening 2001:DB8::/64 up, penalty=8640 suppressed
```

In this example, dampening is enforced on route 2001:DB8::/64 due to flap activity, and lifted after a period of time.

debug ip bgp events

Syntax: [**no**] **debug ip bgp** [**all-vrfs** | *vrf-name*] **events**

This command generates information about BGP events, such as connection attempts and keepalive timer activity, as shown in the following example.

```
Brocade# debug ip bgp events
   BGP: events debugging is on
BGP: From Peer 192.168.1.2 received Long AS_PATH=
AS_CONFED_SET(4) 1 2 3 AS_CONFED_SEQUENCE(3) 4 AS_SET(1)
5 6 7 AS_SEQ(2) 8 9 attribute length (9) More than configured MAXASLIMIT 7
```

The following line of output indicates a four-byte ASN.

```
Sep  9 18:36:42 BGP: 192.168.1.1 rcv capability 65, len 4
```

debug ip bgp graceful-restart

Syntax: [**no**] **debug ip bgp** [**all-vrfs** | *vrf-name*] **graceful-restart**

Enable this command to receive information about BGP graceful restarts. The graceful restart feature minimizes disruptions in forwarding and route flapping when a router experiences a restart. Command output resembles the following example.

```
Brocade# debug ip bgp graceful-restart
         BGP:  graceful-restart debugging is on
SYSLOG: <13>Dec 10 22:46:50 R3 BGP: Peer 10.1.1.2 DOWN (User Reset Peer Session)
```

```
Dec 10 22:46:50 BGP: 10.1.1.2 delete NLRI #RIB_out 6,  (safi 0)
Dec 10 22:46:50 BGP: 10.1.1.2 RIB_out peer reset  #RIB_out 6 (safi 0)

SYSLOG: <13>Dec 10 22:46:59 R3 BGP: Peer 10.1.1.2 UP (ESTABLISHED)
Dec 10 22:47:01 BGP: 10.1.1.2 rcv UPDATE EOR (0), waiting EOR 0
Dec 10 22:47:01 BGP: 10.1.1.2 sending EOR (safi 0)...
Dec 10 22:47:01  BGP: 10.1.1.2 sending UPDATE EOR[0]
```

debug ip bgp keepalives

Syntax: [**no**] **debug ip bgp** [**all-vrfs** | *vrf-name*] **keepalives**

Brocade devices use keepalives to collect information about applications and services. For example, you can configure a keepalive to continually monitor and report on the online status of a resource, such as BGP. Command output resembles the following example.

```
Brocade# debug ip bgp keepalives
BGP:  keepalives debugging is on
BGP: 10.28.156.234 KEEPALIVE received
BGP: 10.142.72.222 KEEPALIVE received
BGP: 10.28.148.234 KEEPALIVE received
BGP: 10.142.72.222 sending KEEPALIVE
BGP: 10.28.156.234 sending KEEPALIVE
```

debug ip bgp neighbor

Syntax: [**no**] **debug ip bgp** [**all-vrfs** | *vrf-name*] **neighbor** [*ipv4-address* | *ipv6-address*]

This command specifies the IPv4 or IPv6 neighbor filter for BGP debugging for all VRFs or for a specified VRF. Only one IPv4 neighbor filter and one IPv6 neighbor filter can be configured on each VRF.

The neighbor filter acts exclusively, which means that wherever an IPv4 neighbor filter is configured, the corresponding IPv6 debugs are not shown. The neighbor filter functions with the **updates [rx|tx]**, **keepalives**, **events,** and **graceful-restart** keywords of the **debug ip bgp** command. For example, when **debug ip bgp updates** and **debug ip bgp neighbor** 2001:DB8::20 are configured, BGP update debugs are displayed only for neighbor 2001:DB8::20.

Command output resembles the following example.

```
Brocade# debug ip bgp neighbor 2001:DB8::20
   BGP:  neighbor 2001:DB8::20 debugging is on
Brocade# debug ip bgp updates rx
   BGP:  updates RX debugging is on
BGP: 2001:DB8::20 rcv UPDATE 2001:DB8::/64 -- withdrawn
BGP: rcv UPDATE w/attr: Origin=IGP AS_PATH= AS_SEQ(2) 65400 65181 209 7018
NextHop=2001:DB8::20
```

debug ip bgp ip-prefix

Syntax: [**no**] **debug ip bgp** [**all-vrfs** | *vrf-name*] **ip-prefix** *ip prefix/mask*

debug ip bgp ipv6-prefix

Syntax: [**no**] **debug ip bgp** [**all-vrfs** | *vrf-name*] **ipv6-prefix** ipv6 prefix/mask

debug ip bgp ip-prefix-list

Syntax: [**no**]**debug ip bgp** [**all-vrfs** |*vrf-name*] **ip-prefix-list** *name*

debug ip bgp ipv6-prefix-list

Syntax: [**no**] **debug ip bgp** [**all-vrfs** | *vrf-name*] **ipv6-prefix-list** *name*

These commands specify the IPv4 or IPv6 prefix filter for BGP debugging information for all VRFs or for a specified VRF. Only one IPv4 prefix filter or prefix list and one IPv6 prefix filter or prefix list can be configured on a VRF. Prefix filters and prefix lists cannot be configured simultaneously. IPv6 and IPv4 filters are applied separately. Configuring an IPv4 prefix list or prefix filter does not automatically block IPv6 debugging, unlike the neighbor filter. To block IPv6 debugging, AFI/SAFI must be configured along with the prefix filter.

NOTE
Prefix filtering is not functional during update receive processing for VPNv4. Prefix filtering is functional with the **updates** [**rx**/**tx**], **route-selection,** and **dampening** keywords of the **debug ip bgp** command.

NOTE
Removing the configured **ip-prefix-list** *name* or **ipv6-prefix-list** *name* also automatically removes the associated filter.

Output from any of these commands resembles the following example, where the first example reflects an IPv4 prefix filter, or a similar IPv4 prefix list along with AFI/SAFI as IPv4 and unicast.

```
Brocade# debug ip bgp ip-prefix 10.1.1.1/24
         BGP:  ip-prefix debugging is on
                         permit 10.1.1.0/24

/**** Route-Addition ****/
Sep  9 18:38:13 BGP: BGP rcv UPDATE w/attr: Origin=IGP AS_PATH= AS_SEQ(2) 2
NextHop=10.1.1.2 MED=0
Sep  9 18:38:13 BGP: (0): 10.1.1.2 rcv UPDATE 10.1.1.0/24

/**** Route-Deletion ****/
Sep  9 18:38:24 BGP: 10.1.1.2 rcv UPDATE 10.1.1.0/24 - withdrawn
```

The next example reflects an IPv6 prefix filter, or a similar IPv6 prefix list along with AFI/SAFI as IPv6 and unicast.

```
Brocade# debug ip bgp ipv6-prefix 2001:DB8::1/64
         BGP:  ipv6-prefix debugging is on
                         permit 2001:DB8::/64
/**** Route-Addition ****/
Sep  9 24:01:32 BGP: rcv UPDATE w/attr: Origin=IGP AS_PATH= AS_SEQ(2) 65400 65181
209 7018 NextHop=21:1::20
Sep  9 24:01:32 BGP: (2): 2001:DB8::20 rcv UPDATE 2001:DB8::/64

/**** Route-Deletion ****/
Sep  9 24:01:40 BGP: 2001:DB8::20 rcv UPDATE 2001:DB8::/64 -- withdrawn
```

debug ip bgp route-map

Syntax: [**no**] **debug ip bgp** [**all-vrfs** |*vrf-name*] **route-map** *map-name*

This command associates an existing route map filter with BGP debugging. The route map filter is functional with the **route-selection** and **dampening** keywords of the **debug ip bgp** command.

CAUTION

This command may degrade performance.

NOTE
Removing the configured route map automatically removes the associated filter.

NOTE
To debug the entire update receive process until route selection and route updates correspond to the update, you must enable the **debug ip bgp updates rx**, and **debug ip bgp route-selection** commands along with the appropriate filters. In this case, all of the filters (NBR+AFI/SAFI+PREFIX+ROUTE_MAP) are applied if they are configured.

---

debug ip bgp route-updates

Syntax: [**no**] **debug ip bgp route-updates**

This command shows the routes that have been shared with a neighbor. The route information includes the four-byte AS4_PATH attribute and the AS_PATH attribute.

Command output resembles the following example.

```
Brocade# debug ip bgp route-updates
Sep  9 18:41:59 BGP: BGP: 192.168.1.1 rcv UPDATE w/attr: Origin=INCOMP AS_PATH=
AS_SEQ(2) 90000 70001 70002 70003 75000 NextHop=192.168.1.5
Sep  9 18:41:59 BGP: BGP: 192.168.1.1 rcv UPDATE w/attr: Origin=INCOMP AS4_PATH=
AS_SEQ(2) 90000 70001 70002 70003 75000 NextHop=192.168.1.5
```

debug ip bgp updates

Syntax: [**no**] **debug ip bgp** [**all-vrfs** | *vrf-name*] **updates** [**rx** | **tx**]

This command enables debugging of BGP update message processing for all VRFs or for a specific VRF. Update debugging is supported in both IPv4 and IPv6 update message processing. If you do not specify either all VRFs or a specific VRF, the **debug ip bgp updates** command enables debugging for both receive and transmit update message processing.

Command output resembles the following example.

```
Brocade# debug ip bgp updates
Sep 9 18:38:13 BGP: BGP rcv UPDATE w/attr: Origin=IGP AS_PATH= AS_SEQ(2) 2
NextHop=10.1.1.2 MED=0
BGP: (0): 10.1.1.2 rcv UPDATE 10.1.1.0/24
BGP: 10.1.1.2 rcv UPDATE 10.1.1.0/24 - withdrawn
BGP: rcv UPDATE w/attr: Origin=IGP AS_PATH= AS_SEQ(2) 65400 65181 209 7018
NextHop=21:1::20
BGP: (2): 2001:DB8::20 rcv UPDATE 2001:DB8::/64
BGP: 2001:DB8::20 rcv UPDATE 2001:DB8::/64 -- withdrawn
```

BFD for BGP4 debug commands

To debug Bidirectional Forwarding Detection (BFD) for BGP4, use the following debugging commands:

- **debug ip bgp** [**all-vrfs** | **vrf** *vrf-name*] **bfd**

- **debug bfd application** (Refer to this command on page 214.)
- **debug bfd itc** (Refer to this command on page 217.)

debug ip bgp bfd

Syntax: debug ip bgp [**all-vrfs** | **vrf** *vrf-name*] **bfd**

- **all-vrfs** - Displays information for all BGP BFD events.
- **vrf** *vrf-name* - Displays information for a specific VRF event.

Command output resembles the following example.

```
Brocade# debug ip bgp bfd
Sep 9 18:37:07 BFD:ITC, Received BFD MHOP ITC Create S
ession Request from bgp(0)
Sep 9 18:37:07 BFD: BFD MHOP ITC Create Session Response Sent to bgp(0)
Sep 9 18:37:07 BGP: 10.1.1.2 Peer Received BFD MHOP Create Session Respo
nse ITC message
Sep 9 18:37:07 BFD: BFD MHOP ITC Update Session Negotiated Parameters Request S
ent to bgp(0)
Sep 9 18:37:07 BGP: 10.1.1.2 Peer Received BFD MHOP BFD Session State Ch
ange Notify ITC message
Sep 9 18:37:07 BGP: 10.1.1.2 Peer Received BFD session UP state notification
Sep 9 18:37:07 BGP: 10.1.1.2 Peer BGP-BFD state changed to UP
Sep 9 18:37:07 BGP: 10.1.1.2 Peer Received BFD MHOP Update Session Negot
Sep 9 18:37:14 BFD:ITC, Received BFD MHOP ITC Route Change Indication from bgp(0)
```

IPv6 ND debug commands

This section describes the IPv6 ND-related debug commands.

debug ipv6 nd

Syntax: [**no**] **debug ipv6 nd**

This command enables debugging for all neighbors. To enable debugging for a specific neighbor, enter the **debug ipv6 nd** command followed by the **debug ipv6 address** *X.X::X.X* command.

Command output resembles the following example.

```
Brocade# debug ipv6 nd
IPv6 Generic:
          IPv6:  nd debugging is on
Debug message destination:  Console
Dec 10 15:04:10 ICMPv6-ND: Received NS for fe80::21b:edff:fe19:692 on 4/3 (4/3)
from fe80::21b:edff:fe17:6632
Dec 10 15:04:10 ICMPv6-ND: New neighbor entry fe80::21b:edff:fe17:6632 on port
4/3, Link-Addr 0000.0017.6632
Dec 10 15:04:10 ICMPv6-ND: INCOMP->STALE: fe80::21b:edff:fe17:6632 on 4/3
Dec 10 15:04:10 ICMPv6-ND: Sending NA for fe80::21b:edff:fe17:6632 on 4/3
Dec 10 15:04:10 ICMPv6-ND: STALE->DELAY: fe80::21b:edff:fe17:6632 on 4/3 (Resolve
Nbr)
Dec 10 15:04:10 ICMPv6-ND: adding neighbor to request list
fe80::21b:edff:fe17:6632
Dec 10 15:04:15 ICMPv6-ND: Sending NS for fe80::21b:edff:fe17:6632 on port 4/3,
dest fe80::21b:edff:fe17:6632
Dec 10 15:04:15 ICMPv6-ND: DELAY->PROBE: fe80::21b:edff:fe17:6632 on 4/3
Dec 10 15:04:15 ICMPv6-ND: Received NA for fe80::21b:edff:fe17:6632 on port 4/3
from fe80::21b:edff:fe17:6632
```

```
Dec 10 15:04:15 ICMPv6-ND: PROBE->REACH: fe80::21b:edff:fe17:6632 on 4/3
Dec 10 15:04:15 ICMPv6-ND: removing neighbor from request list
fe80::21b:edff:fe17:6632 (#6)
```

Configuration notes

- If you configure the Brocade NetIron XMR and Brocade MLX series router to use a loopback interface to communicate with a BGP4 neighbor, you must also configure a loopback interface on the neighbor, and configure the neighbor to use that loopback interface to communicate with the router.
- Brocade NetIron XMR and Brocade MLX series routers use the same router ID for both OSPF and BGP4. If the router is already configured for OSPF, you may want to use the router ID that is already in use rather than set a new one. To display the router ID, enter the **show ip** command at any CLI level.
- The command to set the router ID for a specified VRF is the same as the command for the default VRF. The only difference is that when setting it for a specific VRF, the **ip router-id** command is configured within the VRF.
- When setting the router ID, you can specify an IP address that is being used for an interface, but do not specify an IP address that is in use by another device.
- The OSPF Stub Router Advertisement feature is helpful for preventing a loss of traffic during short periods when adjacency failures are detected and traffic is rerouted. With this feature, traffic can be rerouted before an adjacency failure occurs (due to common service interruptions such as a router being shut down for maintenance). This feature is also useful during router startup because it gives the router enough time to build up the routing table before forwarding traffic. This is helpful where BGP is enabled because it takes time for the BGP routing table to converge.
- Under normal operation, restarting a BGP router reconfigures the network. In this situation, routes available through the restarting router are deleted when the router goes down and are then rediscovered and added back to the routing tables when the router comes back up. In a network where routers are regularly restarted, performance may be significantly degraded, limiting the availability of network resources. The BGP Graceful Restart feature dampens the network response and limits route flapping by allowing routes to remain available during a restart. BGP graceful restart operates between a router and its peers and must be configured on all devices.
- When you enable BGP4, you do not need to reset the system. The protocol is activated as soon as it is enabled. The router also begins a BGP4 session with a BGP4 neighbor as soon as you add the neighbor.
- The router attempts to establish a BGP4 session with a neighbor as soon as you enter a command specifying the neighbor IP address. To completely configure neighbor parameters before the router can establish a session, you can administratively shut down the neighbor.

Disabling BGP4

If you disable BGP4, the router removes all configuration information for the disabled protocol from the running configuration. To restore the BGP4 configuration, you must reload the software, which loads the configuration from the startup configuration. When you save the configuration to the startup configuration file after disabling the protocol, all configuration information for the disabled protocol is removed from the startup configuration file. You will see a CLI error message similar to the following example.

```
Brocade(config)# no router bgp
router bgp mode now disabled. All bgp configuration data will be lost when writing
to flash?
```

When you test a BGP4 configuration, and are likely to disable and re-enable the protocol, you might want to make a backup of the startup configuration file that contains the BGP4 configuration information. If you remove the configuration information by saving the configuration after you disable the protocol, you can then restore the configuration by copying the backup to the flash memory.

To disable BGP4 without losing the BGP4 configuration information, remove the local AS (for example, by entering the **no local-as** *num* command). In this case, BGP4 retains the other configuration information but is not operational until you again set the local AS.

Forwarding disruptions and port flapping

The BGP Graceful Restart feature supports high-availability routing. With this feature enabled, disruptions in forwarding are minimized and route flapping diminished to provide continuous service during times when a router experiences a restart. For more information about BGP Graceful Restart, refer to the *Brocade NetIron Routing Configuration Guide*.

Performance degrades during restarts

BGP Graceful Restart also helps minimize performance degradation during restarts. For more information about BGP Graceful Restart, refer to the *Brocade NetIron Routing Configuration Guide.*

Reducing the complexity of BGP configurations

The following information can help to simplify your BGP configurations.

Confederations

A *confederation* is a BGP4 Autonomous System (AS) that has been subdivided into multiple, smaller autonomous systems. Subdividing an autonomous system into smaller autonomous systems simplifies administration and reduces BGP-related traffic, thus reducing the complexity of the Interior Border Gateway Protocol (IBGP) mesh among the BGP routers in the AS.

Router reflection

Another way to reduce the complexity of an IBGP mesh is to use route reflection. However, if you want to run different Interior Gateway Protocols (IGPs) within an AS, configure a confederation. You can run a separate IGP within each sub-AS.

For more information on how to configure these features, refer to the *Brocade NetIron Routing Configuration Guide*.

BGP memory considerations

BGP4 handles a very large number of routes and therefore requires a substantial amount of memory. For example, in a typical configuration with a single BGP4 neighbor, a BGP4 router may need to manage as many as 250,000 routes. Many configurations, especially those involving more than one neighbor may require the router to retain even more routes. The Brocade devices provide dynamic memory allocation for BGP4 data, automatically allocating memory when needed to support BGP4 neighbors, routes, and route attribute entries. Dynamic memory allocation is performed automatically by the software and does not require a reload.

Common diagnostic scenarios

MD5 Authentication is incorrect between two BGP peers when a BGP session remains connected. Because MD5 authentication occurs at the TCP layer, the **debug ip bgp** command will not help in this situation. To determine the exact error, issue the **debug ip tcp transactions** command.

OSPF

The Open Shortest Path First (OSPF) protocol uses link state advertisements (LSAs) to update neighboring routers about interfaces and information on those interfaces. OSPF routers maintain identical databases that describe their area topology, helping any individual router to determine the shortest path between itself and any neighboring router.

OSPF show commands

This section describes the show commands that display OSPF information.

show ip ospf

Syntax: **show ip ospf**

This command displays information related to an OSPF router advertisement configuration. Command output resembles the following example.

```
Brocade# show ip ospf
OSPF Version                  Version 2
Router Id                     10.1.1.1
ASBR Status                   Yes
ABR Status                    Yes        (1)
Redistribute Ext Routes from  Static BGP
Initial SPF schedule delay    0          (msecs)
Minimum hold time for SPFs    0          (msecs)
Maximum hold time for SPFs    0          (msecs)
External LSA Counter          10001
External LSA Checksum Sum     13a1baba
Originate New LSA Counter     662043
Rx New LSA Counter            93400
External LSA Limit            14447047
Database Overflow Interval    0
Database Overflow State :     NOT OVERFLOWED
RFC 1583 Compatibility :      Enabled
Slow neighbor Flap-Action :   Enabled,   timer 300
Nonstop Routing:              Disabled
Graceful Restart:             Disabled,   timer 120
Graceful Restart Helper:      Enabled
```

show ip ospf config

Syntax: **show ip ospf config**

This command displays general OSPF configuration information.

```
Brocade# show ip ospf config
Router OSPF: Enabled
Redistribution: Disabled
Default OSPF Metric: 10
```

```
OSPF Redistribution Metric: Type2
OSPF External LSA Limit: 1447047
OSPF Database Overflow Interval: 0
RFC 1583 Compatibility: Enabled
Router id: 192.168.100.1
Interface State Change Trap:                          Enabled
Virtual Interface State Change Trap:                  Enabled
Neighbor State Change Trap:                           Enabled
Virtual Neighbor State Change Trap:                   Enabled
Interface Configuration Error Trap:                   Enabled
Virtual Interface Configuration Error Trap:           Enabled
Interface Authentication Failure Trap:                Enabled
Virtual Interface Authentication Failure Trap:        Enabled
Interface Receive Bad Packet Trap:                    Enabled
Virtual Interface Receive Bad Packet Trap:            Enabled
Interface Retransmit Packet Trap:                     Disabled
Virtual Interface Retransmit Packet Trap:             Disabled
Originate LSA Trap:                                   Disabled
Originate MaxAge LSA Trap:                            Disabled
Link State Database Overflow Trap:                    Disabled
Link State Database Approaching Overflow Trap:        Disabled
OSPF Area currently defined:
Area-ID          Area-Type Cost
0                normal    0
OSPF Interfaces currently defined:
Ethernet Interface: 3/1-3/2
ip ospf md5-authentication-key-activation-wait-time 300
ip ospf cost 0
ip ospf area 0
Ethernet Interface: v1
ip ospf md5-authentication-key-activation-wait-time 300
ip ospf cost 0
ip ospf area 0
```

show tasks

Syntax: show tasks

This command displays CPU usage statistics and other OSPF tasks.

```
Brocade# show tasks
Task Name   Pri   State     PC      Stack   Size  CPU Usage(%) task id  task vid
----------  ---  -----  ---------  --------  -----  ---------  ------ -------
       idle 0    ready  00001904  04058fa0  4096    99          0        0
    monitor 20    wait  0000d89c  0404bd80  8192    0           0        0
        int 16    wait  0000d89c  04053f90  16384   0           0        0
      timer 15    wait  0000d89c  04057f90  16384   0           0        0
        dbg 30    wait  0000d89c  0404ff08  8192    0           0        0
      flash 17    wait  0000d89c  0409ff90  8192    0           0        0
         wd 31    wait  0000d89c  0409df80  8192    0           0        0
       boot 17    wait  0000d89c  04203e28  65536   0           0        0
       main 3     wait  0000d89c  2060cf38  65536   0           0        1
        itc 6     wait  0000d89c  20612ae8  16384   0           0        1
        tmr 5     wait  0000d89c  20627628  16384   0           0        1
      ip_rx 5     wait  0000d89c  2062ff48  16384   0           0        1
        scp 5     wait  0000d89c  20635628  16384   0           0        1
    console 5     wait  0000d89c  2063e618  32768   0           0        1
       vlan 5     wait  0000d89c  20648618  16384   0           0        1
    mac_mgr 5     wait  0000d89c  20657628  16384   0           0        1
    mrp_mgr 5     wait  0000d89c  2065c628  16384   0           0        1
```

```
        vsrp 5     wait  0000d89c  20663620  16384   0           0        1
        snms 5     wait  0000d89c  20667628  16384   0           0        1
         rtm 5     wait  0000d89c  20674628  16384   0           0        1
        rtm6 5     wait  0000d89c  2068a628  16384   0           0        1
       ip_tx 5    ready  0000d89c  206a9628  16384   0           0        1
         rip 5     wait  0000d89c  20762628  16384   0           0        1
         bgp 5     wait  0000d89c  207e6628  16384   0           0        1
      bgp_io 5     wait  0000d89c  2082ef00  16384   0           0        1
        ospf 5     wait  0000d89c  20832628  16384   1           0        1
 ospf_r_calc 5     wait  0000d89c  2089ff10  16384   0           0        1
   isis_task 5     wait  0000d89c  208a3628  16384   0           0        1
    isis_spf 5     wait  0000d89c  208a8f10  16384   0           0        1
       mcast 5     wait  0000d89c  208ac628  16384   0           0        1
        vrrp 5     wait  0000d89c  208b4628  16384   0           0        1
       ripng 5     wait  0000d89c  208b9628  16384   0           0        1
       ospf6 5     wait  0000d89c  208c3628  16384   0           0        1
    ospf6_rt 5     wait  0000d89c  208c7f08  16384   0           0        1
      mcast6 5     wait  0000d89c  208cb628  16384   0           0        1
          l4 5     wait  0000d89c  208cf620  16384   0           0        1
         stp 5     wait  0000d89c  209a7620  16384   0           0        1
        snmp 5     wait  0000d89c  209c3628  32768   0           0        1
        rmon 5     wait  0000d89c  209cc628  32768   0           0        1
         web 5     wait  0000d89c  209d6628  32768   0           0        1
        lacp 5     wait  0000d89c  209da628  16384   0           0        1
       dot1x 5     wait  0000d89c  209e0620  16384   0           0        1
   hw_access 5     wait  0000d89c  209e6628  16384   0           0        1
```

show ip ospf area

Syntax: **show ip ospf area** [*area-id* | *num*]

- *area-id* - Shows information for the specified area.
- *num* - Corresponds to the entry number you enter. The entry number identifies the position of the entry in the area table.

This command displays OSPF area information.

```
Brocade# show ip ospf area
Indx  Area        Type    Cost  SPFR  ABR  ASBR  LSA  Chksum(Hex)
1   0.0.0.0      normal  0    1      0     0    1    0000781f
2   10.147.60.0  normal  0    1      0     0    1    0000fee6
3   10.147.80.0  stub    1    1      0     0    2    000181cd
```

show ip ospf neighbor

Syntax: **show ip ospf neighbor** [**extensive** |*num* | **router-id** *ip-addr*]

- **extensive** - Displays detailed information about the neighbor.
- *num* - Specifies the index position in the neighbor table for which you want to display neighbor information. For example, if you enter "1", only the first entry in the table is displayed.
- **router-id** *ip-addr* - Displays only the neighbor entries for the specified router.

The **show ip ospf neighbor extensive** command displays detailed information about the neighbor. Command output resembles the following example.

```
Brocade# show ip ospf neighbor extensive
Number of Neighbors is 2, in FULL state 2

Port   Address         Pri State      Neigh Address   Neigh ID        Ev Opt Cnt
```

```
1/7    10.1.1.1        1   FULL/BDR   10.1.1.2        10.1.1.36      5  66  0
Neighbor is known for 8d:16h:17m:33s and up for 8d:16h:16m:50s
1/5    10.1.1.1        1   FULL/BDR   10.1.1.5         10.4.4.4      4  72  8276
Neighbor is marked SLOW and is known for 8d:16h:16m:34s and up for 8d:16h:15m:50s
```

The **show ip ospf neighbor 1** command output resembles the following example.

```
Brocade# show ip ospf neighbor 1
Port   Address         Pri State      Neigh Address   Neigh ID        Ev Opt Cnt
1/7    10.1.1.1        1   FULL/BDR   10.1.1.2        10.1.1.36      5  66  0
address 10.1.1.2, priority 1, id 10.1.1.36
designated_router 10.1.1.1, backup_designated_router 10.1.1.2, interface state DR
state 8, event 5, mode 2, flags 00000001, option 00000042
ls_request_queue_count 0, ls_request_list_has_changed 0, ls_req_can_be_sent 0
retransmit_queue_count 0, database_summary_queue_count 0
pkt_rx_count 0
inactivity_timer_enabled 1,  periodic_inactivity_time_counter 1
md5_sequence 0,  sequence 1762200, neighbor_sequence 0
last_dd_sequence 1762199,  last_exchange 0
last_dd_flags 26e44b29,  last_dd_options 26e44b28
periodic_slave_hold_time_counter 165
slow_neighbor_wait_timer 10
flag_ret_Q_full_slow_neighbor 1
sptr_retransmit 00000000, sptr_retransmit_tail 00000000
sptr_database_summary 00000000
sptr_ls_request[1-5, 9] 00000000 00000000 00000000 00000000 00000000 00000000
interface 1/7, address 10.1.1.1, subnet/nexthop 10.1.1.0
```

show ip ospf interface

Syntax: **show ip ospf interface** [*ip-addr*]

The *ip-addr* variable specifies the IP address.

This command displays OSPF interface information, including the IP address, the OSPF state, link ID, and cost, as shown in the following example.

```
Brocade# show ip ospf interface 192.168.1.1
Ethernet 2/1,OSPF enabled
     IP Address 192.168.1.1, Area 0
     OSPF state ptr2ptr, Pri 1, Cost 1, Options 2,  Type pt-2-pt Events 1
     Timers(sec): Transit 1, Retrans 5, Hello 10, Dead 40
     DR:  Router ID 0.0.0.0           Interface Address 0.0.0.0
     BDR: Router ID 0.0.0.0           Interface Address 0.0.0.0
     Neighbor Count = 0, Adjacent Neighbor Count= 1
     Neighbor: 10.2.2.2
     Authentication-Key:None
     MD5 Authentication: Key None, Key-Id None, Auth-change-wait-time 300
```

show ip ospf interface brief

Syntax: **show ip ospf interface brief**

This command displays the OSPF database information in brief format, as shown in the following example.

```
Brocade# show ip ospf interface brief
Number of Interfaces is 1
Interface Area IP Addr/Mask Cost State Nbrs(F/C)
eth 1/2    0    10.1.1.2/24   1     down  0/0
```

show ip ospf routes

Syntax: show ip ospf routes [*ip-addr*]

- *ip-addr* - Specifies a destination IP address for which to display route entries.

This command displays OSPF route information, as shown in the following example.

```
Brocade# show ip ospf routes
OSPF Area 0x00000000 ASBR Routes 1:
      Destination     Mask            Path_Cost Type2_Cost Path_Type
      10.65.12.1      10.255.255.255 1         0          Intra
      Adv_Router      Link_State      Dest_Type State      Tag       Flags
      10.65.12.1      10.65.12.1      Asbr      Valid      0          6000
      Paths Out_Port  Next_Hop        Type      State
      1     v49       10.1.49.2       OSPF      21 01
      2     v12       10.1.12.2       OSPF      21 01
      3     v11       10.1.11.2       OSPF      21 01
      4     v10       10.1.10.2       OSPF      00 00
OSPF Area 0x00000041 ASBR Routes 1:
      Destination     Mask            Path_Cost Type2_Cost Path_Type
      10.65.12.1      10.255.255.255 1         0          Intra
      Adv_Router      Link_State      Dest_Type State      Tag       Flags
      10.65.12.1      10.65.12.1      Asbr      Valid      0          6000
      Paths Out_Port  Next_Hop        Type      State
      1     v204      10.65.5.251     OSPF      21 01
      2     v201      10.65.2.251     OSPF      20 d1
      3     v202      10.65.3.251     OSPF      20 cd
      4     v205      10.65.6.251     OSPF      00 00
OSPF Area Summary Routes 1:
      Destination     Mask            Path_Cost Type2_Cost Path_Type
      10.65.0.0       10.255.0.0     0         0          Inter
      Adv_Router      Link_State      Dest_Type State      Tag       Flags
      10.1.10.1       0.0.0.0         Network   Valid      0          0000
      Paths Out_Port  Next_Hop        Type      State
      1     1/1       0.0.0.0         DIRECT    00 00
OSPF Regular Routes 208:
      Destination     Mask            Path_Cost Type2_Cost Path_Type
      10.1.10.0       10.255.255.252 1         0          Intra
      Adv_Router      Link_State      Dest_Type State      Tag       Flags
      10.1.10.1       10.1.10.2       Network   Valid      0          0000
      Paths Out_Port  Next_Hop        Type      State
      1     v10       0.0.0.0         OSPF      00 00
      Destination     Mask            Path_Cost Type2_Cost Path_Type
      10.1.11.0       10.255.255.252 1         0          Intra
      Adv_Router      Link_State      Dest_Type State      Tag       Flags
      10.1.10.1       10.1.11.2       Network   Valid      0          0000
      Paths Out_Port  Next_Hop        Type      State
      1     v11       0.0.0.0         OSPF      00 00
```

show ip ospf redistribute route

Syntax: show ip ospf redistribute route [*ip-addr ip-mask*]

- *ip-addr* - Specifies a destination IP address for which to display route entries.
- *ip-mask* - Specifies the route mask.

This command displays routes that have been redistributed into OSPF.

```
Brocade# show ip ospf redistribute route
10.3.0.0 10.255.0.0 static
10.1.0.0 10.255.0.0 static
10.11.61.0 10.255.255.0 connected
10.1.0.0 10.255.0.0 static
```

In this example, four routes have been redistributed. Three of the routes were redistributed from static IP routes and one was redistributed from a directly-connected IP route.

To display route redistribution for a specific IP address and mask, enter the following command as shown in the following example.

```
Brocade# show ip ospf redistribute route 10.1.0.0 10.255.0.0
10.1.0.0 10.255.0.0 static
```

show ip ospf database

Syntax: show ip ospf database

This command displays the OSPF database, as shown in the following example.

```
Brocade# show ip ospf database
Graceful Link States
Area  Interface  Adv Rtr  Age Seq(Hex) Prd Rsn  Nbr Intf IP
0     eth 1/2    10.2.2.2  7   80000001 60  SW   10.1.1.2

Router Link States
Index AreaID            Type LS ID      Adv Rtr     Seq(Hex)   Age  Cksum
SyncState
1     0                 Rtr  10.2.2.2    10.2.2.2     80000003   93   0xac6c
Done
2     0                 Rtr  10.1.1.1    10.1.1.1     80000005   92   0x699e
Done
3     0                 Net  10.1.1.2    10.2.2.2     80000002   93   0xbd73
Done
4     0                 OpAr 10.0.0.3    10.1.1.1     80000005   83   0x48e7
Done
5     0                 OpAr 10.0.0.2    10.2.2.2     80000006   80   0x50da
Done
6     10.111.111.111     Rtr  10.1.1.1    10.1.1.1     80000004   142  0x0a38
Done
7     10.111.111.111     Summ 10.1.1.1    10.1.1.1     80000001   147  0x292b
Done
8     10.111.111.111     OpAr 10.0.0.2    10.1.1.1     80000002   179  0x063f
Done
Type-5 AS External Link States
Index Age  LS ID     Router   Netmask   Metric    Flag  Fwd Address
1     147  10.9.1.13  10.1.1.1  ffffffff  0000000a  0000  0.0.0.0
2     147  10.9.1.26  10.1.1.1  ffffffff  0000000a  0000  0.0.0.0
```

show ip ospf database external-link-state

Syntax: show ip ospf database external-link-state [**advertise** *num* | **extensive** | **link-state-id** *ip-addr* | **router-id** *ip-addr* | **sequence-number** *num(Hex)*]

- **advertise** *num* - Displays the hexadecimal data in the specified LSA packet. The *num* variable identifies the LSA packet by its position in the router's External LSA table. Enter the **show ip ospf database external-link-state** command to display the table.
- **extensive** - Displays the LSAs in decrypted format.

NOTE
The **extensive** option displays the entire database and cannot be used in combination with other display options.

- **link-state-id** *ip-addr* - Displays the External LSAs for the LSA source for the specified IP address.
- **router-id** *ip-addr* - Shows the External LSAs for the specified OSPF router.
- **sequence-number** *num(Hex)* - Displays the External LSA entries for the specified hexadecimal LSA sequence number.

This command displays external link state information, as shown in the following example.

```
Brocade# show ip ospf database external-link-state
Index Aging  LS ID           Router          Netmask  Metric    Flag  Fwd Address
SyncState
1     591    10.65.13.0      10.65.12.1      ffffff00 8000000a  0000  0.0.0.0
Done
2     591    10.65.16.0      10.65.12.1      ffffff00 8000000a  0000  0.0.0.0
Done
3     591    10.65.14.0      10.65.12.1      ffffff00 8000000a  0000  0.0.0.0
Done
4     591    10.65.17.0      10.65.12.1      ffffff00 8000000a  0000  0.0.0.0
Done
5     592    10.65.12.0      10.65.12.1      ffffff00 8000000a  0000  0.0.0.0
Done
6     592    10.65.15.0      10.65.12.1      ffffff00 8000000a  0000  0.0.0.0
Done
7     592    10.65.18.0      10.65.12.1      ffffff00 8000000a  0000  0.0.0.0
Done
```

show ip ospf database database-summary

Syntax: **show ip ospf database database-summary**

This command displays database summary information, as shown in the following example.

```
Brocade# show ip ospf database database-summary
Area ID      Router  Network Sum-Net  Sum-ASBR  NSSA-Ext  Opq-Area  Subtotal
0.0.0.0      104      184     19       42        0         0          349
AS External                                                           308
Total        104       184    19       42        0         0          657
```

show ip ospf database link-state

Syntax: **show ip ospf database link-state** [**advertise** *num* | **asbr** [*ip-addr*] [**adv-router** *ip-addr*] | **extensive** | **link-state-id** *ip-addr* | **network** [*ip-addr*] [**adv-router** *ip-addr*] | **nssa** *ip-addr* [**adv-router** *ip-addr*] | **router** [*ip-addr*] [**adv-router** *ip-addr*] | **router-id** *ip-addr* | **self-originate** l **sequence-number** *num(Hex)* | **summary** [*ip-addr*] [**adv-router** *ip-addr*]]

- **advertise** *num* - Displays the hexadecimal data in the specified LSA packet. The *num* variable identifies the LSA packet by its position in the router LSA table. To determine an LSA packet's position in the table, enter the **show ip ospf database link-state** command to display the table.
- **asbr** - Displays the link state by ASBR link.
- *ip-addr* - Displays the link state ID.
- [**adv-router** *ip-addr*] - Displays the link state by advertising router.
- **extensive** - Displays the LSAs in decrypted format.

NOTE
The **extensive** option displays the entire database and cannot be used in combination with other display options.

- **link-state-id** *ip-addr* - Displays the LSAs for the LSA source for the specified IP address.
- **network** - Shows network LSAs.
- **nssa** - Shows NSSA LSAs.
- **router** - Displays LSAs by router link.
- **router-id** *ip-addr* - Shows the LSAs for the specified OSPF router.
- **self-originate** - Shows self-originated LSAs.
- **sequence-number** *num(Hex)* - Displays the LSA entries for the specified hexadecimal LSA sequence number.
- **summary** - Shows summary information.

This command displays database link state information, as shown in the following example.

```
Brocade# show ip ospf database link-state
Index Area ID        Type   LS ID           Adv Rtr          Seq(Hex) Age  Cksum
1     0              Rtr  10.1.10.1       10.1.10.1       800060ef 3    0x4be2
2     0              Rtr  10.65.12.1      10.65.12.1      80005264 6    0xc870
3     0              Net  10.1.64.2       10.65.12.1      8000008c 1088 0x06b7
4     0              Net  10.1.167.2      10.65.12.1      80000093 1809 0x86c8
5     0              Net  10.1.14.2       10.65.12.1      8000008c 1088 0x2ec1
6     0              Net  10.1.117.2      10.65.12.1      8000008c 1087 0xbccb
7     0              Net  10.1.67.2       10.65.12.1      8000008c 1088 0xe4d5
8     0              Net  10.1.170.2      10.65.12.1      80000073 604  0xa5c6
9     0              Net  10.1.17.2       10.65.12.1      8000008c 1088 0x0ddf
10    0              Net  10.1.120.2      10.65.12.1      8000008c 1087 0x9be9
11    0              Net  10.1.70.2       10.65.12.1      8000008c 1088 0xc3f3
12    0              Net  10.1.173.2      10.65.12.1      80000017 1087 0x3d88
13    0              Net  10.1.20.2       10.65.12.1      8000008c 1088 0xebfd
14    0              Net  10.1.123.2      10.65.12.1      8000008c 1087 0x7a08
15    0              Net  10.1.73.2       10.65.12.1      8000008c 1088 0xa212
16    0              Net  10.1.176.2      10.65.12.1      80000025 1087 0xffb4
17    0              Net  10.1.23.2       10.65.12.1      8000008c 1088 0xca1c
18    0              Net  10.1.126.2      10.65.12.1      8000008c 1087 0x5926
```

show ip ospf border-routers

Syntax: show ip ospf border-routers [*ip-addr*]

- *ip-addr* - Displays the ABR and ASBR entries for the specified IP address.

This command displays OSPF ABR and ASBR information, as shown in the following example.

```
Brocade# show ip ospf border-routers
      router ID       router type next hop router  outgoing interface  Area
1     10.65.12.1      ABR         10.1.49.2        v49                 0
1     10.65.12.1      ASBR        10.1.49.2        v49                 0
1     10.65.12.1      ABR         10.65.2.251      v201                65
1     10.65.12.1      ASBR        10.65.2.251      v201                65
```

show ip ospf trap

Syntax: show ip ospf trap

All traps are enabled by default when you enable OSPF. This command displays the state of each OSPF trap.

```
Brocade# show ip ospf trap
Interface State Change Trap:                        Enabled
Virtual Interface State Change Trap:                Enabled
Neighbor State Change Trap:                         Enabled
Virtual Neighbor State Change Trap:                 Enabled
Interface Configuration Error Trap:                 Enabled
Virtual Interface Configuration Error Trap:         Enabled
Interface Authentication Failure Trap:              Enabled
Virtual Interface Authentication Failure Trap:      Enabled
Interface Receive Bad Packet Trap:                  Enabled
Virtual Interface Receive Bad Packet Trap:          Enabled
Interface Retransmit Packet Trap:                   Disabled
Virtual Interface Retransmit Packet Trap:           Disabled
Originate LSA Trap:                                 Disabled
Originate MaxAge LSA Trap:                          Disabled
Link State Database Overflow Trap:                  Disabled
Link State Database Approaching Overflow Trap:      Disabled
```

show run

Syntax: show run

This command displays OSPF virtual neighbor and virtual link information, as shown in the following example.

```
Brocade# show run
Current configuration:
ver V2.2.1T143
module 1 rx-bi-1g-24-port-fiber
module 2 rx-bi-10g-4-port
module 6 rx-bi-10g-4-port
module 7 rx-bi-1g-24-port-copper
!
no spanning-tree
!
vlan 1 name DEFAULT-VLAN
!
clock summer-time
clock timezone us Pacific
hostname R11-RX8
router ospf
 area 2
 area 1
 area 1 virtual-link 10.1.1.10
```

show ip ospf virtual neighbor

Syntax: show ip ospf virtual neighbor [*num*]

The *num* variable specifies the index entry of the neighbors.

This command displays OSPF virtual neighbor information, as shown in the following example.

```
Brocade# show ip ospf virtual neighbor
Indx Transit Area    Router ID       Neighbor address options
1    1               10.1.1.10      135.14.1.10      2
     Port   Address         state           events          count
\     6/2    10.11.1.27      FULL            5               0
```

show ip ospf virtual link

Syntax: **show ip ospf virtual link** [*num*]

- *num* - Displays the table beginning at the specified entry number.

This command displays OSPF virtual link information, as shown in the following example.

```
Brocade# show ip ospf virtual link
Indx Transit Area     Router ID        Transit(sec) Retrans(sec) Hello(sec)
1    1                10.1.1.10        1            5            10
     Dead(sec)        events           state        Authentication-Key
     40               1                ptr2ptr      None
     MD5 Authentication-Key:       None
     MD5 Authentication-Key-Id:    None
     MD5 Authentication-Key-Activation-Wait-Time:      300
```

show ip ospf database grace-link-state

Syntax: **show ip ospf database grace-link-state**

This command displays Type 9 graceful LSAs, as shown in the following example.

```
Brocade#show ip ospf database grace-link-state
Graceful Link States
Area  Interface  Adv Rtr  Age Seq(Hex) Prd Rsn  Nbr Intf IP
0     eth 1/2    10.2.2.2  7   80000001 60  SW   10.1.1.2
```

show ip ospf debug

Syntax: **show ip ospf debug**

This command generates descriptive information about OSPF activity, as shown in the following example.

```
Brocade# show ip ospf debug
External LSA Counter          1
Timer enable 1, 1s counter 5, ticks/sec 10, currtime 1167, md5_seq 0
sptr_area_list 0x0b10002c, import_routes 1
build_routing_table 0, is pending 0, ospf_spf_pending_list_fwd 0
route_calculation_in_progress 0
ospf->ospf_schedule_build_wait_time 0
SPF build timers:  last end 896, scheduled 821, init 863, run 821
route_calc_process_take_semaphore 6
process_redis_events 0, ospf_flush_cache_for_new_route 0
originate_ext_lsa_counts 2, ospfOriginateNewLsas 20
*ospf->of_max_one_second_timer_value 0, ospf->of_one_second_timer_value 0
*ospf->of_max_redis_timer_value 0, ospf->of_redis_timer_value 0
ospf->of_max_r_cal_time_value 0 0 0 0 0 0 0 0
of_max_change_ext_route_fwd_addr_time_value
ospf->of_max_originate_external_lsa_time_value 0
```

```
ospf->of_max_retransmit_lsa_timer_value 0, ospf->of_retransmit_lsa_timer_value 0
ospf->of_max_retransmit_db_timer_value 0, ospf->of_retransmit_db_timer_value 0
*of_max_neighbor_retransmit_queue_count 0, ospf->ospf_retransmit_queue_count 0
*of_max_retransmit_queue_count_exceed 0
ospf->set_one_shot_timer 0, ospf->ospf_one_shot_timer_token 0
*ospf->of_max_one_shot_timer_value 0, ospf->of_one_shot_timer_value 0
*ospf->of_max_one_shot_timer_count_long 0,
ospf->of_max_one_shot_timer_count_short 0
ospf->of_max_flood_refresh_lsa_timer_value 0, ospf-
>of_flood_refresh_lsa_timer_value 0
total ospf->of_flood_refresh_lsa_func_count 0, current flood_lsa_count 20
ospf->ospf_out_of_memory_for_send_packet 0
ospf->of_rtm_add_redis_count 1,ospf->of_rtm_add_redis_added_count 1,
invalid_count 0
ospf->of_rtm_del_redis_count 0, ospf->of_rtm_del_redis_deleted_count 0
of_rtm_clear_count 0,  of_rtm_clear_all_count 0, of_rtm_default_count 0
number_of_routes_imported 1, ospf->of_rtm_modify_redis_count 0
*of_max_process_adver_time_value 0, of_process_adver_time_value 0
*ospf->of_max_find_lsa_time_value 0, ospf->of_find_lsa_time_value 0
*ospf->of_max_cleaup_database_time_value 0, ospf->of_max_find_database_time_value
0
*ospf->of_max_find_ls_request_time_value 0, ospf->of_max_count_on_finding_lsa 0
msg_q_length = 0 msg_q_high_mark = 1
```

show ip ospf debug memory

Syntax: show ip ospf debug memory

This command displays information about OSPF memory pools. Command output resembles the following example.

```
Brocade# show ip ospf debug memory
OSPF Memory Use 1302832
Pid SBlock TBlocks UBlocks FBlocks EBlocks SAddress CAddress
0   0      0       0       0       0       00000000 00000000
1   40     2000    30      1970    0       09207010 09207470
2   56     4000    25      3975    0       0921a8a0 0921ae18
3   132    32      10      22      0       07ab7b48 07ab8070
4   260    16      2       14      0       07ab8bd8 07ab8bd8
5   516    32      4       28      0       07ab9c28 07aba438
6   1504   32      0       32      0       092513b0 092513b0
7   4290   16      1       15      0       0925cfc0 0925e082
8   53571  16      3       13      0       0b100028 0b1273f1
9   0      0       0       0       0       00000000 00000000
Total Memory blocks allocated 75
Mega Memory List
Pool Id = 1, Total Mega blocks = 1 Errors = 0
Pool Id = 2, Total Mega blocks = 1 Errors = 0
Pool Id = 3, Total Mega blocks = 1 Errors = 0
Pool Id = 4, Total Mega blocks = 1 Errors = 0
Pool Id = 5, Total Mega blocks = 1 Errors = 0
Pool Id = 6, Total Mega blocks = 1 Errors = 0
Pool Id = 7, Total Mega blocks = 1 Errors = 0
Pool Id = 8, Total Mega blocks = 1 Errors = 0
OSPF Main Routing Table: 078dd444
node_count 6, top 0x078dd5b4, default_valid 0, default_route 0xffffffff
Table private pool:
init#=4096 unit_s=36 total=4096 in_use=5 fail=0 upper=no-limit min_mem=0
    UBlocks  EBlocks Total   PType
0   0      0      0       OSPF_MEMORY_POOL_ANY
```

```
1   3      0      14      OSPF_MEMORY_POOL_ROUTER_LINK_ADVERTISEMENT
2   1      0      2       OSPF_MEMORY_POOL_NETWORK_LINK_ADVERTISEMENT
3   15     0      17      OSPF_MEMORY_POOL_SUMMARY_LINK_ADVERTISEMENT
4   1      0      2       OSPF_MEMORY_POOL_EXTERNAL_LINK_ADVERTISEMENT
5   0      0      0       OSPF_MEMORY_POOL_GRACE_LINK_ADVERTISEMENT
6   5      0      9       OSPF_MEMORY_POOL_OPAQUE_AREA_ADVERTISEMENT
7   0      0      1       OSPF_MEMORY_POOL_LS_DATABASE_SUMMARY
8   0      0      10      OSPF_MEMORY_POOL_LS_DATABASE_NODE
9   0      0      30      OSPF_MEMORY_POOL_SHORTEST_PATH_NODE
Total Memory blocks allocated 75
Mega Memory List
Pool Id = 1, Total Mega blocks = 1 Errors = 0
Pool Id = 2, Total Mega blocks = 1 Errors = 0
Pool Id = 3, Total Mega blocks = 1 Errors = 0
Pool Id = 4, Total Mega blocks = 1 Errors = 0
Pool Id = 5, Total Mega blocks = 1 Errors = 0
Pool Id = 6, Total Mega blocks = 1 Errors = 0
Pool Id = 7, Total Mega blocks = 1 Errors = 0
Pool Id = 8, Total Mega blocks = 1 Errors = 0
OSPF Main Routing Table: 078dd444
node_count 6, top 0x078dd5b4, default_valid 0, default_route 0xffffffff
init#=4096 unit_s=36 total=4096 in_use=5 fail=0 upper=no-limit min_mem=0
Mega Memory List
Pool Id = 1, Total Mega blocks = 1 Errors = 0
Pool Id = 2, Total Mega blocks = 1 Errors = 0
Pool Id = 3, Total Mega blocks = 1 Errors = 0
Pool Id = 4, Total Mega blocks = 1 Errors = 0
Pool Id = 5, Total Mega blocks = 1 Errors = 0
Pool Id = 6, Total Mega blocks = 1 Errors = 0
Pool Id = 7, Total Mega blocks = 1 Errors = 0
Pool Id = 7, Total Mega blocks = 1 Errors = 0
Pool Id = 8, Total Mega blocks = 1 Errors = 0
OSPF Main Routing Table: 078dd444
node_count 6, top 0x078dd5b4, default_valid 0, default_route 0xffffffff
init#=4096 unit_s=36 total=4096 in_use=5 fail=0 upper=no-limit min_mem=0
UBlocks  EBlocks Total   PType
0   0      0      0       OSPF_MEMORY_POOL_ANY
1   3      0      14      OSPF_MEMORY_POOL_ROUTER_LINK_ADVERTISEMENT
2   1      0      2       OSPF_MEMORY_POOL_NETWORK_LINK_ADVERTISEMENT
3   15     0      17      OSPF_MEMORY_POOL_SUMMARY_LINK_ADVERTISEMENT
4   1      0      2       OSPF_MEMORY_POOL_EXTERNAL_LINK_ADVERTISEMENT
5   0      0      0       OSPF_MEMORY_POOL_GRACE_LINK_ADVERTISEMENT
6   5      0      9       OSPF_MEMORY_POOL_OPAQUE_AREA_ADVERTISEMENT
7   0      0      1       OSPF_MEMORY_POOL_LS_DATABASE_SUMMARY
8   0      0      10      OSPF_MEMORY_POOL_LS_DATABASE_NODE
9   0      0      30      OSPF_MEMORY_POOL_SHORTEST_PATH_NODE
10  8      0      8       OSPF_MEMORY_POOL_OSPF_ROUTE_INFO
11  6      0      6       OSPF_MEMORY_POOL_OSPF_MAIN_ROUTE_ENTRY
12  0      0      0       OSPF_MEMORY_POOL_OSPF_SUMMARY_ROUTE_ENTRY
13  0      0      0       OSPF_MEMORY_POOL_OSPF_EXT_SUMMARY_ROUTE_ENTRY
14  1      0      1       OSPF_MEMORY_POOL_OSPF_ABR_ROUTE_ENTRY
15  1      0      1       OSPF_MEMORY_POOL_OSPF_ASBR_ROUTE_ENTRY
16  1      0      1       OSPF_MEMORY_POOL_EXTERNAL_ROUTE
17  0      0      0       OSPF_MEMORY_POOL_ADVERTISEMENT_NODE
18  25     0      35      OSPF_MEMORY_POOL_LS_DATABASE_ENTRY
19  0      0      2       OSPF_MEMORY_POOL_LS_REQUEST
20  0      0      4       OSPF_MEMORY_POOL_LS_HEADER_QUEUE
21  0      0      5       OSPF_MEMORY_POOL_NEIGHBOR_LIST
22  0      0      0       OSPF_MEMORY_POOL_TRANSIT_AREA_ENTRY
23  0      0      29      OSPF_MEMORY_POOL_NEXT_HOP_BLOCK
```

```
24  0      0      0       OSPF_MEMORY_POOL_HOSTS
25  0      0      0       OSPF_MEMORY_POOL_ADDRESS_RANGE_LIST
26  1      0      1       OSPF_MEMORY_POOL_NEIGHBOR
27  4      0      4       OSPF_MEMORY_POOL_INTERFACE
28  0      0      1       OSPF_MEMORY_POOL_OSPF_HEADER
29  3      0      3       OSPF_MEMORY_POOL_AREA
Total Memory blocks allocated 75
```

show ip ospf debug misc

Syntax: show ip ospf debug misc

This command displays miscellaneous OSPF information, including router counts and SPF calculations, as shown in the following example.

```
Brocade# show ip ospf debug misc
Type-5 Forwarding Addr Count :0
Imported Route Count :        1
External Route Flap Count :   0
NSSA Route Flap Count :       0
External Lsa Count :          1
NSSA Lsa Count :              0
OSPF Recalc Statistics:
        phase_number: 0, area_id: 0xffffffff, next_chunk: 0x00000000
        duration(50ms): 0
MAX_AGE EXT lsa count 0, total EXT lsa count 1
```

show ip ospf debug graceful-restart

Syntax: show ip ospf debug graceful-restart

This command displays information about OSPF graceful restart events, as shown in the following example.

```
Brocade# show ip ospf debug graceful-restart
MP active: 1, standby up 0, nbr (1 0), vi (0, 0)
OSPF graceful-restart: enable 0, helper 1, timer 120/0, count 0, restarting 0
OSPF graceful-restart helper:
     Neighbor             ID      Area   Interface State Grace Helper Time
   10.1.1.1           10.1.1.1     0      1/1      8       0      0 0
OSPF graceful-restart LSA:
Area Interface         ID      Type Age   Max Seq        Interface Option
```

show ip ospf debug filtered-lsa area

Syntax: show ip ospf debug filtered-lsa area [**area_ID** | **area_IP**] [**in** | **out**]

- **area_ID** - Specifies the area ID.
- **area_IP** - Specifies the area IP address.
- **in** - Displays filtered LSAs in an area in **IN** direction.
- **out** - Displays filtered LSAs in an area in **OUT** direction.

This command displays the type-3 filtered LSAs for an area either in **IN** direction or **OUT** driection. The command output resembles the following example

```
Brocade# show ip ospf debug filtered-lsa area 0 in
Prefix List Name: 233deny
Direction : IN
Area : 0
```

```
Filtered LSA list:
Prefix     Mask
10.3.0.0   10.255.252.0
```

Clearing OSPF neighbors

You can clear all OSPF neighbors or a specified OSPF neighbor using the following command.

clear ip ospf neighbor

Syntax: **clear ip ospf neighbor** [**all** | *ip-address*]

- **all** - Clears all of the OSPF neighbors on the router.
- *ip-address* - Clears a specific OSPF neighbor.

OSPF debug commands

The following section describes the OSPF debug commands and shows examples of output from these commands.

debug ip ospf

Syntax: [**no**] **debug ip ospf** [*A.B.C.D* | **adj** | **all-vrfs** | **bfd** | **error** | **events** | **flood** | **graceful-restart** | **log-debug-message** | **log-empty-lsa** | **lsa-filtering** | **lsa-generation** | **max-metric** | **packet** | **retransmission** | **route** *ip-addr*| **sham-link** | **shortcuts** | **spf** | **vrf**]

- *A.B.C.D* - Displays OSPF information for a specific IP address.
- **adj** - Displays information about IP OSPF adjacencies.
- **all-vrfs** - Displays OSPF information specific to all VRFs.
- **bfd** - Displays information about OSPF BFD events.
- **error** - Displays IP OSPF errors.
- **events** - Displays IP OSPF events.
- **flood** - Displays IP OSPF flood information.
- **graceful-restart** - Displays information about graceful restarts.
- **log-debug-message** - Displays log-debug messages.
- **log-empty-lsa** - Displays information about empty link state advertisements (LSAs).
- **lsa-filtering** - Displays type-3 LSAs as and when a router generates and filters these LSAs.
- **lsa-generation** - Displays information about LSAs.
- **max-metric** - Displays information about a max-metric configuration.
- **packet** - Displays IP OSPF packet information.
- **retransmission** - Displays IP OSPF retransmission information.
- **route** *ip-addr* - Displays information about IP OSPF routes.
- **sham-link** - Displays information about a sham-link configuration.
- **shortcuts** - Displays information about OSPF shortcuts for IP over MPLS.
- **spf** - Displays IP OSPF SPF information.
- **vrf** - Displays OSPF information for the specified VRF.

debug ip ospf

Syntax: [no] debug ip ospf *A.B.C.D*

This command generates OSPF debugging information about a specific neighbor. Output indicates state transitions, hello packets received, LSA acknowledgements received, LSA processing and flooding information, and database descriptions, similar to the following example.

```
Brocade# debug ip ospf 10.1.1.1
OSPF: rvcd
10.1.1.1
OSPF: Neighbor 10.1.1.1, int 1/1, state FULL processing event HELLO_RECEIVED
hello from 10.1.1.1 area 0 on interface 10.1.1.2, state DR, DR 10.1.1.2, BDR
OSPF: Neighbor 10.1.1.1, int 1/1, state FULL processing event ADJACENCY_OK
OSPF: rcv lsa ack from neighbor 10.1.1.1, state FULL
OSPF: rcv LSA ack from 10.1.1.1, type 10, id 10.0.0.7,seq 0x80000009,adv 10.2.2.2,
age 1
```

debug ip ospf adj

Syntax: [no] debug ip ospf adj

This command displays information about OSPF adjacencies and authentication, including designated router (DR) and backup designated router (BDR) elections, sent and received hello packets, neighbor state transitions, and database description information. Command output resembles the following example.

```
Brocade# debug ip ospf adj
OSPF:  adjacency events debugging is on
OSPF: rvcd hello from 10.1.1.1 area 0 on interface 10.1.1.2, state DR, DR 0.0.0.0,
BDR 0.0.0.0
OSPF: Neighbor 10.1.1.1, int 1/1,  state DOWN processing event HELLO_RECEIVED
OSPF: Neighbor 10.1.1.1 state changed from Down to Initializing - event
HELLO_RECEIVED, intf-type 1
OSPF: Neighbor 10.1.1.1, int 1/1,  state INITIALIZING processing event ONE_WAY
OSPF: send hello on area 0 interface 10.2.2.2
OSPF: send hello on area 0 interface 10.1.1.2
OSPF: Neighbor 10.1.1.1, int 1/1,  state INITIALIZING processing event
TWO_WAY_RECEIVED
OSPF: establish_adjacency with 10.1.1.1
OSPF: DR/BDR election for 10.1.1.2 on 1/1
OSPF: Run interface 10.1.1.2 DR elect, state changed to DR from DR
OSPF: interface (10.1.1.2) state = INTERFACE_DESIGNATED_ROUTER
OSPF: Neighbor 10.1.1.1, int 1/1,  state EXCHANGE_START processing event
ADJACENCY_OK
OSPF: 10.1.1.2 Flushing Network LSA if needed as we are not DR anymore, state old
5, new 5
OSPF: elect BDR(backup designated router): Router ID 10.1.1.1 IP interface
10.1.1.1
OSPF: elect DR(designated router): Router ID 10.2.2.2, IP interface 10.1.1.2
OSPF: Neighbor 10.1.1.1 state changed from Initializing to ExStart - event
TWO_WAY_RECEIVED, intf-type 1
```

debug ip ospf all-vrfs

Syntax: [no] debug ip ospf all-vrfs

This command enables OSPF debugging for all VPN routing and forwarding activity. Output is similar to that of the **debug ip ospf** command.

debug ip ospf bfd

Syntax: [no] debug ip ospf bfd

This command displays information about OSPF BFD events.

debug ip ospf error

Syntax: [no] debug ip ospf error

This command reports the receipt of OSPF packets with errors, or mismatches between hello packet options.

CAUTION

If the router receives too many packets with errors, substantial output may be generated and severely affect system performance. To prevent a disruption of system activity, use this command only when network traffic levels are low.

debug ip ospf events

Syntax: [no] debug ip ospf events

This command displays information about internal OSPF events related to configuration or interaction with the standby management processor and interface state transitions. Command output resembles the following example.

```
Brocade# debug ip ospf events
OSPF: Interface 1/1 (10.1.1.2) state Down processing event Interface Up
OSPF: interface 10.1.1.2 up, state changed to WAITING from Down
OSPF: Interface 1/1 (10.1.1.2) state Waiting processing event Neighbor Change
OSPF: Interface 1/1 (10.1.1.2) state Waiting processing event Backup Seen
```

debug ip ospf flood

Syntax: [no] debug ip ospf flood

This command displays information about LSA flooding activity. Command output resembles the following example.

```
Brocade# debug ip ospf flood
OSPF:  flooding debugging is on
OSPF: flood LSA Type:1 AdvRtr:10.2.2.2 Age:0 LsId:10.2.2.2
OSPF: flood advertisement throughout a specific area = 00000001
OSPF: flood LSA Type:3 AdvRtr:10.2.2.2 Age:0 LsId:0.0.0.0
OSPF: flood advertisement throughout a specific area = 00000001
OSPF: flood LSA Type:3 AdvRtr:10.2.2.2 Age:0 LsId:0.0.0.0
OSPF: flood advertisement throughout a specific area = 00000003
OSPF: flood LSA Type:3 AdvRtr:10.2.2.2 Age:0 LsId:10.2.2.2
```

debug ip ospf graceful-restart

Syntax: [no] debug ip ospf graceful-restart

Enable this command to receive information about OSPF graceful restart events, including restart phases, graceful LSA transmit and receive activity, and syslog messages, as shown in the following example.

```
Brocade# debug ip ospf graceful-restart
Restart Router:
rw_mbridge_isr is called (cause = 00ff0002)
rw_isr_active_mp_lost() called
MP Manufacture Info:
=== Manufacturing Information ====
Board Class : 01 (Mgmt)
Foundry Assembly Part Number : 31524-000A
Chassis Type: NI-XMR (ac)
Foundry Serial Number : PR30050521
Date of Manufacture : 255-255-2225
Bench Test Status : UNKNOWN
Burn-in Test Status : UNKNOWN
Manufacturing Deviation : ÿÿÿÿÿÿ
RMA Number : ÿÿÿÿÿÿÿÿÿÿ
PCB Revision : ÿÿÿÿ
MFG Test : ÿÿÿÿÿÿÿÿÿÿ
g_slot_presence_mask = 00000189, g_snm_presence_mask = 00000007
End Time: my_slot = 18, active_mp_slot = 18, standby_mp_slot = 17
rw_mbridge_isr is called (cause = 00fc0005)
rw_isr_present is called:
OLD: prw = 000000fc, fan = 00000000, lp = 0000fe76, snm = 00080000, mnm = 00000000
NEW: prw = 000000fc, fan = 00000000, lp = 0000fe76, snm = 00080000, mnm = 00000000
rw_isr_power is called (snapshot = 000000fc)
RW_MBRIDGE_CARD_PRESENT_REG = 0008fe76
RW_MBRIDGE_CARD_POWER_OFF_REG = 00070189
SNM1 presence detected, powering it on
SNM2 presence detected, powering it on
SNM3 presence detected, powering it on
Power on SNM1: Writing 00070189 to RW_MBRIDGE_CARD_POWER_OFF_REG
Power on SNM2: Writing 00070189 to RW_MBRIDGE_CARD_POWER_OFF_REG
Power on SNM3: Writing 00070189 to RW_MBRIDGE_CARD_POWER_OFF_REG
MGMT board temp is: 35.500C  52.625C
Power Supply 1 is Installed (OK)
Power Supply 2 is Installed (OK)
Power Supply 3 is Not Installed (FAILED)
Power Supply 4 is Not Installed (FAILED)
Power Supply 5 is Not Installed (FAILED)
Power Supply 6 is Not Installed (FAILED)
Power Supply 7 is Not Installed (FAILED)
Power Supply 8 is Not Installed (FAILED)
Write 00070189 to RW_MBRIDGE_CARD_POWER_OFF_REG
Module is up in slot 1
Module is up in slot 4
Module is up in slot 8
Module is up in slot 9
All Modules Are Up (4 total)
SYSLOG: Feb 1 23:59:19:<13>XMR1, System: Module up in slot 4
SYSLOG: Feb 1 23:59:19:<13>XMR1, System: Module up in slot 8
SYSLOG: Feb1 23:59:19:<13>XMR1, System: Module up in slot 9

SYSLOG: Feb 1 23:59:19:<13>XMR1, System: Module up in slot 1
SYSLOG: Feb 1 23:59:19:<14>XMR1, System: Interface ethernet1/1, state up
SYSLOG: Feb 1 23:59:19:<14>XMR1, System: Interface ethernet1/9, state up
SYSLOG: Feb 1 23:59:19:<14>XMR1, System: Interface ethernet1/12, state up
SYSLOG: Feb 1 23:59:19:<14>XMR1, System: Interface ethernet9/1, state up
SYSLOG: Feb 1 23:59:19:<9>XMR1, System: Management module at slot 18 state changed
from standby to active
OSPF: switchover handoff done
OSPF: send_grace_ls to 224.0.0.5, intf addr 10.1.1.1, age 0, auth 0
```

```
OSPF: send_grace_ls to 224.0.0.5, intf addr 10.0.0.1, age 0, auth 0
OSPF: send_grace_ls to 224.0.0.5, intf addr 10.0.0.1, age 0, auth 0
OSPF: send_grace_ls to 224.0.0.5, intf addr 10.0.0.1, age 0, auth 0
OSPF: send_grace_ls to 224.0.0.5, intf addr 10.1.1.1, age 0, auth 0
OSPF: send_grace_ls to 224.0.0.5, intf addr 10.0.0.1, age 0, auth 0
OSPF: send_grace_ls to 224.0.0.5, intf addr 10.0.0.1, age 0, auth 0
OSPF: send_grace_ls to 224.0.0.5, intf addr 10.0.0.1, age 0, auth 0
OSPF: send_grace_ls to 224.0.0.5, intf addr 10.1.1.1, age 0, auth 0
OSPF: send_grace_ls to 224.0.0.5, intf addr 10.0.0.1, age 0, auth 0
OSPF: send_grace_ls to 224.0.0.5, intf addr 10.0.0.1, age 0, auth 0
OSPF: send_grace_ls to 224.0.0.5, intf addr 10.0.0.1, age 0, auth 0
OSPF: Graceful Restart setup, waiting for 2 (0) peers

SYSLOG: Feb 1 23:59:20:<14>XMR1, System: Interface ethernetmgmt1, state up
ipc_send_mp_red_active_boot_info: reboot_needed = 0
Start code flash synchronization to standby MP.
Code flash synchronization to standby MP is done.
OSPF: GR no waiting from neighbor 10.0.0.2, interface state Waiting, DR 10.0.0.2
OSPF: Graceful Restart (1) SPF waiting for 1 (0) neighbors, 116 sec
OSPF: Graceful Restart (1) SPF waiting for 1 (0) neighbors, 115 sec
OSPF: Graceful Restart (1) SPF waiting for 1 (0) neighbors, 115 sec
OSPF: GR no waiting from neighbor 10.0.0.2, interface state Waiting, DR 10.0.0.2
OSPF: Graceful Restart (1) SPF waiting for 1 (0) neighbors, 115 sec
OSPF: Graceful Restart (1) SPF waiting for 1 (0) neighbors, 115 sec
OSPF: Graceful Restart (1) SPF waiting for 1 (0) neighbors, 114 sec
OSPF: Graceful Restart (1) SPF waiting for 1 (0) neighbors, 114 sec
OSPF: Graceful Restart (1) SPF waiting for 1 (0) neighbors, 113 sec
OSPF: Graceful Restart (1) SPF waiting for 1 (0) neighbors, 112 sec
OSPF: Graceful Restart (1) SPF waiting for 1 (0) neighbors, 112 sec
Start running config synchronization to standby MP.

Running config synchronization to standby MP is done.
OSPF: Graceful Restart (1) SPF waiting for 1 (0) neighbors, 112 sec
OSPF: Graceful Restart (1) SPF waiting for 1 (0) neighbors, 112 sec
OSPF: Graceful Restart (1) SPF waiting for 1 (0) neighbors, 111 sec
OSPF: Graceful Restart (1) SPF waiting for 1 (0) neighbors, 111 sec
OSPF: Graceful Restart (1) SPF waiting for 1 (0) neighbors, 110 sec
OSPF: Graceful Restart all none-VI neighbors back to FULL state

OSPF: GR restart phase neighbor connect Done, neighbor 0 (0), abort 0
OSPF: Graceful Restart phase neighbor full Done
OSPF: Graceful Restart phase VI neighbor full done
OSPF: restart_detected = OSPF_RESTART_STATE_SPF_ORIGINATE_LSA
OSPF: Graceful Restart reoriginate router LSA, restart_detected =
OSPF_RESTART_STATE_SPF_ORIGINATE_LSA
OSPF: restart_detected = OSPF_RESTART_STATE_SPF_ORIGINATE_LSA
OSPF: restart_detected = OSPF_RESTART_STATE_SPF_ORIGINATE_LSA
OSPF: Graceful Restart  originated router/network LSAs
OSPF: Graceful restart: start SPF
RTM: switch over done for protocol ospf
RTM: switch over done for ALL protocol
OSPF: Graceful Restart phase originate lsa DONE
OSPF: originate grace LSA, interface 10.1.1.1
OSPF: send_grace_ls to 224.0.0.5,intf addr 10.1.1.1, age 3600, auth 0
OSPF: originate grace LSA, interface 10.0.0.1
OSPF: send_grace_ls to 224.0.0.5,intf addr 10.0.0.1, age 3600, auth 0
OSPF: originate grace LSA, interface 10.0.0.1
OSPF: send_grace_ls to 224.0.0.5,intf addr 10.0.0.1, age 3600, auth 0
OSPF: originate grace LSA, interface 10.0.0.1
OSPF: send_grace_ls to 224.0.0.5,intf addr 10.0.0.1, age 3600, auth 0
```

```
OSPF: Graceful Restart phase flush lsa DONE
Standby MP is ready
OSPF: GR restart phase neighbor connect Done, neighbor 0 (0), abort 0
OSPF: Graceful Restart phase neighbor full Done
OSPF: Graceful Restart phase VI neighbor full done
OSPF: restart_detected = OSPF_RESTART_STATE_SPF_ORIGINATE_LSA
OSPF: Graceful Restart reoriginate router LSA, restart_detected =
OSPF_RESTART_STATE_SPF_ORIGINATE_LSA
OSPF: restart_detected = OSPF_RESTART_STATE_SPF_ORIGINATE_LSA
OSPF: restart_detected = OSPF_RESTART_STATE_SPF_ORIGINATE_LSA
OSPF: Graceful Restart  originated router/network LSAs
OSPF: Graceful restart: start SPF
RTM: switch over done for protocol ospf
RTM: switch over done for ALL protocol
OSPF: Graceful Restart phase originate lsa DONE
OSPF: originate grace LSA, interface 10.1.1.1
OSPF: send_grace_ls to 224.0.0.5,intf addr 10.1.1.1, age 3600, auth 0
OSPF: originate grace LSA, interface 10.0.0.1
OSPF: send_grace_ls to 224.0.0.5,intf addr 10.0.0.1, age 3600, auth 0
OSPF: originate grace LSA, interface 10.0.0.1
OSPF: send_grace_ls to 224.0.0.5,intf addr 10.0.0.1, age 3600, auth 0
OSPF: originate grace LSA, interface 10.0.0.1
OSPF: send_grace_ls to 224.0.0.5,intf addr 10.0.0.1, age 3600, auth 0
OSPF: Graceful Restart phase flush lsa DONE
Standby MP is ready
```

The following example shows output from a graceful restart on a helper router.

```
OSPF: rcv GRACE LSA from 10.0.0.1, age 0, Adv 10.1.1.1
OSPF:  install new GraceLSA, int 0, neighbor 10.0.0.1, age 0
OSPF: rcv Grace_LSA from 10.0.0.1, area 0
OSPF:  neighbor 10.0.0.1 entering graceful restart state, timer 120, lsa age 0,
max 120, helping 0
OSPF: flood grace LSA, AdvRtr:10.1.1.1,  Age:0
OSPF: rcv GRACE LSA from 10.0.0.1, age 0, Adv 10.1.1.1

SYSLOG: Dec 15 17:29:43:<13>XMR2, OSPF: nbr state changed, rid 10.2.2.2, nbr addr
10.0.0.1, nbr rid 10.1.1.1, state full
OSPF: rcv GRACE LSA from 10.0.0.1, age 3600, Adv 10.1.1.1
OSPF: LSA flush rcvd Type:9 AdvRtr:10.1.1.1 LsId:10.0.0.0
OSPF:  install new GraceLSA, int 0, neighbor 10.0.0.1, age 3600
OSPF: rcv Grace_LSA from 10.0.0.1, area 0
OSPF:  neighbor 10.0.0.1 exiting graceful restart state, timer 120, lsa age 3600,
max 3600,
helping 0
OSPF: flood grace LSA, AdvRtr:10.1.1.1,  Age:3600
OSPF: age out GraceLSA, from 10.1.1.1, age 3600
OSPF: remove grace LSA, age 3600
```

debug ip ospf log-debug-message

Syntax: [no] debug ip ospf log-debug-message

This command logs instances when large (greater than MTU) LSA update messages are sent or received. Command output resembles the following example.

```
Brocade# debug ip ospf log-debug-message
         OSPF:  debug-message logging debugging is on
SYSLOG: <13>Dec 10 23:55:28 R3 OSPF: nbr state changed, rid 10.13.13.13, nbr addr
10.1.1.2, nbr rid 10.15.15.15, state down
```

```
SYSLOG: <13>Dec 10 23:55:28 R3 OSPF: interface state changed, rid 10.13.13.13,
intf addr 10.1.1.1, state designated router
SYSLOG: <13>Dec 10 23:55:28 R3 OSPF: nbr state changed, rid 10.13.13.13, nbr addr
10.101.1.2, nbr rid 10.15.15.15, state down
SYSLOG: <13>Dec 10 23:55:28 R3 OSPF: interface state changed, rid 10.13.13.13,
intf addr 10.101.1.1, state designated router
```

debug ip ospf log-empty-lsa

Syntax: [no] debug ip ospf log-empty-lsa

This command logs instances when empty or truncated LSA update messages are sent or received. Command output resembles the following example.

```
Brocade# debug ip ospf log-empty-lsa
         OSPF:  empty-LSA logging debugging is on
SYSLOG: <13>Dec 10 23:56:45 R3 OSPF: nbr state changed, rid 10.13.13.13, nbr addr
10.1.1.2, nbr rid 10.15.15.15, state down
SYSLOG: <13>Dec 10 23:56:45 R3 OSPF: interface state changed, rid 10.13.13.13,
intf addr 10.1.1.1, state designated router
SYSLOG: <13>Dec 10 23:56:45 R3 OSPF: nbr state changed, rid 10.13.13.13, nbr addr
10.101.1.2, nbr rid 10.15.15.15, state down
SYSLOG: <13>Dec 10 23:56:45 R3 OSPF: interface state changed, rid 10.13.13.13,
intf addr 10.101.1.1, state designated router
SYSLOG: <13>Dec 10 23:56:47 R3 OSPF: interface state changed, rid 10.13.13.13,
intf addr 10.1.1.1, state backup designated router
SYSLOG: <13>Dec 10 23:56:47 R3 OSPF: nbr state changed, rid 10.13.13.13, nbr addr
10.1.1.2, nbr rid 10.15.15.15, state full
SYSLOG: <13>Dec 10 23:56:47 R3 OSPF: intf rcvd bad pkt: Cannot associate neighbor,
rid 10.13.13.13, intf addr 10.101.1.1, pkt size 120, checksum 41856, pkt src addr
10.101.1.2, pkt type link state update
SYSLOG: <13>Dec 10 23:56:48 R3 OSPF: intf rcvd bad pkt: Cannot associate neighbor,
rid 10.13.13.13, intf addr 10.101.1.1, pkt size 60, checksum 48577, pkt src addr
10.101.1.2, pkt type link state update
SYSLOG: <13>Dec 10 23:56:52 R3 OSPF: intf rcvd bad pkt: Cannot associate neighbor,
rid 10.13.13.13, intf addr 10.101.1.1, pkt size 88, checksum 50589, pkt src addr
10.101.1.2, pkt type link state update
```

debug ip ospf lsa-filtering

Syntax: [no] debug ip ospf lsa-filtering

This command displays type-3 LSAs as and when a router generates and filters these LSAs as shown in the following example.

```
Brocade# debug ip ospf lsa-filtering
Jan 18 17:14:39 Prefix Fitering IN Prefix = 10.0.10.0, Mask = 255.255.255.0, Area
= 0
Jan 18 17:14:39 Action: DENY
Jan 18 17:14:39 Prefix Fitering IN Prefix = 10.30.30.0, Mask = 255.255.255.0, Area
= 0
Jan 18 17:14:39 Action: DENY
Jan 18 17:14:39 Prefix Fitering IN Prefix = 10.30.31.0, Mask = 255.255.255.0, Area
= 0
Jan 18 17:14:39 Action: PERMIT
```

debug ip ospf lsa-generation

Syntax: [no] debug ip ospf lsa-generation

This command generates information about LSAs in output similar to the following example.

```
Brocade# debug ip ospf lsa-generation
Jan 15 16:35:25 OSPF: install a new lsa, type 3, ls_id 0.0.0.0, age 0, seq
80000003 area-id 10
Jan 15 16:35:25 OSPF: NSR : Sync node add, type 3, ls_id 0.0.0.0, age 0, seq
80000003
Jan 15 16:35:25 OSPF: originate router LSA, area 0
Jan 15 16:35:25 OSPF: install a new lsa, type 1, ls_id 10.2.2.2, age 0, seq
80000004 area-id 0
Jan 15 16:35:25 OSPF: NSR : Sync node add, type 1, ls_id 10.2.2.2, age 0, seq
80000004
Jan 15 16:35:25 OSPF:NSR Sync ACK received for LSA
Jan 15 16:35:25 OSPF:ls_header.id 0.0.0.0 type 3 ToBesyncedState 2
Jan 15 16:35:25 OSPF:NSR Sync ACK received for LSA
Jan 15 16:35:25 OSPF:ls_header.id 10.2.2.2 type 1 ToBesyncedState 2
Jan 15 16:35:25 OSPF: install a new lsa, type 3, ls_id 10.0.0.0, age 0, seq
80000001 area-id 1
Jan 15 16:35:25 OSPF: NSR : Sync node add, type 3, ls_id 10.0.0.0, age 0, seq
80000001
Jan 15 16:35:25 OSPF:NSR Sync ACK received for LSA
Jan 15 16:35:25 OSPF:ls_header.id 10.0.0.0 type 3 ToBesyncedState 2
Jan 15 16:35:28 OSPF: redistribute into ospf 10.0.0.0 with ffffff00 forwarding
address 0.0.0.0
Jan 15 16:35:28 OSPF: originate external lsa 10.0.0.0 with ffffff00
Jan 15 16:35:28 OSPF: install a new lsa, type 5, ls_id 10.0.0.0, age 0, seq
80000001 area-id 0
Jan 15 16:35:28 OSPF: NSR : Sync node add, type 5, ls_id 10.0.0.0, age 0,
seq80000001
Jan 15 16:35:29 OSPF:NSR Sync ACK received for LSA
Jan 15 16:35:29 OSPF:ls_header.id 10.0.0.0 type 5 ToBesyncedState2
```

This output indicates that the sequence number (seq) is a unique identifier for each LSA. When a router initiates an LSA, it includes a sequence number, which is recorded in the link state database of every receiving router. If a router receives an LSA that is already in the database and has the same sequence number, the received LSA is discarded. If the information is the same but the sequence number is greater, the LSA information and new sequence number are entered into the database and the LSA is flooded. Sequence numbers allow LSA flooding to stop when all routers have received the most recent LSA.

NOTE
This command can be enabled on the standby MP as well.

debug ip ospf packet

Syntax: [**no**] **debug ip ospf packet**

This command generates information about OSPF packets. Command output resembles the following example.

```
Brocade# debug ip ospf packet
OSPF: recv from:10.28.172.17 Intf:eth 2/1 Hello  L:44 A:2 Rid:10.28.156.234
DR:10.28.172.17 BDR:0.0.0.0
```

This output describes an OSPF packet received from Ethernet interface 2/1 on router 10.28.172.17. Descriptors include: L = packet length, A = authentication, Rid = router ID, DR = designated router ID, and BDR= backup designated router ID.

debug ip ospf retransmission

Syntax: [**no**] **debug ip ospf retransmission**

This command generates internal information about OSPF retransmission of LSAs. Command output resembles the following example.

```
Brocade# debug ip ospf retransmission
OSPF: retransmission debugging is on
OSPF: examine each neighbor and add advertisement to the retransmission list if
necessary
OSPF: remove current database copy from all neighbors retransmission lists
```

debug ip ospf route

Syntax: [**no**] **debug ip ospf route** *ip-addr*

This command generates network-specific information during Dijkstra computation, routing table calculation, and LSA origination. This command is useful for tracking a specific OSPF prefix. Command output resembles the following example.

```
Brocade# debug ip ospf route 10.1.1.1
OSPF: debug ospf route 10.1.1.1
OSPF: Orig summary LSA to area 0, route 10.1.1.1, type 1, prem 1...
OSPF: Originating type 4 summary LSA to area 0, route 10.1.1.1
OSPF: Orig summary LSA to area 1, route 10.1.1.1, type 1, prem 1...
OSPF: Orig summary LSA to area 3, route 10.1.1.1, type 1, prem 1...
OSPF: delete route 10.1.1.1 from rtm 0x053742b0, not_in_main 0
```

debug ip ospf sham-link

Syntax: [**no**] **debug ip ospf sham-link**

This command generates information about OSPF sham-links. A sham-link is required between any two VPN sites that belong to the same OSPF area and share an OSPF backdoor link. If no backdoor link exists between the sites, no sham-link is required.

debug ip ospf shortcuts

Syntax: [**no**] **debug ip ospf shortcuts**

This command generates information about OSPF shortcuts for IP over MPLS. Command output resembles the following example.

```
Brocade# debug ip ospf shortcuts
OSPF: Clearing OSPF DSPT Route Table, num of entries 5
OSPF: Clearing OSPF DSPT Route Table completed, num of entries 5
```

debug ip ospf spf

Syntax: [**no**] **debug ip ospf spf**

This command generates information about OSPF SPF activity including SPF runs and calculations. Command output resembles the following example.

```
Brocade# debug ip ospf spf
         OSPF:  spf-short debugging is on
Dec 10 20:20:50 OSPF: Schedule SPF(12001), in prog 0, ospf build_routing_table 0
phase 1
Dec 10 20:20:50 OSPF: schedule spf, init spf delay 0, next hold 0 (ticks)
```

```
Dec 10 20:20:50 OSPF: Add to spf pending list, current time 2983323, scheduled
2983323, next run 2983323
Dec 10 20:20:50 OSPF: timer: give semaphore, start spf phase 1, time 2983323,
scheduled 2983323, run time 2983323
Dec 10 20:20:50 OSPF: begin intra SPF run, chunk-id 00000000/-1 just_become_abr 0,
is_abr 0
Dec 10 20:20:50 OSPF: invalidate whole routing table, recal_just_become_abr 0,
just_become_abr 0
Dec 10 20:20:50 OSPF: completed SPF for all areas
Dec 10 20:20:50 OSPF: ROUTE CALC PHASE_INTRA end at 2983323, is_abr 0
Dec 10 20:20:50 OSPF: ROUTE CALC PHASE_TRANSIT end at 2983323
Dec 10 20:20:50 OSPF: ROUTE CALC PHASE_TYPE5 end at 2983323
Dec 10 20:20:50 OSPF: ROUTE CALC PHASE_TYPE7 end at 2983323
Dec 10 20:20:50 OSPF: summary phase, is_abr 0
Dec 10 20:20:50 OSPF: ROUTE CALC PHASE_SUMMARY end at 2983323
Dec 10 20:20:50 OSPF: No tunnel phase
Dec 10 20:20:50 OSPF: ROUTE CALC PHASE_NO_TNNL end at 2983323
Dec 10 20:20:50 OSPF: translation phase, is_abr 0
Dec 10 20:20:50 OSPF: ROUTE CALC PHASE_TRANSLATION end at 2983323
Dec 10 20:20:50 OSPF: SPF_cleanup: current 2983323, set next run time 2983323,
current hold 0, next hold 0
Dec 10 20:20:50 OSPF: ROUTE CALC end at 2983323, pending 0
Dec 10 20:20:53 OSPF: Schedule SPF(12001), in prog 0, ospf build_routing_table 0
phase 1
Dec 10 20:20:53 OSPF: schedule spf, init spf delay 0, next hold 0 (ticks)
Dec 10 20:20:53 OSPF: Add to spf pending list, current time 2983393, scheduled
2983393, next run 2983393
Dec 10 20:20:53 OSPF: Schedule SPF(12002), in prog 0, ospf build_routing_table 0
phase 1
Dec 10 20:20:54 OSPF: timer: give semaphore, start spf phase 1, time 2983395,
scheduled 2983393, run time 2983393
Dec 10 20:20:54 OSPF: begin intra SPF run, chunk-id 00000000/-1 just_become_abr 0,
is_abr 0
Dec 10 20:20:54 OSPF: invalidate whole routing table, recal_just_become_abr 0,
just_become_abr 0
Dec 10 20:20:54 OSPF: completed SPF for all areas
```

IPv6 OSPF debug commands

This section describes the debug commands used for monitoring the IPv6 OSPF environment.

debug ipv6 ospf

Syntax: [**no**] **debug ipv6 ospf** [**bfd** | **gr-helper** | **ism** | **ism-events** | **ism-status** | **lsa** | **lsa-flooding** | **lsa-generation** | **lsa-install** | **lsa-maxage** | **lsa-refresh** | **nsm** | **nsm-events** | **nsm-status** | **packet** | **packet-dd** | **packet-hello** | **packet-lsa-ack** | **packet-lsa-req** | **packet-lsa-update** | **route** | **route-calc-external** | **route-calc-inter-area** | **route-calc-intra-area** | **route-calc-spf** | **route-calc-transit** | **route-install** | **virtual-link**]

- **bfd** - Displays information about OSPFv3 BFD events.
- **gr-helper** - Displays information about graceful restart (GR) helper operation.
- **ism** - Displays debug information about the ISM.
- **ism-events** - Displays events on the ISM.
- **ism-status** - Displays status of the ISM.
- **lsa** - Displays LSAs.

- **lsa-flooding** - Displays LSA-flooding activity.
- **lsa-generation** - Displays information about LSA generation.
- **lsa-install** - Displays installed LSAs.
- **lsa-maxage** - Displays the maximum aging information for LSAs.
- **lsa-refresh** - Displays LSA refresh information.
- **nsm** - Displays information about the NSM.
- **nsm-events** - Displays event information for the NSM.
- **nsm-status** - Displays NSM status information.
- **packet** - Displays all OSPFv3 packets in rx or tx mode.
- **packet-dd** - Displays all OSPFv3 data description packets in rx or tx mode.
- **packet-hello** - Displays all OSPFv3 hello packets in rx or tx mode.
- **packet-lsa-ack** - Displays all OSPFv3 LSA ACK packets in rx or tx mode.
- **packet-lsa-req** - Displays all OSPFv3 LSA request packets in rx or tx mode.
- **packet-lsa-update** - Displays all OSPFv3 LSA update packets in rx or tx mode.
- **route** - Displays all OSPFv3 routes.
- **route-calc-external** - Displays external route calculations.
- **route-calc-inter-area** - Displays inter-area route calculations.
- **route-calc-intra-area** - Displays intra-area route calculations.
- **route-calc-spf** - Displays SPF route calculations.
- **route-calc-transit** - Displays transit route calculations.
- **route-install** - Displays all OSPFv3 routes installed.
- **virtual-link** - Displays all OSPFv3 virtual links.

The **debug ipv6 ospf** command displays information about OSPF activity, including interface state machine (ISM), neighbor state machine (NSM), and link state advertisement (LSA) data, packets, routes, and virtual links.

debug ipv6 ospf gr-helper

Syntax: [**no**] **debug ipv6 ospf gr-helper**

This command displays information about graceful restart (GR) helper operation. Command output resembles the following example.

```
Brocade# debug ipv6 ospf gr-helper
May 10 15:33:48.860 OSPFv3: GR HELPER: Received type:Grace Lsa Id:10.0.0.2 Adv
Router:10.0.0.7
May 10 15:33:48.860 OSPFv3: GR HELPER: Entering gr helper for nbr 10.0.0.7 on
interface eth 21/19
May 10 15:34:14.535 OSPFv3: GR HELPER: Received type:Grace Lsa Id:10.0.0.2 Adv
Router:10.0.0.7
May 10 15:34:14.535 OSPFv3: GR HELPER: Successfully exiting gr helper for the nbr
10.0.0.7 on interface eth 21/19
```

debug ipv6 ospf ism

Syntax: [**no**] **debug ipv6 ospf ism**

This command generates comprehensive information about OSPF ISM status changes. Command output resembles the following example.

```
Brocade# debug ipv6 ospf ism
OSPFv3 ISM[137]: IntefaceUp
OSPFv3 ISM[137]: Status change Down -> Waiting (Priority > 0)
OSPFv3 ISM[137]: BackupSeen
OSPFv3 ISM[137]: Status change Waiting -> BDR (BackupSeen:DR Election)
OSPFv3 ISM[137]: (dr:0.0.0.0,bdr:0.0.0.0) -> (dr:10.2.2.2,bdr:10.2.3.4)
```

This output indicates a status change for ISM 137, from up to down to waiting. A switch from the designated router (DR) to the backup designated router (BDR) has also occurred.

debug ipv6 ospf ism-events

Syntax: [no] debug ipv6 ospf ism-events

This command displays IPv6 OSPF interface state machine (ISM) activity, such as an interface coming up or going down. Command output resembles the following example.

```
Brocade# debug ipv6 ospf ism-events
OSPFv3 ISM[137]: Interfaces
OSPFv3 ISM[137]: BackupSeen goes up
```

debug ipv6 ospf ism-status

Syntax: [no] debug ipv6 ospf ism-status

This command displays IPv6 OSPF ISM status information. Command output resembles the following example.

```
Brocade# debug ipv6 ospf ism-status
OSPFv3 ISM[137]: Status change Down -> Waiting (Priority > 0)
OSPFv3 ISM[137]: Status change Waiting -> BDR (BackupSeen, DR Election)
OSPFv3 ISM[137]: (dr:0.0.0.0,bdr:0.0.0.0) -> (dr:10.2.2.2,bdr 10.2.3.4)
```

This output indicates that ISM 137 has gone down and is waiting for a switch from the designated router (DR) to the backup designated router (BDR).

debug ipv6 ospf lsa

Syntax: [no] debug ipv6 ospf lsa

This command displays information about OSPF LSAs. Command output resembles the following example.

```
Brocade# debug ipv6 ospf lsa
OSPFv3 LSA Update Intra-Area-Prefix (Stub): Checking Interface 137
OSPFv3 LSA Update Intra-Area-Prefix (Stub): Interface 137 is down
OSPFv3 LSA Update Intra-Area-Prefix (Stub): No prefix to advertise for Area
0.0.0.0
OSPFv3 LSA Update Intra-Area-Prefix (Stub): Area 0.0.0.0
OSPFv3 ISM (137): Status change Down -> Waiting (Priority > 0)
OSPFv3 LSA: Create LSA Type :Router id: 0 Advrouter:10.2.3.4
OSPFv3 LSA Update Intra-Area-Prefix (Stub): Checking Interface 137
OSPFv3 LSA Update Intra-Area-Prefix (Stub): Include 2001:DB8:1::2/64
OSPFv3 LSA: Create LSA Type :Router Id: 0 Advrouter: 10.2.3.4
OSPFv3 :LSA Update Intra-Area Prefix (Stub): Area 0.0.0.0
OSPFv3 :LSA Update Link: Interface 137
OSPFv3 LSA: Create LSA Type :Link id: 137 Advrouter: 10.2.3.4
```

debug ipv6 ospf lsa-flooding

Syntax: [**no**] **debug ipv6 ospf lsa-flooding**

This command displays IPv6 OSPF LSA flooding activity. Command output resembles the following example.

```
Brocade# debug ipv6 ospf lsa-flooding
OSPFV3:LSA: schedule flooding 10.2.2.2
OSPFV3:LSA: schedule flooding 10.2.2.2
OSPFV3:LSA: schedule flooding 10.2.2.2
OSPFV3:LSA: schedule flooding 10.2.2.2
```

debug ipv6 ospf lsa-generation

Syntax: [**no**] **debug ipv6 ospf lsa-generation**

This command shows additions or deletions of LSAs from the link state database. Command output resembles the following example.

```
Brocade# debug ipv6 ospf lsa-generation
OSPFV3 LSA: Create LSA Type :Router Id: 0 Advrouter:10.2.3.4
OSPFV3 LSA: Create LSA Type :IntraPrefix Id: 0 Advrouter: 10.2.3.4
OSPFV3 LSA: Delete LSA Type: Link Id: 137 Advrouter 10.2.3.4
OSPFV3 LSA: Create LSA Header Type: Router Id: 0 Advrouter: 10.2.3.4
OSPFV3 LSA: Create LSA Header Type: Router Id: 0 Advrouter: 10.2.2.2
OSPFV3 LSA: Create LSA Header Type: Router Id: 0 Advrouter: 10.2.3.4
```

debug ipv6 ospf lsa-install

Syntax: [**no**] **debug ipv6 ospf lsa-install**

This command generates information about new LSAs that are installed in the link state database. Command output resembles the following example.

```
Brocade# debug ipv6 ospf lsa-install
OSPFv3 LSA: Turnover type: IntraPrefix Lsa Id: 0.0.0.0 Advrouter:10.2.3.4:
contents not changed
OSPFv3 LSA: Turnover type: Router Lsa Id: 0.0.0.0 AdvRouter:10.2.3.4: contents not
changed
OSPFv3 LSA: Turnover type: Router Lsa Id:0.0.0.0 AdvRouer:10.2.3.4: contents
changed
OSPFv3 LSA: Turnover type: IntraPrefix Lsa Id: 0.0.0.0 AdvRouter: 10.2.2.2:
contents changed
```

debug ipv6 ospf lsa-maxage

Syntax: [**no**] **debug ipv6 ospf lsa-maxage**

This command identifies LSAs that are removed from the link state database because the router has not received any updates about the LSA in a specified amount of time. Command output resembles the following example.

```
Brocade# debug ipv6 ospf lsa-maxage
OSPFv3 LSA: Premature aging: Type: Interface, ID : 0, AdvRouter 10.2.3.4
OSPFv3 LSA : Premature aging: Type: IntraPrefix, ID : 0, AdvRouter 1.
```

debug ip rtm

Syntax: [**no**] **debug ip rtm** [*A.B.C.D* | **all** | **errors**| **nexthop**]

- *A.B.C.D* - Displays RTM information for a specified IPv4 address.
- **all** - Displays all RTM information.
- **errors** - Displays RTM errors.
- **nexthop** - Logs various next hop-related events to the console.

This command displays information about the routing table manager (RTM), including changes in the routing table. With **debug ip rtm** enabled, and using the **show ip route** command, output resembles the following examples for specific routing table activity.

```
Brocade# debug ip rtm
IP: rtm debugging is on
Brocade# show ip route
Total number of IP routes: 9
Type Codes - B:BGP D:Connected I:ISIS S:Static R:RIP O:OSPF; Cost - Dist/Metric
Destination Gateway Port Cost Type
...
7 10.11.11.0/24 10.10.10.2 eth 1/1 110/10 O2
10.11.11.0/24 10.10.11.2 eth 1/1 110/10 O2
...
RTM: Remove 10.11.11.0/24 (ospf) from rtm
RTM: un-install 10.11.11.0/24 (ospf) in rtm
```

This example indicates that OSPF route 10.11.11.0/24 has been deleted from the route table.

```
RTM: Add 10.11.11.0/24 (ospf) to rtm, path 2
RTM: install 10.11.11.0/24 (ospf) in rtm
```

Configuration notes

- If a router is to operate as an Autonomous System Boundary Router (ASBR), you must enable the ASBR capability at the system level.
- All router ports must be assigned to one of the defined areas on an OSPF router. When a port is assigned to an area, all corresponding subnets on that port are automatically included in the assignment.
- For NetIron MLX-32 or XMR 32000 systems running application image version 03.6.00 or later, configured for OSPF graceful restart, and intended for use in switchover or hitless upgrades, the OSPF dead interval must be changed to 60 seconds on OSPF interfaces. This ensures that the graceful restart process succeeds without a timeout. For instructions on changing the OSPF dead interval, refer to the *Brocade NetIron Routing Configuration Guide*.
- If a Brocade NetIron XMR and Brocade MLX series router is to function as a graceful restart router, a secondary management module must be installed. If the router functions as a graceful restart helper router only, a second management module is not necessary.
- If you disable OSPF, the Brocade NetIron XMR and Brocade MLX series router removes all OSPF configuration information from the running configuration. In addition, after disabling OSPF, when you save the configuration to the startup configuration file, all configuration information for OSPF is removed from the startup configuration file.

  The CLI displays a warning message similar to the following example.

  ```
  Brocade(config-ospf-router)# no router ospf
  router ospf mode now disabled. All ospf config data will be lost when writing
  to flash!
  ```

If you have disabled OSPF but have not yet saved the startup configuration file and reloaded the software, you can restore the configuration information by re-entering the **router ospf** command to enable the protocol. If you have already saved the startup configuration file and reloaded the software, the information is gone.

If you are testing an OSPF configuration and are likely to disable and re-enable OSPF, you may want to make a backup copy of the startup configuration file before you begin. Then, if you remove the configuration information by saving the configuration after disabling the protocol, you can restore the configuration by saving the backup copy of the startup configuration file onto the flash memory.

Common diagnostic scenarios

- A third-party router and a Brocade router are both receiving bad OSPF packets.

  This indicates that the carrier may be the source of the corruption.
- The graceful restart process times out on a Brocade MLX-32 or NetIron XMR 32000 system.

  For routers that are configured for graceful restart and intended for use in hitless upgrades, you must change the OSPF dead interval to 60 seconds to prevent timeouts.
- There are frequent OSPF link flapping events.

  This issue is resolved by upgrading the software version to include the latest patches.

RPF

Reverse Path Forwarding (RPF) prevents malicious users from spoofing a source IP address. It does this by checking that the source address specified for a packet is received from a network to which the router has access. Packets with invalid source addresses are not forwarded. Packets that fail the RPF test can be logged.

RPF show commands

This section describes the show commands that display RPF information.

show ip interface

Syntax: **show ip interface ethernet** *slotnum/portnum*

The *slotnum/portnum* variable specifies the slot and port number of the Ethernet interface.

This command displays information about RPF configurations and packets that have been dropped because they failed the RPF check, as shown in the following example in bold.

```
Brocade# show ip interface ethernet 7/1
Interface Ethernet 7/1 (384)
   port enabled
   port state: UP
   ip address: 10.2.3.4/8
   Port belongs to VRF: default
   encapsulation: Ethernet, mtu: 1500
   MAC Address 0000.0024.a6c0
   directed-broadcast-forwarding: disabled
   No inbound ip access-list is set
```

```
No outbound ip access-list is set
No Helper Addresses are configured
RPF mode: strict RFP Log: Disabled
376720 unicast RPF drop 36068 unicast RPF suppressed drop
```

NOTE
RPF accounting information is always available through the physical interface, even if the physical port belongs to one or more VEs.

show ip rpf

Syntax: show ip rpf

This command displays information about the RPF statistics check.

```
Brocade# show ip rpf
Global RPF check enabled
RPF-Exclude-Default disabled
All Unicast RPF metro information.
  Current operating mode: LOOSE; loose mode count: 3
  Current urpf fail count log mode: DISABLED, log mode count: 0
  Current Hardware URPF fail mode:
     PPCR0:  soft drop
     PPCR1:  soft drop
     PPCR2:  soft drop
```

show ip next-hop

Syntax: show ip next-hop *ip-addr*

The *ip-addr* variable specifies the next-hop IP address.

This command displays forwarding information for the specified IP address.

```
Brocade# show ip next-hop 10.2.2.2
Forwarding route entry: 10.2.2.2
Software forwarding for matching Prefix: 10.0.0.0/8
  No of paths: 2; CAM index: 0x0002802e (163886);
  HW virtual Next-Hop index in shadow:  0x0200001b (33554459)
  HW real    Next-Hop index in shadow:  0x0000001b (27)

Hardware Forwarding entry as programed:
  No of paths: 2; Type of route entry: ECMP
  Next-Hop index as programed in hw:  0x0000001b (27)

Software Next Hop Entry for address: 0x250532e8
  action: fwd
  Paths: (control plane): 2, (forwarding plane): 2
  Buckets: (as paths seen by lpm): 2, (max allocated): 2
  bucket assignment:(0, 1, 0, 1, 0, 1, 0, 1)
  Virtual HW next-hop id: 0x0200001b (33554459)
  ref count: 1, set id: 0x2c7dbb40

Hardware Next Hop Entry for virtual id: 0x0200001b
PPCR id  real id     cmd            port  tx index   vlan id  urpf mode
0        27          route          2/1   0          20       vlan mode
0        28          route          1/29  1          1        port mode
1        27          route          2/1   0          20       vlan mode
1        28          route          1/29  1          1        port mode
2        27          route          2/1   0          20       vlan mode
```

```
2        28          route          1/29  1          1        port mode

Software ECMP SIP Entry at: 0x2506b000
  Virtual Hardware id: 0x0200001e, Real Hardware id: 30; urpf action: port mode,
SIP filter: disable, SIP Access: 0

  Path  Interface      If no     Port      FID no
  0     ve 20          149       2/1       64
  1     eth 1/29       28        1/29      28

Hardware ECMP SIP Entry for virtual id: 0x0200001e; Real hardware id: 30
  PPCR  URPF Mode  SIP_EN. SIP Acc. ||  ECMP no  Valid  Value  Software
  0     port mode  FALSE   0        ||  0        TRUE   1      Trunk Pri: 2/1
                                    ||  1        TRUE   4      eth 1/5
                                    ||  2        TRUE   1      Trunk Pri: 2/1
                                    ||  3        TRUE   4      eth 1/5
                                    ||  4        TRUE   1      Trunk Pri: 2/1
                                    ||  5        FALSE  0
                                    ||  6        TRUE   1      Trunk Pri: 2/1
                                    ||  7        TRUE   4      eth 1/5
  1     port mode  FALSE   0        ||  0        TRUE   1      Trunk Pri: 2/1
                                    ||  1        TRUE   4      eth 1/5
                                    ||  2        TRUE   1      Trunk Pri: 2/1
                                    ||  3        TRUE   4      eth 1/5
                                    ||  4        TRUE   1      Trunk Pri: 2/1
                                    ||  5        FALSE  0
                                    ||  6        TRUE   1      Trunk Pri: 2/1
                                    ||  7        TRUE   4      eth 1/5
  2     port mode  FALSE   0        ||  0        TRUE   1      Trunk Pri: 2/1
                                    ||  1        TRUE   4      eth 1/5
                                    ||  2        TRUE   1      Trunk Pri: 2/1
                                    ||  3        TRUE   4      eth 1/5
                                    ||  4        TRUE   1      Trunk Pri: 2/1
                                    ||  5        FALSE  0
                                    ||  6        TRUE   1      Trunk Pri: 2/1
                                   || 7        TRUE 4      eth 1/5
```

show logging

Syntax: show logging

If you have enabled the **log** option of the **rpf-mode** command, packet information is saved to the system log. To display the log, enter the **show logging** command, as shown in the following example.

```
Brocade# show logging
Syslog logging: enabled (0 messages dropped, 0 flushes, 1305 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
Dynamic Log Buffer (50 lines):
May 11 12:12:54:I:RPF: Denied 1 packets on port 7/5 tcp 10.4.4.1(0) -> 10.6.7.8(0)
```

NOTE
A maximum of 256 RPF messages are logged per minute.

Clearing RPF statistics

To clear RPF statistics on a specific physical interface, use the following command.

clear ip interface ethernet

Syntax: **clear ip interface ethernet** *slot/port*

RPF debug commands

There are no debug commands specific to RPF.

Configuration notes

- IP packets with the source IP address of 0.0.0.0 will always fail the RPF check.
- If you attempt to enable the global RPF command on a system with incompatible CAM settings, the command will be rejected and you will receive a console message.
- Because the RPF feature requires that the entire IP route table be available in hardware, the feature must work in conjunction with Foundry Direct Routing (FDR). FDR is the default mode of operation for the Brocade NetIron XMR series and Brocade MLX series routers.
- You cannot configure RPF on a physical port that has VRF configured on it, or if the physical port belongs to a virtual interface with a VRF configuration.
- Only RPF loose mode is supported for GRE routes.
- If a default route is present on the router, loose mode will permit all traffic.
- RPF can only be configured at the physical port level. It must not be configured on virtual interfaces.
- The following considerations must be taken into account when configuring Reverse Path Forwarding (RPF) with ECMP routes:
    - For a source IP address matching an ECMP route, RPF will permit the packet if it arrives on any of the next-hop interfaces for that route. For example, if there are two best next hops for a network route 10.11.11.0/24, one pointing to 10.10.10.1 (Gigabit Ethernet 7/1) and the other to 10.10.30.1 (Gigabit Ethernet 7/12), then incoming packets with source address matching 10.11.11.0/24 will be permitted on either Gigabit Ethernet 7/1 or Gigabit Ethernet 7/12.
    - A disadvantage of this configuration is that if some other route shares any of these next hops, the packets with a source IP address matching that route will also be permitted from any of the interfaces associated with those next hops. For example, if 10.12.12.0/24 has the next hop 10.10.10.1, then packets from 10.12.12.0/24 will also be permitted on either Gigabit Ethernet 7/1 or Gigabit Ethernet 7/12.

Common diagnostic scenarios

- The RPF check fails.

  The OSPF best route back to the BSR is different from the interface where PIM is enabled.
- RPF packet drop occurs.

Rate limiting is configured on the port where the drops occurred. The problem is resolved when the configuration is changed.

RIP

Routing Information Protocol (RIP) is an IP route exchange protocol that uses a distance vector (a number representing distance) to measure the cost of a given route. The cost is a distance vector because the cost often is equivalent to the number of router hops between the Brocade device and the destination network. A Brocade device supports the following RIP versions:

- Version 1
- Version 1 compatible with Version 2
- Version 2 (the default)

IPv6 RIP, known as Routing Information Protocol Next Generation (RIPng), functions similarly to IPv4 RIP version 2. RIPng supports IPv6 addresses and prefixes.

RIP debug commands

This section describes the debug commands that display RIP information.

debug ip rip

Syntax: **debug ip rip** [**all-vrfs** | **database** | **events** | **packet** | **trigger** | **vrf**]

- **all-vrfs** - Displays information about all VRFs of the RIP.
- **database** - Displays RIP database related events.
- **events** - Displays RIP events.
- **packet** - Displays RIP packet information.
- **trigger** - Displays information about triggered updates and periodic updates.
- **vrf** - Displays information about a specific VRF of the RIP.

debug ip rip all-vrfs

Syntax: **debug ip rip all-vrfs** [**database** | **events** | **packet** | **trigger**]

This command displays information about all VRFs of the RIP based on the options specified. The following is the sample output from the **debug ip rip all-vrfs database** command.

```
Brocade# debug ip rip all-vrfs database
Feb  7 09:26:23.825 RIP(default-vrf):  Sending update on interface 1/5
Feb  7 09:26:23.825 RIP(default-vrf):        src 10.4.15.3, port 520
Feb  7 09:26:23.825 RIP(default-vrf):        dest 224.0.0.9 (1/5), port 520
Feb  7 09:26:23.825        command response version 2 packet size 52
Feb  7 09:26:23.825        prefix 10.3.3.3/32 metric 1 tag 0 NextHop10.4.15.3
Feb  7 09:26:23.825 RIP(default-vrf):  Sending update on interface lb1
Feb  7 09:26:23.825 RIP(default-vrf):        src 10.3.3.3, port 520
Feb  7 09:26:23.825 RIP(default-vrf):        dest 224.0.0.9 (lb1), port 520
Feb  7 09:26:23.825        command response version 2 packet size 72
Feb  7 09:26:23.825        prefix 10.4.15.0/24 metric 1 tag 0 NextHop10.3.3.3
Feb  7 09:26:23.825        prefix 10.4.4.4/32 metric 2 tag 0 NextHop10.4.15.4
Feb  7 09:26:23.825 RIP(red):  Sending update on interface v22
Feb  7 09:26:23.825 RIP(red):        src 10.1.1.1, port 520
```

```
Feb  7 09:26:23.825 RIP(red):        dest 224.0.0.9 (v22), port 520
Feb  7 09:26:23.825        command response version 2 packet size 52
Feb  7 09:26:23.825        prefix 10.3.3.33/32 metric 1 tag 0 NextHop10.1.1.1
Feb  7 09:26:23.825 RIP(red):  Sending update on interface lb3
Feb  7 09:26:23.825 RIP(red):        src 10.3.3.33, port 520
Feb  7 09:26:23.825 RIP(red):        dest 224.0.0.9 (lb3), port 520
Feb  7 09:26:23.825        command response version 2 packet size 52
Feb  7 09:26:23.825        prefix 10.1.1.0/24 metric 1 tag 0 NextHop10.3.3.3
```

debug ip rip database

Syntax: debug ip rip database

This command displays information about RIP database events. The following is the sample output from the **debug ip rip database** command.

```
Brocade# debug ip rip database
Feb  7 08:48:54.462 RIP(default-vrf):  Adding local connected route 10.3.3.3/32 on
interface lb1
Feb  7 08:48:54.462 RIP(default-vrf):    new 10.3.3.3/32 metric 1 from 0.0.0.0
1793
Feb  7 08:49:22.363 RIP(default-vrf):  Adding local connected route 10.4.15.0/24
on interface 1/5
Feb  7 08:49:22.363 RIP(default-vrf):    new 10.4.15.0/24 metric 1 from 0.0.0.0 4
Feb  7 08:49:22.363 RIP(default-vrf):  (v2) process response packet
Feb  7 08:49:22.363     header: type:RESPONSE PACKET, version:2
Feb  7 08:49:22.363 RIP(default-vrf):    new 10.4.4.4/32 metric 2 from 10.4.15.4 4
Feb  7 08:49:22.363 RIP(default-vrf):  Sending update on interface lb1
Feb  7 08:49:22.364 RIP(default-vrf):        src 10.3.3.3, port 520
Feb  7 08:49:22.364 RIP(default-vrf):        dest 224.0.0.9 (lb1), port 520
Feb  7 08:49:22.364        command response version 2 packet size 72
Feb  7 08:49:22.364        prefix 10.4.15.0/24 metric 1 tag 0 NextHop10.3.3.3
Feb  7 08:49:22.364        prefix 10.4.4.4/32 metric 2 tag 0 NextHop10.4.15.4
Feb  7 08:49:34.973 RIP(default-vrf):  (v2) process response packet
Feb  7 08:49:34.973     header: type:RESPONSE PACKET, version:2
Feb  7 08:49:34.973 RIP(default-vrf):    refresh 10.4.4.4/32 metric 2 from
10.4.15.4 eth 1/5
Feb  7 08:49:34.973 RIP(default-vrf):          existing route metric 2 from
10.4.15.4 eth 1/5
Feb  7 08:49:44.661 RIP(default-vrf):  Sending update on interface 1/5
Feb  7 08:49:44.661 RIP(default-vrf):        src 10.4.15.3, port 520
Feb  7 08:49:44.661 RIP(default-vrf):        dest 224.0.0.9 (1/5), port 520
Feb  7 08:49:44.661        command response version 2 packet size 52
Feb  7 08:49:44.661        prefix 10.3.3.3/32 metric 1 tag 0 NextHop10.4.15.3
```

debug ip rip events

Syntax: debug ip rip events

This command displays information about RIP events. The following is the sample output from the **debug ip rip events** command.

```
Brocade# debug ip rip events
Feb  7 08:57:14.060 RIP(default-vrf):  stop running on interface 1/5
Feb  7 08:57:14.060 RIP(default-vrf):  update timer expired
Brocade(config-if-e1000-1/5)# ip rip v2-only
Feb  7 08:57:16.910 RIP(default-vrf):  start running on interface 1/5
Brocade(config-if-e1000-1/5)# Feb  7 08:57:16.911 RIP(default-vrf):  update timer
expired
Feb  7 08:57:17.409 RIP(default-vrf):  update timer expired
```

debug ip rip packet

Syntax: debug ip rip packet

This command displays information about RIP packets sent and received. The following is the sample output from the **debug ip rip packet** command.

```
Brocade# debug ip rip packet
Feb  7 09:05:14.412 RIP(default-vrf):  send updates(periodic) to 224.0.0.9 via eth
1/5 (10.4.15.4)
Feb  7 09:05:14.412 RIP: build route
Feb  7 09:05:14.412     header: type:RESPONSE PACKET, version:2
Feb  7 09:05:14.412 RIP(default-vrf):  build one route
Feb  7 09:05:14.412 RIP:  route entry: family:2, target:10.4.4.4,
subnet_mask:255.255.255.255, metric:1, next_hop:10.4.15.4, route_tag:0
Feb  7 09:05:14.412 RIP(default-vrf):  send updates(periodic) to 224.0.0.9 via
loopback 1 (10.4.4.4)
Feb  7 09:05:14.412 RIP: build route
Feb  7 09:05:14.412     header: type:RESPONSE PACKET, version:2
Feb  7 09:05:14.412 RIP(default-vrf):  build one route
Feb  7 09:05:14.412 RIP:  route entry: family:2, target:10.3.3.3,
subnet_mask:255.255.255.255, metric:2, next_hop:10.4.15.3, route_tag:0
Feb  7 09:05:14.412 RIP(default-vrf):  build one route
Feb  7 09:05:14.412 RIP:  route entry: family:2, target:10.4.15.0,
subnet_mask:255.255.255.0, metric:1, next_hop:10.4.4.4, route_tag:0
Feb  7 09:05:35.131 RIP(default-vrf):  rcvd updates from 10.4.15.3 on eth 1/5
Feb  7 09:05:35.131 RIP(default-vrf):  received response from 10.4.15.3: 24 bytes
Feb  7 09:05:35.131 RIP:  route entry: family:2, target:10.3.3.3,
subnet_mask:255.255.255.255, metric:1, next_hop:10.4.15.3, route_tag:0
Feb  7 09:05:41.412 RIP(default-vrf):  send updates(periodic) to 224.0.0.9 via eth
1/5 (10.4.15.4)
Feb  7 09:05:41.412 RIP: build route
```

debug ip rip trigger

Syntax: debug ip rip trigger

This command displays information about triggered updates which is being sent, and periodic updates. The following is the sample output from the **debug ip rip trigger** command.

```
Brocade# debug ip rip trigger
Feb  7 09:10:35.964 RIP(default-vrf):  triggered update sent on port 1793
Brocade(config-if-e1000-1/5)# ip rip v2-only
Brocade(config-if-e1000-1/5)# Feb  7 09:10:39.413 RIP(default-vrf):  triggered
update sent on port 4
Feb  7 09:10:39.414 RIP(default-vrf):  triggered update sent on port 1793
Feb  7 09:10:49.413 RIP(default-vrf):  periodic update sent on port 4
Feb  7 09:10:49.414 RIP(default-vrf):  periodic update sent on port 1793
```

debug ip rip vrf

Syntax: debug ip rip vrf *vrf_name* [**database** | **events** | **packet** | **trigger**]

This command displays information about a particular VRF of the RIP for the specified option. The following is the sample output from the **debug ip rip vrf** command when **packet** option is specified.

```
Brocade# debug ip rip vrf red packet
Feb  7 09:18:53.822 RIP(red):  send updates(periodic) to 224.0.0.9 via ve 22
(10.1.1.1)
Feb  7 09:18:53.822 RIP(red):  build one route
```

```
Feb  7 09:18:53.822 RIP:  route entry: family:2, target:10.3.3.33,
subnet_mask:255.255.255.255, metric:1, next_hop:10.1.1.1, route_tag:0
Feb  7 09:18:53.822 RIP(red):  send updates(periodic) to 224.0.0.9 via loopback 3
(10.3.3.33)
Feb  7 09:18:53.822 RIP(red):  build one route
Feb  7 09:18:53.822 RIP:  route entry: family:2, target:10.1.1.0,
subnet_mask:255.255.255.0, metric:1, next_hop:10.3.3.33, route_tag:0
```

RIPng debug commands

This section describes the debug commands that display RIPng information.

debug ipv6 rip

Syntax: debug ipv6 rip [**all-vrfs** | **events** | **receive** | **transmit** | **vrf**]

- **all-vrfs** - Displays information about all VRFs of the RIPng.
- **events** - Displays RIPng events.
- **receive** - Displays received update packets in RIPng.
- **transmit** - Displays transmitted update packets in RIPng.
- **vrf** - Displays information about a specific VRF of the RIPng.

debug ipv6 rip all-vrfs

Syntax: debug ipv6 rip all-vrfs [**events** | **receive** | **transmit**]

This command displays information about all VRFs of the RIPng based on the options specified. The following is the sample output from the **debug ipv6 rip all-vrfs transmit** command.

```
Brocade# debug ipv6 rip all-vrfs transmit
Feb  7 09:48:59.133 RIPng(red): RIPng: update timer expired
Feb  7 09:48:59.133 RIPng(red): RIPng: Sending update on interface v22
Feb  7 09:48:59.133 RIPng(red):        src fe80::21b:edff:fe9f:6f00, port 521
Feb  7 09:48:59.133 RIPng(red):        dest ff02::9 (v22), port 521
Feb  7 09:48:59.133        command response version 1 packet size 24
Feb  7 09:48:59.133        prefix 2001:DB8:3::33/128 metric 1 tag 0
Feb  7 09:49:05.133 RIPng(default-vrf): RIPng: update timer expired
Feb  7 09:49:05.133 RIPng(default-vrf): RIPng: Sending update on interface 1/5
Feb  7 09:49:05.133 RIPng(default-vrf):        src fe80::21b:edff:fe9f:6f04, port
521
Feb  7 09:49:05.133 RIPng(default-vrf):        dest ff02::9 (1/5), port 521
Feb  7 09:49:05.133        command response version 1 packet size 24
Feb  7 09:49:05.133        prefix 2001:DB8:3::3/128 metric 1 tag 0
Feb  7 09:49:18.999 RIPng(default-vrf): RIPng: received packet from
fe80::224:38ff:fe2d:1e04 port 521 on interface 1/5
Feb  7 09:49:18.999        command response version 1 packet size 24
Feb  7 09:49:18.999        prefix 2001:DB8:4::4/128 metric 1 tag 0
Feb  7 09:49:30.133 RIPng(red): RIPng: update timer expired
Feb  7 09:49:30.133 RIPng(red): RIPng: Sending update on interface v22
Feb  7 09:49:30.133 RIPng(red):        src fe80::21b:edff:fe9f:6f00, port 521
Feb  7 09:49:30.133 RIPng(red):        dest ff02::9 (v22), port 521
Feb  7 09:49:30.133        command response version 1 packet size 24
Feb  7 09:49:30.133        prefix 2001:DB8:3::33/128 metric 1 tag 0
```

debug ipv6 rip events

Syntax: debug ipv6 rip events

This command displays information about RIPng events. The following is the sample output from the **debug ipv6 rip events** command.

```
Brocade# debug ipv6 rip events
Feb  7 09:31:25.029 RIPng(default-vrf): RIPng: disable on interface 1/5
Feb  7 09:31:25.029 RIPng(default-vrf): RIPng: stop running on interface 1/5,
disable 0
Feb  7 09:31:25.029 RIPng(default-vrf): RIPng: Removing local connected route
2001:DB8:15::3/64 on interface 1/5
Feb  7 09:31:25.029 RIPng(default-vrf): RIPng: garbage prefix 2001:DB8:15::/64
timer 16, metric 0, tag 674353720
Feb  7 09:31:25.029 RIPng(default-vrf):          from :: on interface Ethernet 1/5
Feb  7 09:31:25.029 RIPng(default-vrf): RIPng: update timer expired
Brocade(config-if-e1000-1/5)# ipv6 rip enable
Feb  7 09:31:29.378 RIPng(default-vrf): RIPng: enable on interface 1/5
Feb  7 09:31:29.378 RIPng(default-vrf): RIPng: start running on interface 1/5
Feb  7 09:31:29.378 RIPng(default-vrf): RIPng: Adding local connected route
2001:DB8:15::3/64 on interface 1/5
Feb  7 09:31:30.076 RIPng(default-vrf): RIPng: triggered update
```

debug ipv6 rip receive

Syntax: debug ipv6 rip receive

This command displays information about RIPng received update packets. The following is the sample output from the **debug ipv6 rip receive** command.

```
Brocade# debug ipv6 rip receive
Feb  7 09:36:15.015 RIPng(default-vrf): RIPng: received packet from
fe80::224:38ff:fe2d:1e04 port 521 on interface 1/5
Feb  7 09:36:15.015        command response version 1 packet size 24
Feb  7 09:36:15.015        prefix 2001:DB8:4::4/128 metric 1 tag 0
Feb  7 09:36:45.014 RIPng(default-vrf): RIPng: received packet from
fe80::224:38ff:fe2d:1e04 port 521 on interface 1/5
Feb  7 09:36:45.014        command response version 1 packet size 24
Feb  7 09:36:45.014        prefix 2001:DB8:4::4/128 metric 1 tag 0
```

debug ipv6 rip transmit

Syntax: debug ipv6 rip transmit

This command displays information about RIPng transmitted update packets. The following is the sample output from the **debug ipv6 rip transmit** command.

```
Brocade# debug ipv6 rip transmit
Feb  7 09:39:28.079 RIPng(default-vrf): RIPng: Sending update on interface 1/5
Feb  7 09:39:28.079 RIPng(default-vrf):        src fe80::21b:edff:fe9f:6f04, port
521
Feb  7 09:39:28.079 RIPng(default-vrf):        dest ff02::9 (1/5), port 521
Feb  7 09:39:28.079        command response version 1 packet size 24
Feb  7 09:39:28.079        prefix 2001:DB8:3::3/128 metric 1 tag 0
Feb  7 09:39:58.079 RIPng(default-vrf): RIPng: Sending update on interface 1/5
Feb  7 09:39:58.079 RIPng(default-vrf):        src fe80::21b:edff:fe9f:6f04, port
521
Feb  7 09:39:58.079 RIPng(default-vrf):        dest ff02::9 (1/5), port 521
Feb  7 09:39:58.079        command response version 1 packet size 24
```

```
Feb  7 09:39:58.079        prefix 2001:DB8:3::3/128 metric 1 tag 0
```

debug ipv6 rip vrf

Syntax: **debug ipv6 rip vrf** *vrf_name* [**events** | **receive** | **transmit**]

This command displays information about a particular VRF of the RIPng for the specified option. The following is the sample output from the **debug ipv6 rip vrf** command when **events** option is specified.

```
Brocade# debug ipv6 rip vrf red events
Feb  7 09:43:11.733 RIPng(red): RIPng: state up on interface lb3
Feb  7 09:43:11.733 RIPng(red): RIPng: start running on interface lb3
Feb  7 09:43:11.733 RIPng(red): RIPng: Adding local connected route
2001:DB8:3::33/128 on interface lb3
Feb  7 09:43:11.733 RIPng(red): RIPng: Adding local connected route
2001:DB8:3::33/128 on interface lb3
Feb  7 09:43:13.131 RIPng(red): RIPng: triggered update
Feb  7 09:43:13.131 RIPng(red): RIPng: Sending update on interface v22
Feb  7 09:43:13.131 RIPng(red):        src fe80::21b:edff:fe9f:6f00, port 521
Feb  7 09:43:13.131 RIPng(red):        dest ff02::9 (v22), port 521
Feb  7 09:43:13.131        command response version 1 packet size 24
Feb  7 09:43:13.131        prefix 2001:DB8:3::33/128 metric 1 tag 0
```

IS-IS

Intermediate System to Intermediate System (IS-IS) is a link-state interior gateway protocol. IS-IS designates an intermediate system (router) as either a Level 1 or Level 2 router. A Level 1 router routes traffic only within the area where it resides. A Level 2 router routes traffic between areas within a routing domain.

NOTE
The Brocade device does not support routing of Connectionless-Mode Network Protocol (CLNP) packets. The Brocade device uses IS-IS for TCP/IP only.

Detailed IS-IS configuration instructions can be found in the *Brocade NetIron Routing Configuration Guide*.

IS-IS show commands

This section describes the show commands that display information about IS-IS activity and configurations. Some of these are regular user commands, and some are debug commands.

show isis

Syntax: **show isis**

This command displays general IPv4 IS-IS information, as shown in the following example.

```
Brocade# show isis
 IS-IS Routing Protocol Operation State: Enabled
IS-Type: Level-1-2
System ID: 0000.0011.1111
Manual area address(es):47
Level-1-2 Database State: On
```

```
Administrative Distance: 115
Maximum Paths: 4
Default redistribution metric: 0
Protocol Routes redistributed into IS-IS: Static
Number of Routes redistributed into IS-IS: 11
Level-1 Auth-mode: None
Level-2 Auth-mode: None
Metric Style Supported for Level-1: Wide
Metric Style Supported for Level-2: Wide
IS-IS Partial SPF Optimizations: Enabled
Timers:
L1 SPF: Max-wait 120s Init-wait 100ms Second-wait 120000ms
L2 SPF: Max-wait 100s Init-wait 100ms Second-wait 100000ms
L1 SPF is not scheduled
L2 SPF is not scheduled
PSPF: Max-wait 120000ms Init-wait 120000ms Second-wait 120000ms
PSPF is not scheduled
 LSP: max-lifetime 1200s, refresh-interval 900s, gen-interval 10s
retransmit-interval 5s, lsp-interval 33ms
SNP: csnp-interval 10s, psnp-interval 2s
Global Hello Padding : Enabled
Global Hello Padding For Point to Point Circuits: Enabled
Ptpt Three Way HandShake Mechanism: Enabled
IS-IS Traffic Engineering Support: Disabled
BFD: Disabled
Interfaces with IPv4 IS-IS configured:
eth 1/1
```

show isis interface

Syntax: show isis interface

This command displays information about IS-IS interfaces, as shown in the following example.

```
Brocade# show isis interface
Total number of IS-IS Interfaces: 2
Interface : gre_tnl 1
   Circuit State: UP Circuit Mode: LEVEL-1-2
   Circuit Type : PTP Passive State: FALSE
   Circuit Number: 0x02, MTU: 1497
   Level-1 Metric: 10, Level-1 Priority: 64
   Level-1 Auth-mode: None
   Level-2 Auth-mode: None
   Level-1 Metric: 10, Level-1 Priority: 50
   Level-1 Hello Interval: 10 Level-1 Hello Multiplier: 3
   Level-1 Designated IS: XMR1-02 Level-1 DIS Changes: 0
   Level-2 Metric: 10, Level-2 Priority: 50
   Level-2 Hello Interval: 10 Level-2 Hello Multiplier: 3
   Level-2 Designated IS: MLX2-02 Level-2 DIS Changes: 0
   Circuit State Changes: 1 Circuit Adjacencies State Changes: 1
   Rejected Adjacencies: 0
   Circuit Authentication L1 failures: 0
   Circuit Authentication L2 failures: 0
   Bad LSPs 0
   Control Messages Sent: 318 Control Messages Received: 229
   IP Enabled: TRUE
   IP Address and Subnet Mask:
   10.50.50.20         10.255.255.0
   IPv6 Enabled: FALSE
```

show isis neighbor

Syntax: show isis neighbor

This command displays information about IS-IS neighbors, as shown in the following example.

```
Brocade# show isis neighbor
Total number of IS-IS Neighbors: 2
System Id   Interface   SNPA           State Holdtime Type Pri StateChgeTime Protocol
R3          eth 1/1     0000.0002.c000 UP    9        ISL2 64  0  :0 :21:47  M-ISIS
R3          eth 1/1     0000.0002.c000 UP    9        ISL1 64  0  :0 :21:47  M-ISIS
R4          eth 1/1     0000.0004.c000 UP    9        ISL2 64  0  :0 :21:47  ISIS
R4          eth 1/1     0000.0004.c000 UP    9        ISL1 64  0  :0 :21:47  ISIS
```

show isis debug

Syntax: show isis debug [**adj-options-order** | **adj-timer** | **child-link-info** | **ip-nexthop-set** | **ipv6-nexthop-set** | **ipv6-pent-level-info** | **link-info** | **lsp-list** | **lsp-timer** | **memory** | **nexthops** | **parent-link-info** |**pent** | **pent-level-info** | **pspf-lsp-list** | **redis** | **route-info** | **summary** | **v6-nexthops** | **v6route-info**]

- **adj-options-order** - Displays IS-IS adjacency options in a label switched path (LSP).
- **adj-timer** - Displays IS-IS adjacency hold timers.
- **child-link-info** - Displays IS-IS child link debugging information.
- **ip-nexthop-set** - Displays IS-IS IP next-hop information.
- **ipv6-nexthop-set** - Displays IS-IS IPv6 next-hop information.
- **ipv6-pent-level-info** - Displays an integrated IS-IS IPv6 level information list associated with path entries.
- **link-info** - Displays IS-IS link debugging information.
- **lsp-list** - Displays IS-IS LSP list debugging information.
- **lsp-timer** - Displays IS-IS LSP hold timers.
- **memory** - Displays IS-IS memory debugging information.
- **nexthops** - Displays IS-IS next-hop lists debugging information.
- **parent-link-info** - Displays IS-IS parent link debugging information.
- **pent** - Displays IS-IS SPD path entries.
- **pent-level-info** - Displays an integrated IS-IS level information list associated with path entries.
- **pspf-lsp-list** - Displays integrated IS-IS PSPF LSP list.
- **redis** - Displays IS-IS redistribution debugging information.
- **route-info** - Displays integrated IS-IS route information list.
- **summary** - Displays a summary of IS-IS debugging information.
- **v6-nexthops** - Displays debugging information for IS-IS next-hop lists.
- **v6route-info** - Displays the IS-IS route information list.

The command output resembles the following example.

```
Brocade# show isis debug
Router-id: 10.140.140.4
Tics: 168833
[SPF: Act:NOT RUNNING Bld:N  Run[N,N]
```

```
Code assertions are ON
Ptpt 3way HandShake Enabled
Manual Area Addresses: 57(1) 00.0057(3) 00.0001(3)
Union Area Addresses: 00.0001(3) 00.0057(3) 57(1)
isis.run_pspf=N, Ir_pspf_leve11=N, Ir_pspf_level2=Y
PSPF bucket count: Reached Maximum
L1 SPF bucket count: 0
L2 SPF bucket count: 0
ISIS IPv4 Default Entry 00000000
ISIS IPv6 Default Entry 00000000
PSPF Newevent=Y, Masklevel=0, Timerstate=R, RemainingTime=400
L1-SPF Newevent=N, Masklevel=0, Timerstate=S, RemainingTime=0, FailCount=0
L2-SPF Newevent=N, Masklevel=0, Timerstate=S, RemainingTime=0, FailCount=0
isis.l1default: 0 isis_ip6.ip6_l1default: 0
IPv4 L1 SPF Uses:Native Topology, L2 SPF Uses:Native Topology
IPv6 L1 SPF Uses:Native Topology, L2 SPF Uses:Native Topology
NSR State: Normal
isis.sync_instance.sync_enabled: TRUE
isis.sync_instance.asi.peer_device_status 2/Ready
```

show isis debug adj-options-order

Syntax: show isis debug adj-options-order

This command displays IS-IS adjacency options in a label switched path (LSP).

```
Brocade# show isis debug adj-options-order
Level-1 List
 LSP-ID Dut2.00-00
        Metric:  10       IS Dut2.01
 LSP-ID Dut2.01-00
        Metric:  0        IS Dut2.00
Level-2 List
 LSP-ID Dut2.00-00
        Metric:  10       IS Dut2.01
 LSP-ID Dut2.01-00
        Metric:  0        IS Dut2.00
        Metric:  0        IS Dut4.00
```

show isis debug adj-timer

Syntax: show isis debug adj-timer

This command displays information about IS-IS adjacency hold timers.

```
Brocade# show isis debug adj-timer
Summary:
WheelTimer: cur_time 371080, cur_slot 280, num_slots 400, active_slots 11
        Callbacks[Tmo=0x08acb064,Print=0x08acb1a0]
        Buckets Callbacks[Ins=0x0849f108,Rem=0x0849f178,GetRdy=0x0849f1d0]
        Ready Queue: Empty
        Total(Rdy+Slots):    BuckNodes 11, ElemNodes 14
        Avg/Active Slot:     BuckNodes 1, ElemNodes 1
        Slot HighWaterMark:  BuckNodes 1, ElemNodes 2
```

show isis debug child-link-info

Syntax: show isis debug child-link-info *system-id*

This command displays all the child link information for a specific router. The *system-id* variable specifies the router name or system ID.

Command output resembles the following example.

```
Brocade# show isis debug child-link-info R1
Link-1
    Parent: R1, Child:  R2
    Link Metric: 10, Total Metric: 10, Link State: Active
Link-2
    Parent: R1, Child:  R3
    Link Metric: 10, Total Metric: 10, Link State: Active
```

show isis debug ip-nexthop-set

Syntax: **show isis debug ip-nexthop-set**

This command displays the IP address of each set of next hops in all sets of next hops. The maximum number of hops in a set is eight.

Command output resembles the following example.

```
Brocade# show isis debug ip-nexthop-set
Set 1 with No Of Nexthops 1 Address 277ac1a4 up since 0  :0 :0 :6
    Nexthop IPAddr 10.1.1.3, Circ-id 0(eth 1/1)
Pent List Pointing to this Nexthop Set
    Pent-Id XMR16.00-00 level-1
    Pent-Id XMR16.01-00 level-1
```

show isis debug ipv6-nexthop-set

Syntax: **show isis debug ipv6-nexthop-set**

This command displays the IPv6 next-hop set, as the following example illustrates.

```
Brocade# show isis debug ipv6-nexthop-set
Set 1 with No Of Nexthops 1 Address 277ac1e0 up since 0  :0 :0 :43
                Nexthop IPAddr fe80::20c:dbff:fef6:3300, Circ-id 0(eth 1/1)
        Pent List Pointing to this Nexthop Set
                Pent-Id XMR16.00-00 level-1
                Pent-Id XMR16.01-00 level-1
```

show isis debug ipv6-pent-level-info

Syntax: **show isis debug ipv6-pent-level-info**

This command displays an integrated IS-IS IPv6 level information list associated with path entries, when IS-IS multi-topology is enabled.

Command output resembles the following example, when IS-IS multi-topology is enabled.

```
Brocade# show isis debug ipv6-pent-level-info
Pent-Id   level metric pref Chg(N-D-MC-PC-IPV4NC-IPV6NC-PSPFC)
 XMR44.00-00 L1      10     1 (0-0-0-0-0-0-0)
2001:DB8::/32 cost:10, Pre:1, Up/Down:0, Credit: 0, flags:Act Ena   IPv6
2001:DB8::/32 cost:10, Pre:1, Up/Down:0, Credit: 0, flags:Act Ena   IPv6
2001:DB8::/32 cost:10, Pre:1, Up/Down:0, Credit: 0, flags:Act Ena   IPv6

Pent is not found 0000.0000.0000.00-00 level 2
```

```
XMR43.00-00 L2       0     1 (0-0-0-0-0-0-0)
```

show isis debug link-info

Syntax: show isis debug link-info

This command displays all the link information that exists in the IS-IS domain, as shown in the following example.

```
Brocade# show isis debug link-info
Link-1
   Parent: R1, Child:  R2
   Link Metric: 10, Total Metric: 10, Link State: Active
Link-2
   Parent: R1, Child:  R3
   Link Metric: 10, Total Metric: 10, Link State: Active
Link-3
   Parent: R2, Child:  R4
   Link Metric: 10, Total Metric: 20, Link State: Active
Link-4
   Parent: R3, Child:  R4
   Link Metric: 15, Total Metric: 25, Link State: In-Active
```

show isis debug lsp-list

Syntax: show isis debug lsp-list

This command displays the number of instances of certain IS-IS items, such as the number of LSPs in each hash table, the number of partial sequence numbers (PSNPs), and so on.

```
Brocade# show isis debug lsp-list
sizeof(LSPI) = 124, LSPI_SIZE = 382, nd_srm_size = 129
LSP Hash L1 Count: 1
LSP Hash L2 Count: 39
LSP Sort L1 Count: 1
LSP Sort L2 Count: 39
LSP PSNP List Count: 0
LSP Tx List Count: 0
LSP Flood Count: 25
```

show isis debug lsp-timer

Syntax: show isis debug lsp-timer

This command displays information about the LSP timer, as shown in the following example.

```
Brocade# show isis debug lsp-timer
Summary:
WheelTimer: cur_time 371574, cur_slot 374, num_slots 400, active_slots 19
        Callbacks[Tmo=0x08ad8438,Print=0x08ad8498]
        Buckets Callbacks[Ins=0x0849f108,Rem=0x0849f178,GetRdy=0x0849f1d0]
        Ready Queue: Empty
        Total(Rdy+Slots):    BuckNodes 19, ElemNodes 32
        Avg/Active Slot:     BuckNodes 1, ElemNodes 1
        Slot HighWaterMark:  BuckNodes 1, ElemNodes 3
```

show isis debug memory

Syntax: show isis debug memory

This command displays various dimensions of memory (in bytes). Primarily, you would note any indication of a failure or error. If you notice errors, then the other items in the display might lead to a part of memory related to the problem.

```
Brocade# show isis debug memory
 Total IS-IS Memory In Use: 3050881
 Total P2 Route Memory In Use: 464928
 Total P2 Other Memory In Use: 0
 Total Memory Allocated: 10569
 Total Memory Allocation Failed: 0
 Total Packet Buffer Allocated: 0
 Total Packet Buffer Allocation Failed: 0
 Errors in freeing memory (bad addr): 0
 Errors in freeing memory (bad pool-id): 0
 Maximum Memory IS-IS allowed to use: 104857600
```

show isis debug memory pool

Syntax: show isis debug memory pool

This command displays information about the memory pool, as shown in the following example.

```
Brocade# show isis debug memory pool
Pool# 0 @ 0x0928ef4c
blk_size: 292,  initial_blk_cnt: 60,  exp_blk_cnt: 20
curr_#_of_blks 60,  #_of_sub_pools: 1
#_of_blks_in_use: 3,  #_of_blks_free: 57,  #_of_mem_alloc_failed: 0
total_memory_allocated_for_this_pool: 17524
sptr_memory_list: 0x28ca6370, &0x0928ef70
sptr_sub_pool_list: 0x28ca6000 &0x0928ef74
Pool# 1 @ 0x0928ef78
blk_size: 68,  initial_blk_cnt: 60,  exp_blk_cnt: 20
curr_#_of_blks 60,  #_of_sub_pools: 1
#_of_blks_in_use: 0,  #_of_blks_free: 60,  #_of_mem_alloc_failed: 0
total_memory_allocated_for_this_pool: 4084
sptr_memory_list: 0x28cab004, &0x0928ef9c
sptr_sub_pool_list: 0x28cab000 &0x0928efa0
Pool# 2 @ 0x0928efa4
blk_size: 44,  initial_blk_cnt: 1000,  exp_blk_cnt: 250
curr_#_of_blks 1000,  #_of_sub_pools: 1
#_of_blks_in_use: 1,  #_of_blks_free: 1000,  #_of_mem_alloc_failed: 0
total_memory_allocated_for_this_pool: 44004
sptr_memory_list: 0x28cac004, &0x0928efc8
sptr_sub_pool_list: 0x28cac000 &0x0928efcc
Pool# 3 @ 0x0928efd0
blk_size: 285,  initial_blk_cnt: 60,  exp_blk_cnt: 20
curr_#_of_blks 60,  #_of_sub_pools: 1
#_of_blks_in_use: 1,  #_of_blks_free: 59,  #_of_mem_alloc_failed: 0
total_memory_allocated_for_this_pool: 17104
sptr_memory_list: 0x28cb7121, &0x0928eff4
sptr_sub_pool_list: 0x28cb7000 &0x0928eff8
Pool# 4 @ 0x0928effc
blk_size: 471,  initial_blk_cnt: 256,  exp_blk_cnt: 20
curr_#_of_blks 256,  #_of_sub_pools: 1
#_of_blks_in_use: 4,  #_of_blks_free: 252,  #_of_mem_alloc_failed: 0
total_memory_allocated_for_this_pool: 120580
sptr_memory_list: 0x28cbc760, &0x0928f020
sptr_sub_pool_list: 0x28cbc000 &0x0928f024
Pool# 5 @ 0x0928f028
blk_size: 121,  initial_blk_cnt: 1024,  exp_blk_cnt: 20
```

```
curr_#_of_blks 1024,  #_of_sub_pools: 1
#_of_blks_in_use: 0,  #_of_blks_free: 1024,  #_of_mem_alloc_failed: 0
total_memory_allocated_for_this_pool: 123908
sptr_memory_list: 0x28cda004, &0x0928f04c
sptr_sub_pool_list: 0x28cda000 &0x0928f050
```

show isis debug nexthops

Syntax: show isis debug nexthops

This command displays all of the next hops available on the system, as shown in the following example.

```
Brocade# show isis debug nexthops
IS-IS IP Nexthops List:
  Node (2185b00c) -> Nexthop (29124050) ref 3, If 158(eth 4/15) Addr 10.0.0.38
  Node (2185b030) -> Nexthop (291240c0) ref 3, If 146(eth 4/3) Addr 10.0.0.14
  Node (2185b0b4) -> Nexthop (291240e0) ref 3, If 42(eth 1/43) Addr 10.1.6.2
  Node (2185b06c) -> Nexthop (29124090) ref 1, If 14(eth 1/15) Addr 10.1.1.2
  Node (2185b060) -> Nexthop (291240a0) ref 1, If 150(eth 4/7) Addr 10.0.0.30
  Node (2185b03c) -> Nexthop (291240b0) ref 1, If 154(eth 4/11) Addr 10.0.0.18
  Node (2185b0e4) -> Nexthop (29124100) ref 1, If 157(eth 4/14) Addr 10.0.0.34
  Node (2185b1bc) -> Nexthop (29124240) ref 3, If 17(eth 1/18) Addr 10.1.2.2
  Node (2185b1d4) -> Nexthop (29124250) ref 2, If 152(eth 4/9) Addr 10.0.0.25
  Node (2185b1ec) -> Nexthop (29124260) ref 2, If 155(eth 4/12) Addr 10.0.0.46
  Node (2185b3e4) -> Nexthop (29124410) ref 1, If 144(eth 4/1) Addr 10.0.0.21
  Node (2185b288) -> Nexthop (29124480) ref 1, If 147(eth 4/4) Addr 10.0.0.6
  Node (2185b1b0) -> Nexthop (291243e0) ref 3, If 49(eth 2/2) Addr 10.0.0.10
```

show isis debug parent-link-info

Syntax: show isis debug parent-link-info *system-id*

This command displays all the parent link information for a specific router. The *system-id* variable specifies the router name or system ID.

Command output resembles the following example.

```
Brocade# show isis debug parent-link-info R4
Link-1
    Parent: R2, Child:  R4
    Link Metric: 10, Total Metric: 20, Link State: Active
Link-2
    Parent: R3, Child:  R4
    Link Metric: 15, Total Metric: 25, Link State: In-Active
```

show isis debug pent

Syntax: show isis debug pent

This command displays path entries. It displays all the nodes in the topology and the cost to each node from the root node, the preference, flags, and the IPv4 and IPv6 next-hop associations, as the following example illustrates.

```
Brocade# show isis debug pent
Path Table for Level 1:
Pent 2aa8e008
 Hash-Idx 34 PENT_IS  R2-0-0    cost 0 pref 1  flags 0
                No IP Nexthops associated with this Pent entry
                No IPv6 Nexthops associated with this Pent entry
Path Table for Level 2:
Pent 2aa8e2b0
 Hash-Idx 0 PENT_IS  R4-3-0     cost 61 pref 2  flags 0
        Pent IPv4 Nexthop Set 2c2e1020
                10.3.1.4 eth 3/1
                No IPv6 Nexthops associated with this Pent entry
Pent 2aa8e118
 Hash-Idx 0 PENT_IS  R4-0-0     cost 1 pref 2  flags 0
        Pent IPv4 Nexthop Set 2c2e1020
                10.3.1.4 eth 3/1
                No IPv6 Nexthops associated with this Pent entry
Pent 2aa8e448
 Hash-Idx 0 PENT_IS  R1-3-0     cost 120 pref 2  flags 0
        Pent IPv4 Nexthop Set 2c2e1030
                10.37.1.1 eth 3/7
                No IPv6 Nexthops associated with this Pent entry
```

show isis debug pent-level-info

Syntax: show isis debug pent-level-info

When IS-IS multi-topology is enabled, this command displays an integrated IS-IS IPv4 level information list associated with path entries.

Command output resembles the following example, when IS-IS multi-topology is enabled.

```
Brocade# show isis debug pent-level-info
Pent-Id   level metric pref Chg(N-D-MC-PC-IPV4NC-IPV6NC-PSPFC)
 XMR44.00-00 L1      10     1 (0-0-0-0-0-0-0)
10.0.0.0/10.255.255.0 cost:10, Pre:1, Up/Down:0, Credit: 0, flags:Act Ena   IPv4
10.0.0.0/10.255.255.0 cost:10, Pre:1, Up/Down:0, Credit: 0, flags:Act Ena   IPv4
10.4.4.0/10.255.255.0 cost:10, Pre:1, Up/Down:0, Credit: 0, flags:Act Ena   IPv4
10.0.0.0/10.255.255.0 cost:10, Pre:1, Up/Down:0, Credit: 0, flags:Act Ena   IPv4

Pent is not found 0000.0000.0000.00-00 level 2
 XMR43.00-00 L2       0     1 (0-0-0-0-0-0-0)
```

Command output resembles the following example, when IS-IS multi-topology is not enabled.

```
Brocade# show isis debug pent-level-info
 Pent-Id   level metric pref Chg(N-D-MC-PC-IPV4NC-IPV6NC-PSPFC)
 XMR2.00-00 L2      10     2 (0-0-0-0-0-0-0)
        10.0.0.0/10.255.255.252 cost:306, Pre:2, Up/Down:0 flags:Act Ena   IPv4
        10.0.0.4/10.255.255.252 cost:10, Pre:2, Up/Down:0 flags:Act Ena   IPv4
        10.0.0.36/10.255.255.252 cost:1000, Pre:2, Up/Down:0 flags:Act Ena   IPv4
        10.0.0.0/10.255.255.0 cost:10, Pre:2, Up/Down:0 flags:Act Ena   IPv4
        10.0.0.0/10.255.255.0 cost:10, Pre:2, Up/Down:0 flags:Act Ena   IPv4
        10.1.6.0/10.255.255.0 cost:10, Pre:2, Up/Down:0 flags:Act Ena   IPv4
        10.0.0.8/10.255.255.252 cost:10, Pre:2, Up/Down:0 flags:Act Ena   IPv4
        10.0.0.12/10.255.255.252 cost:10, Pre:2, Up/Down:0 flags:Act Ena   IPv4
        10.0.0.12/10.255.255.252 cost:10, Pre:2, Up/Down:0 flags:Act Ena   IPv4
        10.0.0.0/10.255.255.0 cost:10, Pre:2, Up/Down:0 flags:Act Ena   IPv4
        10.22.22.22/10.255.255.255 cost:10, Pre:2, Up/Down:0 flags:Act Ena   IPv4
```

```
 GSR.00-00 L2      20     2 (0-0-0-0-0-0-0-0)
        10.0.1.0/10.255.255.0 cost:10, Pre:2, Up/Down:0 flags:Act Ena   IPv4
        10.1.1.0/10.255.255.0 cost:10, Pre:2, Up/Down:0 flags:Act Ena   IPv4
        10.9.9.9/10.255.255.255 cost:10, Pre:2, Up/Down:0 flags:Act Ena   IPv4
        10.25.25.0/10.255.255.0 cost:10, Pre:2, Up/Down:0 flags:NoAct Ena   IPv4
```

show isis debug pspf-lsp-list

Syntax: show isis debug pspf-lsp-list *level*

- *level* - Displays information for a specific list level.

This command displays the LSPs in the partial SPF list. The argument “level” is optional. If you do not mention the argument level, the LSPs in both the Level-1 and Level-2 lists are displayed.

```
Brocade # show isis debug  pspf-lsp-list
 ISIS Level-1 PSPF LSP List
LSP-ID            State      Time-Stamp(Tics)            Last-Trans-Time(Tics)
mu2.00-00         Update     0m0s      (1202772185)       1h10m9s   (0)
mu2.00-01         New        0m0s      (1202772185)      1h10m9s   (0)

ISIS Level-2 PSPF LSP List
LSP-ID             State      Time-Stamp(Tics)            Last-Trans-Time(Tics)
mu2.00-00         Update      0m0s      (1202772185)     1h10m9s   (0)
mu2.00-01         New         0m0s      (1202772185)     1h10m9s   (0)
```

show isis debug redis

Syntax: show isis debug redis

This command shows all routes that have been added, deleted, and so on, from other protocols.

```
Brocade# show isis debug redis
ISIS Redistribution Stats:
   Add 20, Modify 0, Del 0, Clear 0, Clear-All 0, Invalid 0
   Invalid-Add 0, Invalid-Mod 0
Prefix Mask level rt_type cost met_type
10.1.10.0/24 3 0 1 1
10.1.2.0/24 3 0 1 1
10.1.5.0/24 3 0 1 1
10.1.2.0/24 3 0 1 1
10.1.5.0/24 3 0 1 1
10.1.8.0/24 3 0 1 1
10.1.8.0/24 3 0 1 1
10.1.11.0/24 3 0 1 1
10.1.3.0/24 3 0 1 1
10.1.3.0/24 3 0 1 1
10.1.6.0/24 3 0 1 1
10.1.9.0/24 3 0 1 1
10.1.6.0/24 3 0 1 1
10.1.9.0/24 3 0 1 1
10.1.1.0/24 3 0 1 1
10.1.4.0/24 3 0 1 1
10.1.7.0/24 3 0 1 1
10.1.4.0/24 3 0 1 1
10.1.7.0/24 3 0 1 1
10.1.10.0/24 3 0 1 1
```

show isis debug route-info

Syntax: **show isis debug route-info** [*A.B.C.D* |*A.B.C.D/L*]

- *A.B.C.D* | *A.B.C.D/L* - Specifies the IP network address.

This command displays the routes in the IS-IS route table. For each route, it displays the next hops, the cost from the root node, flags, and its parent path entries, as shown in the following example.

```
Brocade# show isis debug route-info 10.0.0.1/32
10.0.0.1         255.255.255.255  30        L2   00000000 00000008  000
   Path: 1       Next Hop IP: 192.168.1.125     Interface: 4/16
  Route 3741d938
  Level-1 Info List is empty
  Level-2 Info List
        PENT-V4:AR1.00-00 cost:10, Pre:2, Up/Down:0 flags:Act Ena   IPv4
```

show isis debug summary

Syntax: **show isis debug summary**

This command displays summary information about redistributed routes, such as whether a route is active (UP=yes or no), the metric (cost), the level (L column), and so on.

```
Brocade# show isis debug summary
Summary       UP     Met    L
10.1.0.0      /16 Y  1      3 (e1=11,e2=11,i1=0,i2=0) (l1=12e0fc34/1 l2=12e0fb14/1)
```

show isis debug v6-nexthops

Syntax: **show isis debug v6-nexthops**

This command displays information about IS-IS IPv6 next hops, as shown in the following example.

```
Brocade# show isis debug v6-nexthops
IS-IS IPv6 Nexthops List:
IS-IS IPv6 Null0 Nexthops List:
Node 2185b000 -> Nexthop 2809f000 ref 0, If 65534(drop) Addr ::
IS-IS IPv6 Nexthops Set List:
```

show isis debug v6route-info

Syntax: **show isis debug v6route-info**

This command displays the routes in the IS-IS IPv6 route table. For each route, it displays the next hops, the cost from the root node, flags, and its parent path entries, as shown in the following example.

```
Brocade# show isis debug v6route-info
ISIS IPv6 Routing Table
Total Routes: 1  Level1: 0 Level2: 1 Equal-cost multi-path: 1
Type IPv6 Prefix                Next Hop Router           Interface  Cost
L2   2001:DB8::/64              fe80::202:17ff:fe6e:c41c   eth 1/11   0
        PENT:Cisco.00-00 cost:10, Pre:2, Up/Down:0 flags:Act Ena   IPv6
```

show ip route isis

Syntax: **show ip route isis**

This command displays information about IS-IS routes, as shown in the following example.

```
Brocade# show ip route isis
Type Codes - B:BGP D:Connected I:ISIS S:Static R:RIP O:OSPF; Cost - Dist/Metric
        Destination        Gateway         Port         Cost       Type
1       10.30.30.0/24      10.50.50.10     gre_tnl 1    115/20     IL1
2       10.100.100.0/24    10.50.50.10     gre_tnl 1    115/20     IL1
3       10.100.101.0/24    10.50.50.10     gre_tnl 1    115/20     IL1
4       10.100.102.0/24    10.50.50.10     gre_tnl 1    115/20     IL1
5       10.100.103.0/24    10.50.50.10     gre_tnl 1    115/20     IL1
6       10.100.104.0/24    10.50.50.10     gre_tnl 1    115/20     IL1
7       10.100.105.0/24    10.50.50.10     gre_tnl 1    115/20     IL1
8       10.100.106.0/24    10.50.50.10     gre_tnl 1    115/20     IL1
9       10.100.107.0/24    10.50.50.10     gre_tnl 1    115/20     IL1
```

show isis hostname

Syntax: **show isis hostname**

This command displays IS-IS name mappings, as shown in the following example.

```
Brocade# show isis hostname
Total number of entries in IS-IS Hostname Table: 1
   System ID       Hostname            * = local IS
 * 0000.00cc.dddd   IMR
```

IS-IS debug commands

This section describes the debug commands used for monitoring the IS-IS environment.

debug isis

Syntax: [**no**] **debug isis** [**adj** | **bfd** | **error** | **interface** | **l1-csnp** | **l1-hello** | **l1-lsp** | **l1-psnp** | **l2-csnp** | **l2-hello** | **l2-lsp** | **l2-psnp** | **lsp-flood** | **lsp-dump** | **memory** | **nsr** | **pp-hello** | **ppp** | **pspf** | **pspf-detail** | **redistribution** | **route-table** | **spf** | **spf-log** | **te** | **trace**]

- **adj** - Displays information about IS-IS adjacencies.
- **bdf** - Displays IS-IS BFD information.
- **error** - Displays IS-IS errors.
- **interface** - Limits the display of IS-IS information to a specific interface.
- **l1-csnp** - Displays level 1 CSNP PDU information.
- **l1-hello** - Displays level 1 hello PDU information.
- **l1-lsp** - Displays level 1 LSP PDU information.
- **l1-psnp** - Displays level 1 PSNP PDU information.
- **l2-csnp** - Displays level 2 CSNP PDU information.
- **l2-hello** - Displays level 2 hello PDU information.
- **l2-lsp** - Displays level 2 LSP PDU information.
- **l2-psnp** - Displays level 2 PSNP PDU information.
- **lsp-flood** - Displays information about LSP flooding.
- **lsp-dump** - Displays a dump of context-specific LSP contents.
- **memory** - Displays memory information.

- **nsr** - Displays IS-IS nonstop routing information.
- **pp-hello** - Displays PP hello PDU information.
- **ppp** - Displays OSI Point-to-Point Protocol (PPP) information.
- **pspf** - Displays IS-IS partial SPF information.
- **pspf-detail** - Displays IS-IS partial SPF information details.
- **redistribution** - Displays IS-IS route redistribution information.
- **route-table** - Displays IS-IS route table information.
- **spf** - Displays IS-IS SPF information.
- **spf-log** - Displays information about the SPF log.
- **te** - Displays information about IS-IS traffic engineering.
- **trace** - Displays trace information for the IS-IS code path.

debug isis adj

Syntax: [**no**] **debug isis adj**

This command generates information about IS-IS adjacencies. Command output resembles the following example.

```
Brocade# debug isis adj
ISIS: Clearing all adjacencies on 1/1
ISIS: Deleting PTPT Adj to rtr1 on 1/1 from HT Timer for HT 30 Index 1]
ISIS: L1 DIS change on 1/4 to rtr2-3
ISIS: L2 DIS change on 1/4 to rtr2-3
ISIS: Deleting PTPT Adj to rtr1 on 1/1 [HoldTimer expiry]
ISIS: Deleting PTPT Adj to rtr1 on 1/1 from HT Timer for HT 30 [Index 0]
ISIS: Deleting PTPT Adj to rtr1 on 1/2 [HoldTimer expiry]
ISIS: Deleting PTPT Adj to rtr1 on 1/2 from HT Timer for HT 30 [Index 1]
ISIS: Adding PTPT Adj 0000.0000.0001 on 1/1 to HT Timer for HT 30 [Index 0]
ISIS: Adding PTPT Adj 0000.0000.0000 on 1/2 to HT Timer for HT 7680 [Index 1]
ISIS: L1 DIS change on 1/4 to rtr2-3
ISIS: L2 DIS change on 1/4 to rtr2-3
ISIS: L1 DIS change on 1/4 to rtr2-3
ISIS: Adding PTPT Adj rtr1 on 1/1 to HT Timer for HT 30 [Index 0]
ISIS: Adding PTPT Adj rtr1 on 1/2 to HT Timer for HT 30 [Index 1]
```

debug isis l1-csnp

Syntax: [**no**] **debug isis l1-csnp**

This command displays information about level 1 complete sequence number PDUs (CSNPs) sent and received on the device. Command output resembles the following example.

```
Brocade# debug isis l1-csnp
ISIS: Sending L1 CSNP on 2/24, length 1497
ISIS: Received L1 CSNP on 2/24, length 256 from 0000.0026.b337
```

debug isis l1-hello

Syntax: [**no**] **debug isis l1-hello**

This command generates information about level 1 hello PDUs sent and received. Command output resembles the following example.

```
Brocade# debug isis l1-hello
ISIS: Received L1 LAN IIH on 2/24, length 256 from 0000.0026.b337
ISIS: Sending L1 LAN IIH on 2/24, length 1497
ISIS: Received L1 LAN IIH on 2/24, length 256 from 0000.0026.b337
```

debug isis l1-lsp

Syntax: [no] debug isis l1-lsp

This command generates information about level 1 link state PDUs (LSPs) sent and received. Command output resembles the following example.

```
Brocade# debug isis l1-lsp
ISIS: Sending L1 LSP on 2/24, length 27
ISIS: Received L1 LSP on 2/24, length 256 from 0000.0026.b337
```

debug isis l1-psnp

Syntax: [no] debug isis l1-psnp

This command generates information about level 1 partial sequence number PDUs (PSNPs) sent and received. Command output resembles the following example.

```
Brocade# debug isis l1-psnp
ISIS: Received L1 PSNP on 2/24, length 256
ISIS: Received L1 PSNP on 2/24, length 35
```

debug isis l2-csnp

Syntax: [no] debug isis l2-csnp

This command generates information about level 2 CSN PDUs sent and received. Command output resembles the following example (source MAC addresses are shown).

```
Brocade# debug isis l2-csnp
ISIS: Rcvd L2 CSNP on 2/1, length 906 from fr1.iad.QA.Tes [MAC 0000.00e3.0c02]
ISIS: Rcvd L2 CSNP on 1/1, length 906
ISIS: Rcvd L2 CSNP on 1/2, length 906 from fr1.sjc.QA.Tes [MAC 0000.00e3.b629]
ISIS: Rcvd L2 CSNP on v510, length 906
ISIS: Rcvd L2 CSNP on v510, length 906 from fr1.sjc.QA.Tes [MAC 0000.00e3.b600]
ISIS: Rcvd L2 CSNP on 2/1, length 906
ISIS: Rcvd L2 CSNP on 2/1, length 906 from fr1.iad.QA.Tes [MAC 0000.00e3.0c02]
ISIS: Rcvd L2 CSNP on 1/1, length 906
```

debug isis l2-hello

Syntax: [no] debug isis l2-hello

This command generates information about level 2 hello PDUs sent and received. Command output resembles the following example.

```
Brocade# debug isis l2-hello
ISIS: Received L2 LAN IIH on 2/24, length 256 from 0000.0026.b337
ISIS: Sending L2 LAN IIH on 2/24, length 1497
ISIS: Received L2 LAN IIH on 2/24, length 256 from 0000.0026.b337
```

debug isis l2-lsp

Syntax: [no] debug isis l2-lsp

This command generates information about level 2 link state PDUs sent and received. Command output resembles the following example.

```
Brocade# debug isis l2-lsp
ISIS: Sending L2 LSP on 2/24, length 27
ISIS: Received L2 LSP on 2/24, length 256 from 0000.0026.b337
```

debug isis l2-psnp

Syntax: [**no**] **debug isis l2-psnp**

This command generates information about level 2 PSN PDUs (PSNPs) sent and received. Command output resembles the following example.

```
Brocade# debug isis l2-psnp
ISIS: Received L2 PSNP on 2/24, length 256
ISIS: Received L2 PSNP on 2/24, length 35
```

debug isis memory

Syntax: [**no**] **debug isis memory**

This command generates information about IS-IS memory allocations and releases. Command output resembles the following example.

```
Brocade# debug isis memory
ISIS: Memory Allocated for buffer description at 21a54ad8
ISIS: Memory Allocated for packet-buffer at 211e1680
ISIS: Memory Released for buffer descriptor at 21a54ad
ISIS: Memory Allocation for circuit IP address failed
```

debug isis nsr

Syntax: [**no**] **debug isis nsr**

This command displays information related to LSP, neighbor syncing, and NSR state-related information. Command output resembles the following example.

```
Brocade# debug isis nsr
ISIS: Sending L1-LSP XMR36.01-00 Seq-No 6173 Flags Lsp Update Length 53 to standby
ISIS: Ack recv for L1 Lsp XMR36.01-00 Addition from Standby
ISIS: Sending L1 Nbr CES Flags Neighbor Delete to standby
ISIS: Ack recv for L2 Neighbor CES Deletion from Standby
ISIS: Sending L2 Nbr CES Flags Neighbor Update to standby
ISIS: Ack recv for L2 Neighbor CES Addition from Standby
```

debug isis pp-hello

Syntax: [**no**] **debug isis pp-hello**

This command displays information about point-to-point hello PDUs sent and received. Command output resembles the following example.

```
Brocade# debug isis pp-hello
ISIS: Sending PTP IIH on 9/1, length 1492
ISIS: Received PTP IIH on 9/1, length 256
```

debug isis ppp

Syntax: [**no**] **debug isis ppp**

This command generates information about OSI PPP packets sent and received. Command output resembles the following example.

```
Brocade# debug isis ppp
ISIS PPP: sending isis packet on pos port 512
ISIS: osicp datainput rx pkt length 1492 on unit 32
ISIS: Received PTP IIH on 9/1, length 256
ISIS: Sending PTP IIH on 9/1, length 1492
ISIS PPP: sending isis packet on pos port 512
```

debug isis pspf

Syntax: [no] debug isis pspf

This command generates information about IS-IS PSPF activity. Command output resembles the following example.

```
Brocade# debug isis pspf
ISIS: Comparing old options against new to detect routes that may have been
removed
ISIS: Checking ISOC_EIPREACH,
ISIS: Checking ISOC_EIPREACH,
ISIS: Comparing new options against old to detect routes that may have been added
ISIS: isis_check_if_partial_spf_needed called
ISIS: isis_identify_and_process_changed_ip_information_in_lsp called
ISIS: Checking ISOC_EREACH
ISIS: Checking ISOC_IREACH
```

debug isis pspf-detail

Syntax: [no] debug isis pspf-detail

This command generates detailed information about IS-IS PSPF activity. Command output resembles the following example.

```
Brocade# debug isis pspf-detail
ISIS: Total Route Calculation Time is 0 milliseconds.
ISIS: PSPF Started for level 2
PENT_IP found id = 10.0.6.0 10.255.255.0 cost 41 pref 2 up=0
PENT_IP found id = 10.0.170.0 10.255.255.0 cost 20 pref 2 up=0
PENT_IP found id = 10.0.12.0 10.255.255.0 cost 50 pref 2 up=0
PENT_IP found id = 10.0.18.0 10.255.255.0 cost 50 pref 2 up=0
PENT_IP found id = 10.0.4.0 10.255.255.0 cost 41 pref 2 up=0
PENT_IP found id = 10.0.10.0 10.255.255.0 cost 50 pref 2 up=0
PENT_IP found id = 10.0.16.0 10.255.255.0 cost 50 pref 2 up=0
PENT_IP found id = 10.131.243.16 10.255.255.252 cost 40 pref 2 up=0
```

debug isis redistribution

Syntax: [no] debug isis redistribution

This command displays route redistribution information into or out of IS-IS routes. Command output resembles the following example.

```
Brocade# debug isis redistribution
ISIS: Imported CONNECTED route 10.10.0.3 10.255.0.0
ISIS: Imported CONNECTED route 10.10.0.3 10.255.0.0
ISIS: Added external route 10.10.0.3 10.255.0.0 to L12 LSP
ISIS: Added external route 10.10.0.0 10.255.0.0 to L12 LSP
```

```
ISIS: Unimported CONNECTED route 10.10.0.0 10.255 0.0
ISIS: Unimported CONNECTED route 10.10.0.0. 10.255.0.0
ISIS: Deleted external route 10.10.0.0. 10.255.0.0 from L12 LSP
ISIS: Deleted external route 10.10.0.0 10.255.0.0. from L12 LSP
```

This output indicates several redistribution activities, including importing and unimporting connected routes, and adding and deleting external routes.

debug isis route-table

Syntax: [no] debug isis route-table

This command reports changes to the IS-IS route table. Command output resembles the following example.

```
Brocade# debug isis route-table
ISIS: Deleting route 10.10.0.0. 10.255.0.0 level 2
ISIS: Deleting route 10.10.0.0 10.255.0.0 level 2
ISIS: Creating new route for 10.10.0.0 10.255.0.0. level 2 type 1
ISIS: Adding path Next hop = 10.147.201.200 Interface 2/4
ISIS: Creating new route for 10.10.0.0 10.255.0.0 level 2 type 1
```

debug isis spf

Syntax: [no] debug isis spf

This command generates information about SPF calculations made for IS-IS. Command output resembles the following example.

```
Brocade# debug isis spf
ISIS6: MT IPv6 SPF start...
ISIS: Building SPF tree for Level:1, ISPF:No, SPF_Index: Native Table,
isis.ipv4_rtm_update_index[level1]: Native Table,
isis.ipv6_rtm_update_index[level1]: Native Table
ISIS: Tent-id R3.00-00 type 2 OptType 0 Added with cost 0 Pref 1 Updown bit 0
ISIS: Adding Pent-id R3.00-00 type 2 OptType 0 Added with cost 0 Pref 1 Updown bit
0 to Path List
ISIS: The Pent Already exists in Path
ISIS: The Pent Entry Flags for Native Table 0
ISIS: Updating the Active Info for 2001:DB8::/64 Previous active Level 2, safi
IPv6
ISIS: L1 Level Info List is empty
ISIS: Updating the Active Info for 2001:DB8::/64 Previous active Level 2, safi
IPv6
ISIS: L1 Level Info List is empty
ISIS: Updating the Active Info for 2001:DB8::/64 Previous active Level 2, safi
IPv6
ISIS: L1 Level Info List is empty
ISIS: Building SPF tree for Level:2, ISPF:No, SPF_Index: Native Table,
isis.ipv4_rtm_update_index[level2]: Native Table,
isis.ipv6_rtm_update_index[level2]: Native Table
ISIS: Tent-id R3.00-00 type 2 OptType 0 Added with cost 0 Pref 1 Updown bit 0
ISIS: Adding Pent-id R3.00-00 type 2 OptType 0 Added with cost 0 Pref 1 Updown bit
0 to Path List
ISIS: The Pent Already exists in Path
ISIS: The Pent Entry Flags for Native Table 0
ISIS: Tent-id R2.00-00 type 2 OptType 0 Added with cost 14 Pref 2 Updown bit 0
ISIS: Adding Pent-id R2.00-00 type 2 OptType 0 Added with cost 10 Pref 2 Updown
bit 0to Path List
ISIS: The Pent Already exists in Path
ISIS: The Pent Entry Flags for Native Table 0
```

```
ISIS: The Pent IPv6 Nexthop is not Changed in this SPF run
ISIS: Trying to Add Level Info 2001:DB8::/64 cost 10  Pref 2 Opttype 236 UpDown 0
ISIS: level info element exists for this Pent Entry with cost 10 Pref 2 Updown Bit
0
ISIS: Level info flags after update 33
ISIS:Trying to delete Disabled Route Level Info for 2001:DB8::/64 Level-2 safi
IPv6 Table_Index Native
ISIS: Updating the Active Info for 2001:DB8::/64 Previous active Level 2, safi
IPv6
ISIS: L1 Level Info List is empty
ISIS6: MT IPv6 SPF end!
```

debug isis spf-stct

Syntax: [no] debug isis spf-stct

This command displays debug information related to incremental shortcut LSP SPF optimizations, as shown in the following example.

```
Brocade# debug isis spf-stct
ISIS: Fixing SPF type for Tunnel-0 with Status: UP
ISIS: ISIS: Triggering ISTCT SPF
ISIS: ISTCT_SPF triggered for LSP Tunnel-0 Up Ir_istct_spf_level1:0
Ir_istct_spf_level2 1
ISIS: Level-2 isis.ipv4_rtm_update_index: STCT, isis.prev_ipv4_rtm_update_index:
STCT
ISIS: ISTCT_SPF Scheduled, Ir_istct_spf_level1:0 Ir_istct_spf_level2:1
ISIS: ISTCT_SPF Scheduled for level-2
ISIS: Processing Changed LSP Shortcut List
ISIS: Processing tunnel0 Up State
ISIS: Adding shortcut link to XMR17.00-00 with Tnnl_index: 0
ISIS: Newly added Stct link from XMR16.00-00 to XMR17.00-00 is Active
ISIS: Pent Adj count: 1, MAX_MSPLIT: 8, iso_tmo->rt_msplit: 4
ISIS: Adding Pent XMR17.00-00 to begining of Change Adj List
ISIS: Processing of Changed LSP Shortcut List took 0 msec
ISIS: Processing Pent Changed Adj List
ISIS: Updating Pent XMR17.00-00 Nexthop Set
ISIS: Adding Pent XMR17.00-00 to Level-2 nd_pspf_pe_list
ISIS: Adding Pent XMR17.00-00 Active Child to Pent Change Adj List
ISIS: Processing of Pent Change Adj List took 0 msec
```

debug isis trace

Syntax: [no] debug isis trace

This command generates information about internal IS-IS functions. Command output resembles the following example.

```
Brocade# debug isis trace
ISIS:proc_SNPE
ISIS: build_csnp
ISIS: build_csnp
ISIS: sig_description
```

Configuration notes

None of the IS-IS parameters require a software reload for changes to take effect, and most parameter changes take effect immediately. However, changes for the following parameters take effect only after you disable and then re-enable redistribution:

- Changing the default metric.
- Adding, changing, or negating route redistribution parameters.

Common diagnostic scenarios

- Packets are dropped during authentication mode change.

  Changing the authentication mode can cause packets to drop during the transition period because not all of the routers are reconfigured simultaneously. During such a transition, it can be useful to disable IS-IS authentication checking temporarily until all routers are reconfigured and the network is stable.

  Use the **no isis auth-check** command to disable IS-IS authentication checking on a specified interface.

```
Brocade(config)# interface ethernet 3/1
Brocade(if-e10000-3/1)# no isis auth-check level-1
```

  Syntax: [**no**] **isis auth-check** [**level-1** | **level-2**]

  This command enables and disables IS-IS authentication checking. The default is enabled and the **no** parameter disables authentication checking.

  The **level-1** parameter specifies that authentication checking is enabled or disabled for Level 1 Hello packets.

  The **level-2** parameter specifies that authentication checking is enabled or disabled for Level 2 Hello packets.

  NOTE
  If either **Level 1** or **Level 2** are not specified, the configuration is applied to both **Level 1** and **Level 2**.

- NetIron LSPs are timed-out by its neighbors.

  The maximum LSP lifetime interval and the LSP refresh interval must be set so that the LSPs are refreshed before the maximum LSP lifetime interval expires; otherwise, the Brocade device's originated LSPs may be timed-out by its neighbors.

  To prevent a neighbor from timing-out LSPs, give the LSP refresh interval a shorter setting than that of the maximum LSP lifetime interval. The LSP refresh interval is the maximum number of seconds the Brocade device waits between sending updated LSPs to its IS-IS neighbors. The interval can be from 1 through 65535 seconds. The default is 900 seconds.

  For example, to change the LSP refresh interval to 20000 seconds, enter a command similar to the following example.

```
Brocade(config-isis-router)#lsp-refresh-interval 20000
```

  Syntax: [**no**] **lsp-refresh-interval** *secs*

  The *secs* variable specifies the maximum refresh interval and can be from 1 through 65535 seconds. The default is 900 seconds (15 minutes).

- An IS-IS link is experiencing low performance rates.

If an IS-IS link is performing poorly, it may be due to padding that is added to the end of a hello packet to make the packet the same size as the maximum length of a PDU supported by the Brocade device.The Brocade device adds this padding by default to the following types of hello packets:

- ES hello (ESH PDU)
- IS hello (ISH PDU)
- IS-IS hello (IIH PDU)

When padding is enabled, the maximum length of a hello PDU sent by the Brocade device is 1514 bytes.

If you suspect that padding is affecting the performance of the link, you can disable padding globally or on individual interfaces. If you enable or disable padding on an interface, the interface setting overrides the global setting.

To globally disable padding of IS-IS hello PDUs, enter the following command.

```
Brocade(config-isis-router)# no hello padding
```

This command disables all hello PDU padding on the Brocade device. To re-enable padding, enter the following command.

```
Brocade(config-isis-router)# hello padding
```

Syntax: [**no**] **hello padding**

By default, hello padding is enabled. Enter the **no** form of the command to disable hello padding.

- IS-IS adjacent ports are flapping.

  Check for any high CPU condition or evidence of packet loss. Do a ping test to see if packet loss occurs.

- Summary addresses are used to enhance performance.

  Summary addresses can enhance performance by reducing the size of the Link State database, reducing the amount of data the Brocade device needs to send to its neighbors, and reducing the CPU cycles used for IS-IS.

  When you configure a summary address, the address applies only to Level-2 routes by default. You can specify Level-1 only, Level-2 only, or Level-1 and Level-2 when you configure the address.

  To configure a summary address, enter a command similar to the following example.

```
Brocade(config-isis-router-ipv4u)# summary-address 192.168.0.0 10.255.0.0
```

This command configures a summary address for all Level-2 IS-IS route destinations between 192.168.1.0 and 192.168.255.255.

Syntax: [**no**] **summary-address** *ip-addr subnet-mask* [**level-1** | **level-1-2** | **level-2**]

The *ip-addr subnet-mask* variables specify the aggregate address. The mask indicates the significant bits in the address. Ones are significant, and zeros allow any value. In the example, the mask 10.255.0.0 matches on all addresses that begin with 192.168 and contain any values for the final two octets.

The **level-1** | **level-1-2** | **level-2** parameter specifies the route types to which the aggregate route applies. The default is **level-2**.

VRRP and VRRP-E

This section describes how to troubleshoot the IP VRRP and VRRP-E environments for the Brocade NetIron XMR and Brocade MLX series routers.

Virtual Router Redundancy Protocol (VRRP) is an election protocol that provides alternate router paths for a host without changing the IP address or MAC address by which the host knows its gateway.

Virtual Router Redundancy Protocol Extended (VRRP-E) is a proprietary version of VRRP that overcomes limitations in the standard protocol.

VRRP show commands

This section describes the show commands that display VRRP information.

show ip vrrp

Syntax: **show ip vrrp** [**brief** | **ethernet** *slotnum/portnum* | **ve** *num* | **vrid** *num* | **statistics**]

- **brief** - Displays IPv4 VRRP summary information. If you do not use this parameter, detailed information is displayed instead.
- **ethernet** *slotnum/portnum* - Specifies an Ethernet port. If you use this parameter, the command displays IPv4 VRRP information only for the specified port.
- **ve** *num* - Specifies a virtual interface. If you use this parameter, the command displays IPv4 VRRP information only for the specified virtual interface.
- **vrid** *num*- Displays IPv4 VRRP information only for the specified virtual router ID.
- **statistics** - Displays statistics.

show ip vrrp brief

Syntax: **show ip vrrp brief**

This command displays IPv4 VRRP summary information, as shown in the following example.

```
Brocade# show ip vrrp brief
Total number of VRRP-Extended routers defined: 4
Inte- VRID Current  P  State  Master IP       Backup IP       Virtual IP
rface      Priority           Address         Address         Address
-----------------------------------------------------------------------------
v10   1    100         Init   Unknown         Unknown         192.168.1.1
v20   1    100         Init   Unknown         Unknown         10.10.20.1
v30   1    100         Init   Unknown         Unknown         10.10.30.1
v100  1    100         Init   Unknown         Unknown         10.10.100.1
```

show ip vrrp-extended

Syntax: **show ip vrrp-extended** [**brief** | **ethernet** *slotnum/portnum* | **ve** *num* | **vrid** *num*| **statistics**]

- **brief** - Displays IPv4 VRRP-E summary information.
- **ethernet** *slotnum/portnum* - Specifies an Ethernet port. If you use this parameter, the command displays IPv4 VRRP-E information only for the specified port.

- **ve** *num* - Specifies a virtual interface. If you use this parameter, the command displays IPv4 VRRP-E information only for the specified virtual interface.
- **vrid** *num* - Displays IPv4 VRRP-E information only for the specified virtual router ID.
- **statistics** - Displays statistics.

The following example shows the output from the **show ip vrrp-extended ethernet** *slotnum/portnum* command.

```
Brocade# show ip vrrp-extended ethernet 1/1
Interface ethernet 1/1
auth-type md5-authentication key $$$$$
VRID 1
state master
administrative-status enabled
version v2
mode owner
priority 255
current priority 255
hello-interval 1000 msec
ip-address 10.0.0.3
virtual mac address 0000.0000.0101
advertise backup: disabled
next hello sent in 00:00:00.1
```

The following example shows output from the **show ip vrrp-extended statistics** command.

```
Brocade# show ip vrrp-extended statistics
Interface ethernet 1/1
rxed vrrp header error count = 0
rxed vrrp auth error count = 0
rxed vrrp auth passwd mismatch error count = 0
rxed vrrp vrid not found error count = 0
VRID 1
rxed arp packet drop count = 0
rxed ip packet drop count = 0
rxed vrrp port mismatch count = 0
rxed vrrp ip address mismatch count = 0
rxed vrrp hello interval mismatch count = 0
rxed vrrp priority zero from master count = 0
rxed vrrp higher priority count = 0
transitioned to master state count = 1
transitioned to backup state count = 1
```

show ipv6 vrrp

Syntax: show ipv6 vrrp [**brief** | **ethernet** *slotnum/portnum*| **statistics** | **ve** *num* | **vrid** *num*]

- **brief** - Displays IPv6 VRRP summary information.
- **ethernet** *slotnum/portnum* - Limits the display of IPv6 VRRP information only to the specific Ethernet interface.
- **statistics** - Displays virtual router statistics.
- **ve** *num* - Displays IPv6 VRRP information only for the specified virtual interface.
- **vrid** *num* - Displays IPv6 VRRP information only for the specified virtual router ID.

show ipv6 vrrp brief

Syntax: show ipv6 vrrp brief

This command displays IPv6 VRRP summary information.

```
Brocade# show ip vrrp brief
Total number of VRRP routers defined: 1
Flags Codes - P:Preempt 2:V2 3:V3
Intf   VRID   CurrPrio Flags    State   Master-IPv6- Backup-IPv6- Virtual-IPv6-
                                        Address      Address       Address
-------------------------------------------------------------------------------
1/3    13     100        P3     Master     Local      2001:DB8::2   fe80::3013:2
```

show ipv6 vrrp vrid

Syntax: show ipv6 vrrp vrid *num*

This command displays IPv6 VRRP information only for the specified virtual router ID.

```
Brocade# show ipv6 vrrp vrid 13
Interface 1/3
----------------------------
auth-type no authentication VRID 13 (index 1)
interface 1/3
state master
administrative-status enabled
version v3
mode non-owner(backup)
virtual mac 0000.0000.020d
priority 100
current priority 100
track-priority 1
hello-interval 200 ms
backup hello-interval 60000 ms
advertise backup disabled
dead-interval 700 ms
preempt-mode true
ipv6-address fe80::3011:9
ipv6-address 2001:DB8::9
next hello sent in 200 ms
```

show ipv6 vrrp statistics

Syntax: show ipv6 vrrp statistics

This command displays router statistics information.

```
Brocade# show ipv6 vrrp statistics
Global IPv6 VRRP statistics
-------------------------------
- received vrrp packets with checksum errors = 0
- received vrrp packets with invalid version number = 0
- received vrrp packets with unknown or inactive vrid = 0
Interface 1/3
----------------
VRID 13
- number of transitions to backup state = 1
- number of transitions to master state = 1
- total number of vrrp packets received = 0
. received backup advertisements = 19
. received packets with zero priority = 0
received packets with invalid type = 0
. received packets with invalid authentication type = 0
```

```
. received packets with authentication type mismatch = 0
. received packets with authentication failures = 0
. received packets dropped by owner = 0
. received packets with ttl errors = 0
. received packets with ipv6 address mismatch = 0
. received packets with advertisement interval mismatch = 0
. received packets with invalid length = 0
- total number of vrrp packets sent = 1175
. sent backup advertisements = 0
. sent packets with zero priority = 0
- received neighbor solicitation packets dropped = 0
- received proxy neighbor solicitation packets dropped = 0
- received ipv6 packets dropped = 0
```

show ipv6 vrrp-extended

Syntax: **show ipv6 vrrp-extended** [**brief** | **ethernet** *slotnum/portnum* | **statistics** | **ve** *num* | **vrid** *num*]

- **brief** - Displays IPv6 VRRP-E summary information.
- **ethernet** *slotnum/portnum* - Limits the display of IPv6 VRRP-E information only to the specific Ethernet interface.
- **statistics** - Displays virtual router statistics.
- **ve** *num* - Displays IPv6 VRRP-E information only for the specified virtual interface.
- **vrid** *num* - Displays IPv6 VRRP-E information only for the specified virtual router ID.

show ipv6 vrrp-extended brief

Syntax: **show ipv6 vrrp-extended brief**

This command displays IPv6 VRRP-E summary information.

```
Brocade# show ipv6 vrrp-extended brief
Total number of VRRP routers defined: 1
Flags Codes - P:Preempt 2:V2 3:V3
Intf   VRID   CurrPrio   Flags  State  Master-IPv6   Backup-IPv6   Virtual-IPv6
                                       -Address      -Address      -Address
---------------------------------------------------------
1/3    13     100        P3     Master  Local        2001:DB8::2   2001:DB8::99
```

show ipv6 vrrp-extended vrid

Syntax: **show ipv6 vrrp-extended vrid** *num*

This command displays IPv6 VRRP-E information only for the specified virtual router ID.

```
Brocade# show ipv6 vrrp-extended vrid 13
Interface 1/3
----------------------------
auth-type no authentication VRID 13 (index 1)
interface 1/3
state master
administrative-status enabled
mode non-owner(backup)
virtual mac 0000.001d.000d
priority 100
current priority 100
track-priority 5
```

```
hello-interval 1 sec
backup hello-interval 60 sec
advertise backup enabled
dead-interval 3.1 sec
preempt-mode true
virtual ipv6 address 2001:DB8::99
next hello sent in 0.1 sec
backup router 2001:DB8::2 expires in 175.0 sec
```

show ipv6 vrrp-extended statistics

Syntax: **show ipv6 vrrp-extended statistics**

This command displays virtual router statistics information.

```
Brocade# show ipv6 vrrp-extended statistics
Global IPv6 VRRP-Extended statistics
-------------------------------
- received vrrp-extended packets with checksum errors = 0
- received vrrp-extended packets with invalid version number = 0
- received vrrp-extended packets with unknown or inactive vrid = 0
Interface 1/3
----------------
VRID 13
- number of transitions to backup state = 1
- number of transitions to master state = 1
- total number of vrrp-extended packets received = 0
. received backup advertisements = 19
. received packets with zero priority = 0
. received packets with invalid type = 0
. received packets with invalid authentication type = 0
. received packets with authentication type mismatch = 0
. received packets with authentication failures = 0
. received packets dropped by owner = 0
. received packets with ttl errors = 0
. received packets with ipv6 address mismatch = 0
. received packets with advertisement interval mismatch = 0
. received packets with invalid length = 0
- total number of vrrp-extended packets sent = 1175
. sent backup advertisements = 0
. sent packets with zero priority = 0
- received neighbor solicitation packets dropped = 0
- received proxy neighbor solicitation packets dropped = 0
- received ipv6 packets dropped = 0
```

Clearing VRRP statistics

clear ip vrrp statistics

Syntax: **clear ip vrrp statistics**

This command clears IPv4 VRRP statistics.

clear ipv6 vrrp statistics

Syntax: **clear ipv6 vrrp statistics**

This command clears IPv6 VRRP statistics.

Clearing VRRP-E statistics

clear ip vrrp-extended statistics

Syntax: clear ip vrrp-extended statistics

This command clears IPv4 VRRP-E statistics.

clear ipv6 vrrp-extended statistics

Syntax: clear ipv6 vrrp-extended statistics

This command clears IPv6 VRRP-E statistics.

VRRP debug commands

This section describes the debug commands used for monitoring the VRRP environment.

debug ip vrrp

Syntax: [**no**] **debug ip vrrp** [**all** | **error** | **ethernet** *slotnum/portnum* | **events** | **packets** | **show** | **state** | **ve** *num* | **verbose** | **vrid** *num*]

- **all** - Displays information about all IPv4 VRRP instances (default).
- **error** - Displays error conditions where a packet is not being processed.
- **ethernet** *slotnum/portnum* - Displays information about IPv4 VRRP instances on a specific physical interface.
- **events** - Displays information about activate and shutdown, port up and port down, timer events, backup VRRP router events, and so on.
- **packets** - Displays information about IPv4 VRRP transmitted and received packets, including ARP packets.
- **show** - Shows the current IPv4 VRRP debug settings.
- **state** - Displays information about IPv4 VRRP state changes, such as monitor transitions from master to backup, or vice versa.
- **ve** *num* - Displays information about a specific virtual interface.
- **verbose** - Decodes hex output into more easily understood fields and values.
- **vrid** *num* - Displays information about a specific virtual router ID.

This command displays information about IPv4 VRRP instances. The default is all instances. Several parameters are available to help isolate a specific IPv4 VRRP instance.

VRRP-E controls protocol authentication using the message digest 5 (MD5) authentication algorithm.

If debugging is enabled and MD5 authentication is configured on an interface, the **debug ip vrrp** command output resembles the following example.

```
Brocade# debug ip vrrp
Aug 10 18:17:39 VRRP6: Configuration VRRP_CONFIG_MD5_AUTHENTICATION request
received
Aug 10 18:17:39 VRRP6: Port 2/6, VRID 2 - send advertisement
 Ver:3 Type:1 Vrid:2 Pri:240 #IP:1 AuthType:2 Adv:1 Chksum:0x0000
         HMAC-MD5 CODE:[0000000000000000400010]
```

```
 IpAddr: 2001:DB80::40:10
```

If debugging is enabled and MD5 authentication is valid with its VRRP-E peer, the **debug ip vrrp** command output resembles the following example.

```
Brocade# debug ip vrrp
Aug 10 18:48:51 VRRP6: Port 2/6, VRID 2 - rcvd advertisement from 2001:DB8::40:1
 Ver:3 Type:1 Vrid:2 Pri:255 #IP:1 AuthType:2 Adv:1 Chksum:0x0000
         HMAC-MD5 CODE:[00000000000000000400010]
 IpAddr: 2001:DB8::40:10
```

If debugging is enabled and MD5 authentication fails, the **debug ip vrrp** command output resembles the following example.

```
Brocade# debug ip vrrp
Aug 10 18:32:32 VRRP6: Dropping pkt on port 2/6, vrid 2 - md5 authentication
failed
debAug 10 18:32:33 VRRP6: Port 2/6, VRID 2 - send advertisement
 Ver:3 Type:1 Vrid:2 Pri:240 #IP:1 AuthType:2 Adv:1 Chksum:0x0000
         HMAC-MD5 CODE:[00000000000000000400010]
 IpAddr: 2001:DB8::40:10
```

debug ipv6 vrrp

Syntax: [**no**] **debug ipv6 vrrp** [**all** | **error** | **ethernet** *slotnum/portnum* | **events** | **packets** | **show** | **state** | **ve** *num*| **verbose** | **vrid** *num*]

- **all** - Displays information about all IPv6 VRRP instances.
- **error** - Displays error conditions where a packet is not being processed.
- **ethernet** *slotnum/portnum* - Displays information about IPv6 VRRP instances on a specific physical interface.
- **events** - Displays information about activate and shutdown, port up and port down, timer events, backup VRRP router events, and so on.
- **packets** - Displays information about IPv6 VRRP transmitted and received packets, including ARP packets.
- **show** - Displays the current IPv6 VRRP debug settings.
- **state** - Displays information about IPv6 VRRP state changes, such as monitor transitions from master to backup, or vice versa.
- **ve** *num* - Displays information about a specific virtual interface.
- **verbose** - Decodes hex output into more easily understood fields and values.
- **vrid** *num* - Displays information about a specific virtual router ID.

This command displays information about IPv6 VRRP instances. The default is all instances. Several parameters are available to help isolate a specific IPv6 VRRP instance.

The command output resembles the following example.

```
Brocade# debug ipv6 vrrp
IPV6 VRRP: debugging is on
VRRP6: Port 1/3, VRID 23 - send advertisement
Ver:3 Type:1 Vrid:23 Pri:100 #IP:2 AuthType:0 Adv:100 Chksum:0xd37c
IpAddr: fe80::3013:2 2001:DB8::2
VRRP6: Port 1/3, VRID 23 - send advertisement
Ver:3 Type:1 Vrid:23 Pri:100 #IP:2 AuthType:0 Adv:100 Chksum:0xd37c
IpAddr: fe80::3013:2 2001:DB8::2
```

Configuration notes

The following rules apply to VRRP-E configurations:

- The interfaces of all routers in a virtual router must be in the same IP subnet.
- The IP address assigned to the virtual router cannot be configured on any of the Brocade devices.
- The address you enter with the **ip-address** command cannot be the same as a real IP address configured on the interface.
- The hello interval must be set to the same value on all the Brocade devices.
- The dead interval must be set to the same value on all the Brocade devices.
- The track priority for a virtual router must be lower than the VRRP-E priority.
- When you configure the Master (Owner), the address you enter with the **ip-address** command must already be configured on the interface.
- When you configure a Backup router, the router interface on which you are configuring the virtual router must have a real IP address that is in the same subnet (but is not the same address) as the address associated with the virtual router by the Owner.
- If you disable VRRP-E, the router removes all information for the disabled protocol from the running configuration. When you save to the startup configuration after disabling the protocol, all information for the disabled protocol is removed from the startup configuration.

CAUTION

You must configure a VRF on an interface before configuring a Virtual Router (VRRP-E) on it. If you enable the Virtual Router before you enable the VRF, the Virtual Router configuration will be deleted.

DHCPv6

Dynamic Host Configuration Protocol for IPv6 (DHCPv6) enables DHCP servers to pass configuration parameters such as IPv6 network addresses to IPv6 nodes, and controls automatic allocation of reusable network addresses.

DHCPv6 allows a DHCPv6 server to dynamically delegate IPv6 prefixes to a DHCPv6 client using the DHCPv6 Prefix Delegation (PD) option. Using the DHCPv6 relay agent PD notification feature, service providers automatically delegate IPv6 prefixes to a customer-premises equipment (CPE) which acts as a DHCPv6 client. The CPE then assigns IPv6 subnets from the delegated IPv6 prefix to its downstream customer interfaces.

DHCPv6 debug commands

This section describes the DHCPv6-related debug commands.

debug ipv6 dhcp

Syntax: [no] **debug ipv6 dhcp** [**pd all** | **pd flash** | **pd pd-option** | **pd static** | **sync**]

- **pd all** - Enables all IPv6 PD debugging.
- **pd flash** - Enables PD debugging for read and write to a flash file.
- **pd pd-option** - Enables debugging for processing of PD options in DHCPv6 messages.
- **pd static** - Enables debugging of DHCPv6 interface with IPv6 static route module.
- **sync** - Enables debugging of DHCPv6 synchronization between active and standby Management Processors (MPs).

debug ipv6 dhcp pd all

Syntax: [no] debug ipv6 dhcp pd all

This command enables all IPv6 PD debugging. Command output resembles the following example.

```
Brocade# debug ipv6 dhcp pd all
DHCP6 PD:  all debugging is on
Oct 27 22:16:44.845 DHCP6 PD: writing data to file dhcp6_delegated_prefixes_data
Oct 27 22:16:46.773 DHCP6 PD: Removing route from DHCP DP table, lifetime expired,
2001:DB8:100::/48 with nexthop 2001:DB8:1::1 interface ve 1
Oct 27 22:16:46.773 DHCP6 PD: Removing route from DHCP DP table, lifetime expired,
2001:DB8:100::/48 with nexthop 2001:DB8:1::1 interface ve 1
Oct 27 22:16:48.786 DHCP6 PD: Adding route to DHCP DP table, 2001:DB8:100::/48
with nexthop 2001:DB8:1::1 interface ve 1
Oct 27 22:16:48.786 DHCP6 PD: Adding route to DHCP DP table, 2001:DB8:100::/48
with nexthop 2001:DB8:1::1 interface ve 1
```

debug ipv6 dhcp pd flash

Syntax: [no] debug ipv6 dhcp pd flash

This command enables PD debugging for read and write to a flash file. Command output resembles the following example.

```
Brocade# debug ipv6 dhcp pd flash

Nov  7 19:14:02 DHCP6 PD: writing data to file dhcp6_delegated_prefixes_data
Nov  7 19:14:02 DHCP6 PD: 2 entries saved to flash file
Nov  7 19:12:26 DHCP6,PD: Calendar Time when file last updated was : 18:47:37
Nov  7 19:12:26 DHCP6,PD: Current Calander Time: 19:12:26 GMT+00 Mon Nov 07 2011

Nov  7 19:12:26 DHCP6 PD: reading file dhcp6_delegated_prefixes_data. Time
difference is 1489
Nov  7 19:12:26 DHCP6 PD: Parsing line

Nov  7 19:12:26 DHCP6 PD: Parsing line #VRF Name   IPv6_Prefix
Client              Interface LeaseTime Checksum

Nov  7 19:12:26 DHCP6 PD: Parsing line default-vrf 2001:DB8:100::/48
   2001:DB8:1::1           eth 1/9      3600      00001230

Nov  7 19:12:26 DHCP6 PD: Parsing successful for default-vrf 2001:DB8:100::/48
   2001:DB8:1::1           eth 1/9      3600      00001230
```

debug ipv6 dhcp pd pd-option

Syntax: [no] debug ipv6 dhcp pd pd-option

This command enables debugging for processing of PD options in DHCPv6 messages. Command output resembles the following example.

```
Brocade# debug ipv6 dhcp pd pd-option
Oct 27 22:16:46.773 DHCP6 PD: Removing route from DHCP DP table, lifetime expired,
2001:DB8:100::/48 with nexthop 2001:DB8:1::1 interface ve 1
Oct 27 22:16:48.786 DHCP6 PD: Adding route to DHCP DP table, 2001:DB8:100::/48
with nexthop 2001:DB8:1::1 interface ve 1
Nov  7 19:14:26 DHCP6: IA_PD_PREFIX Option length is not correct.
Nov  7 19:15:26 DHCP6: IA_PD Option has STATUS OPTION set to FAIL.
```

debug ipv6 dhcp pd static

Syntax: [no] debug ipv6 dhcp pd static

This command enables debugging of DHCPv6 interface with IPv6 static route module. Command output resembles the following example.

```
Brocade# debug ipv6 dhcp pd static
Nov  7 19:12:26 DHCP6 PD: Adding static route 2001:DB8:100::/48 with nexthop
2001:DB8:1::1 interface eth 1/9
```

debug ipv6 dhcp sync

Syntax: [no] debug ipv6 dhcp sync

This command enables debugging of DHCPv6 synchronization between active and standby MPs. Command output resembles the following example.

```
Brocade# debug ipv6 dhcp sync
Nov  7 19:14:02 DHCP6: dhcp6_sync_dhcp6_msg_with_standby called
Nov  7 19:14:02 DHCP6: dhcp6_sync_node_pack_dhcp6_msg_to_standby called
Nov  7 19:14:02 DHCP6: dhcp6_sync_node_ack_dhcp6_msg_from_standby called
```

IPv6 neighbor discovery debug commands

This section describes the debug commands that are helpful in debugging neighbor discovery for IPv6 (ND6) feature.

debug ipv6 ra

Syntax: debug ipv6 ra

This command displays debugging information related to ND6 Router Advertisement (RA) messages.

```
Brocade# debug ipv6 ra
Jun 28 10:27:13.147 ICMPv6-RA: DNS server list with lifetime 800
Jun 28 10:27:13.147 DNS address 1::2
Jun 28 10:27:13.147 DNS address 1::1
Jun 28 10:27:13.147 ICMPv6-RA: Domain name list with lifetime 800
Jun 28 10:27:13.147 dnssl_config->domain_name abc.com, dns_name 3abc3com
```

Inter-VRF routing

Inter-VRF routing allows leaking of specific route prefixes from one Virtual Routing and Forwarding (VRF) instance to another VRF on the same router, thereby eliminating the need for external routing or loopback interfaces to selectively access the network. This feature is useful in cases where all VRFs share the same path to reach an external domain, while maintaining their internal routing information limited to their own VRF.

Inter-VRF routing show commands

This section describes the show commands that display information related to inter-VRF routing.

show ip rtm inter-vrf-list

Syntax: **show ip rtm inter-vrf-list**

This command displays a global list of IPv4 VRF nodes processed by the Routing Table Manager (RTM) every 100 ms. The elements in the list are either importing routes from other VRFs or exporting routes to other VRFS. Command output resembles the following example.

```
Brocade# show ip rtm inter-vrf-list
List of vrf nodes which are processed for every 100msec
VRF :vpn1
VRF :vpn5
VRF :vpn7
VRF :vpn8
VRF :vpn4
```

show ipv6 rtm inter-vrf-list

Syntax: **show ipv6 rtm inter-vrf-list**

This command displays a global list of IPv6 VRF nodes processed by the RTM every 100 ms. Command output resembles the following example.

```
Brocade# show ipv6 rtm inter-vrf-list
List of ipv6 vrf nodes which are processed for every 100msec
VRF :default-vrf
VRF :vpn4
```

show ip rtm

Syntax: **show ip rtm** [**vrf** *vrf-name*] [**inter-vrf-route-change-list** | **inter-vrf-imported-vrf-list** | **inter-vrf-export-list**]

- **vrf** *vrf-name* - Specifies the name of the VRF instance.
- **inter-vrf-route-change-list** - Displays a list of changed IP route entries for a specific VRF, which is yet to be leaked to another VRF.
- **inter-vrf-imported-vrf-list** - Displays a list of imported route entries for a specific VRF.
- **inter-vrf-export-list** - Displays a list of exported route entries for a specific VRF.

The **show ip rtm vrf** *vrf-name* **inter-vrf-route-change-list** command displays a list of changed IP route entries for a specified VRF, which is yet to be leaked to another VRF. Command output resembles the following example.

```
Brocade# show ip rtm vrf vpn1 inter-vrf-route-change-list
```

```
Leaked route list count: 208 for VRF:vpn1
        IP address: 10.13.34.0, prefix_length:24, type:4, sub_type: 3, action: 2, 
time_stamp: 18m33s
        IP address: 10.13.35.0, prefix_length:24, type:4, sub_type: 3, action: 2, 
time_stamp: 18m33s
        IP address: 10.13.36.0, prefix_length:24, type:4, sub_type: 3, action: 2, 
time_stamp: 18m33s
        IP address: 10.13.37.0, prefix_length:24, type:4, sub_type: 3, action: 2, 
time_stamp: 18m33s
        IP address: 10.13.38.0, prefix_length:24, type:4, sub_type: 3, action: 2, 
time_stamp: 18m33s
        IP address: 10.13.39.0, prefix_length:24, type:4, sub_type: 3, action: 2, 
time_stamp: 18m33s
        IP address: 10.13.40.0, prefix_length:24, type:4, sub_type: 3, action: 2, 
time_stamp: 18m33s
        IP address: 10.13.41.0, prefix_length:24, type:4, sub_type: 3, action: 2, 
time_stamp: 18m33s
        IP address: 10.13.42.0, prefix_length:24, type:4, sub_type: 3, action: 2, 
time_stamp: 18m33s
        IP address: 10.13.43.0, prefix_length:24, type:4, sub_type: 3, action: 2, 
time_stamp: 18m33s
        IP address: 10.13.44.0, prefix_length:24, type:4, sub_type: 3, action: 2, 
time_stamp: 18m33s
        IP address: 10.13.45.0, prefix_length:24, type:4, sub_type: 3, action: 2, 
time_stamp: 18m33s
        IP address: 10.13.46.0, prefix_length:24, type:4, sub_type: 3, action: 2, 
time_stamp: 18m33s
        IP address: 10.13.47.0, prefix_length:24, type:4, sub_type: 3, action: 2, 
time_stamp: 18m33s
        IP address: 10.13.48.0, prefix_length:24, type:4, sub_type: 3, action: 2, 
time_stamp: 18m33s
        IP address: 10.13.49.0, prefix_length:24, type:4, sub_type: 3, action: 2, 
time_stamp: 18m33s
```

The **show ip rtm vrf** *vrf-name* **inter-vrf-imported-vrf-list** command displays a list of source VRFs with configuration changes, the last VRF index, and the route index processed in the last 100 ms. Command output resembles the following example.

```
Brocade# show ip rtm vrf vpn8 inter-vrf-imported-vrf-list
Imported list for VRF:vpn8
        VRF:vpn7, index: 5, route_map:ml
```

The **show ip rtm vrf** *vrf-name* **inter-vrf-export-list** command displays a list of exported route entries for the specified VRF. Command output resembles the following example.

```
Brocade# show ip rtm vrf vpn7 inter-vrf-export-list
exported list for VRF:vpn7
        VRF:vpn8, index: 6
```

show ipv6 rtm

Syntax: **show ipv6 rtm** [**vrf** *vrf-name*] [**inter-vrf-route-change-list** | **inter-vrf-imported-vrf-list** | **inter-vrf-export-list**]

- **vrf** *vrf-name* - Specifies the name of the VRF instance.
- **inter-vrf-route-change-list** - Displays a list of changed IPv6 route entries for a specific VRF, which is yet to be leaked to another VRF.
- **inter-vrf-imported-vrf-list** - Displays a list of imported route entries for a specific VRF.
- **inter-vrf-export-list** - Displays a list of exported route entries for a specific VRF.

The **show ipv6 rtm vrf** *vrf-name* **inter-vrf-route-change-list** command displays a list of changed IPv6 route entries for a specific VRF, which is yet to be leaked to another VRF. Command output resembles the following example.

```
Brocade# show ipv6 rtm vrf vpn1 inter-vrf-route-change-list
Leaked route list count: 208 for VRF:vpn1
        IPv6 address: 2001:DB8:73::73, prefix_length:24, type:4, sub_type: 3,
action: 2, time_stamp: 18m33s
        IPv6 address: 2001:DB8:83::83, prefix_length:24, type:4, sub_type: 3,
action: 2, time_stamp: 18m33s
        IPv6 address: 2001:DB8:93::93, prefix_length:24, type:4, sub_type: 3,
action: 2, time_stamp: 18m33s
```

The **show ipv6 rtm vrf** *vrf-name* **inter-vrf-imported-vrf-list** command displays a list of imported route entries for a specific VRF. Command output resembles the following example.

```
Brocade# show ipv6 rtm vrf vpn8 inter-vrf-imported-vrf-list
Imported list for VRF:vpn8
        VRF:vpn7, index: 5, route_map:ml
```

The **show ipv6 rtm vrf** *vrf-name* **inter-vrf-export-list** command displays a list of exported route entries for a specific VRF. Command output resembles the following example.

```
Brocade# show ipv6 rtm vrf vpn7 inter-vrf-export-list
exported list for VRF:vpn7
        VRF:vpn8, index: 6
```

Inter-VRF routing debug commands

This section describes the debug commands related to inter-VRF routing.

debug ip rtm inter-vrf-routing

Syntax: [**no**] **debug ip rtm inter-vrf-routing**

This command enables inter-VRF routing debugging for IPv4 routes. Command output resembles the following example.

```
Brocade# debug ip rtm inter-vrf-routing
Mar 26 11:11:24.436 VRF:vpn1 is added to imported list of VRF:vpn4 with
route_map:ospfmetric
Mar 26 11:11:24.436 import route-map is changed for vrf vpn4 for src-vrf vpn1
Mar 26 11:11:24.436 VRF:vpn1 is added to changed list of VRF:vpn4 with flags :1
Mar 26 11:11:24.436 VRF:vpn4 is added to export list of VRF vpn1
Mar 26 11:11:24.517 Processing the changed node list for VRF:vpn4
Mar 26 11:11:24.517 Processing the VRF:vpn1 operation:1,clear_ip_route:0,
cleaning_stale_entries:0, clear_ip_route_op_pending:0
Mar 26 11:11:24.614 Processing the changed node list for VRF:vpn4
Mar 26 11:11:24.614 Processing the VRF:vpn1 operation:1,clear_ip_route:0,
cleaning_stale_entries:0, clear_ip_route_op_pending:0
```

debug ipv6 rtm inter-vrf-routing

Syntax: [**no**] **debug ipv6 rtm inter-vrf-routing**

This command enables inter-VRF routing debugging for IPv6 routes. Command output resembles the following example.

```
Brocade# debug ipv6 rtm inter-vrf-routing
```

```
Mar 26 11:12:26.127 Processing route-map:ospfmetric change for VRF: default-vrf
IPv4 routes
Mar 26 11:12:26.128 Processing route-map:ospfmetric change for VRF: default-vrf
IPv6 routes
Mar 26 11:12:26.128 Processing route-map:ospfmetric change for VRF: vpn1 IPv4
routes
Mar 26 11:12:26.128 Processing route-map:ospfmetric change for VRF: vpn5 IPv4
routes
Mar 26 11:12:26.128 Processing route-map:ospfmetric change for VRF: vpn7 IPv4
routes
Mar 26 11:12:26.128 Processing route-map:ospfmetric change for VRF: vpn8 IPv4
routes
Mar 26 11:12:26.128 Processing route-map:ospfmetric change for VRF: vpn4 IPv4
routes
Mar 26 11:12:26.128 VRF:vpn1 is added to changed list of VRF:vpn4 with flags :3
Mar 26 11:12:26.128 Processing route-map:ospfmetric change for VRF: vpn4 IPv6
routes
Mar 26 11:12:26.216 Processing the changed node list for VRF:vpn4
Mar 26 11:12:26.216 Processing the VRF:vpn1 operation:3,clear_ip_route:0,
cleaning_stale_entries:0, clear_ip_route_op_pending:0
Mar 26 11:12:26.319 Processing the changed node list for VRF:vpn4
Mar 26 11:12:26.319 Processing the VRF:vpn1 operation:3,clear_ip_route:0,
cleaning_stale_entries:0, clear_ip_route_op_pending:0
Mar 26 11:12:26.415 Processing the changed node list for VRF:vpn4

Mar 26 11:14:43.018 Processing the changed route list for VRF:vpn1
Mar 26 11:14:43.018 Leaking Changed route:10.100.100.0/24 to VRF:vpn4
Mar 26 11:14:43.018 Leaking Changed route:10.100.100.0/24 to VRF:vpn5
Mar 26 11:14:43.120 Processing the changed route list for VRF:vpn1
Mar 26 11:14:43.120 Leaking Changed route:10.10.10.0/24 to VRF:vpn4
Mar 26 11:14:43.120 Leaking Changed route:10.10.10.0/24 to VRF:vpn5
Mar 26 11:14:43.121 Leaking Changed route:10.10.11.0/24 to VRF:vpn4
Mar 26 11:14:43.121 Leaking Changed route:10.10.11.0/24 to VRF:vpn5
Mar 26 11:14:43.121 Leaking Changed route:10.10.12.0/24 to VRF:vpn4
Mar 26 11:14:43.121 Leaking Changed route:10.10.12.0/24 to VRF:vpn5
Mar 26 11:14:43.121 Leaking Changed route:10.10.13.0/24 to VRF:vpn4
Mar 26 11:14:43.121 Leaking Changed route:10.10.13.0/24 to VRF:vpn5
Mar 26 11:14:43.121 Leaking Changed route:10.10.14.0/24 to VRF:vpn4
```

RTM failure counter API

Routing Table Manager (RTM) failure counters store information about the number of failed operations along with the number of routes added, modified, and deleted in the routing table on a per-protocol basis.

RTM API show commands

This section describes the show commands that display information about RTM counters.

show ip rtm api

Syntax: show ip rtm api

This command displays information about RTM failure statistics along with the number of IP routes added, modified, and deleted in the routing table on a per-protocol basis. Command output resembles the following example.

```
Brocade# show ip rtm api
RTM API Call Stats for vrf default-vrf safi 0:

            Total  Local   Connect  Static   RIP  OSPF  ISIS   BGP
 RTM_ADD :    10     0       5        2      1     1    1      0
 RTM_MOD :     8     0       5        2      1     0    0      0
 RTM_DEL :     4     0       2        2      0     0    0      0
            Total
 RTM_ADD_LABEL : 0
 RTM_DEL_LABEL : 0
 RTM_RE_ENTRY  : 0

 RTM_ADD_FAIL   Total  Local   Connect  Static   RIP  OSPF  ISIS   BGP
   RTM_RESET   : 4      0        2        0      1     0     1      0
   INVALID_COST: 3      0        1        2      0     0     0      0
   INVALID_PATH: 4      0        2        2      0     0     0      0
   MALLOC_FAIL : 9      0        5        1      1     1     1      0
   FILTER_DENY : 4      0        2        2      0     0     0      0

RTM_MOD_FAIL
  RTM_RESET    : 1      0        0        1      0     0     0      0
  INVALID_COST : 3      0        1        2      0     0     0      0
  MALLOC_FAIL  : 6      0        3        2      1     0     0      0
  INVALID_ROUTE: 4      0        2        2      0     0     0      0

RTM_DEL_FAIL
  INVALID_ROUTE : 1     0        0        1      0     0     0      0

RTM API Call Stats for vrf default-vrf safi 1:

            Total   Connect    Static    BGP
  RTM_ADD :    1        1         0       0
  RTM_MOD :    0        0         0       0
  RTM_DEL :    0        0         0       0
                 Total
  RTM_ADD_LABEL :  0
  RTM_DEL_LABEL :  0
  RTM_RE_ENTRY  :  0

  RTM_ADD_FAIL     Total  Connect   Static    BGP
    RTM_RESET   :    2        1         0      1
    INVALID_COST:    4        1         1      2
    PATH_FAIL   :    5        1         2      2
    MALLOC_FAIL :   10        1         5      1
 FILTER_DENY :    5        1         2      2

  RTM_MOD_FAIL
    RTM_RESET   :   2        1         0      1
    INVALID_COST:   4        1         1      2
    MALLOC_FAIL :   7        1         3      2
    INVALID_ROUTE : 5        1         2      2

  RTM_DEL_FAIL
    INVALID_ROUTE : 2        1         0      1
```

show ipv6 rtm api

Syntax: show ipv6 rtm api

This command displays information about RTM failure statistics along with the number of IPv6 routes added, modified, and deleted in the routing table on a per-protocol basis. Command output resembles the following example.

```
Brocade# show ipv6 rtm api
RTM API Call Stats for vrf default-vrf safi 0:

            Total  Local   Connect  Static   RIP  OSPF  ISIS   BGP
 RTM_ADD :    10      0       5         2      1     1    1      0
 RTM_MOD :     8      0       5         2      1     0    0      0
 RTM_DEL :     4      0       2         2      0     0    0      0
           Total
 RTM_RE_ENTRY  : 0

 RTM_ADD_FAIL   Total  Local   Connect  Static   RIP  OSPF  ISIS   BGP
   RTM_RESET :   4      0        2         0      1     0     1      0
   INVALID_COST: 3      0        1         2      0     0     0      0
   INVALID_PATH: 4      0        2         2      0     0     0      0
   MALLOC_FAIL : 9      0        5         1      1     1     1      0
   FILTER_DENY : 4      0        2         2      0     0     0      0

RTM_MOD_FAIL
  RTM_RESET :    1      0        0         1      0     0     0      0
  INVALID_COST: 3       0        1         2      0     0     0      0
  MALLOC_FAIL :  6      0        3         2      1     0     0      0
  INVALID_ROUTE: 4      0        2         2      0     0     0      0

RTM_DEL_FAIL
  INVALID_ROUTE : 1     0        0         1      0     0     0      0

RTM API Call Stats for vrf default-vrf safi 1:

            Total   Connect    Static    BGP
  RTM_ADD :    1        1         0       0
  RTM_MOD :    0        0         0       0
  RTM_DEL :    0        0         0       0
                 Total
  RTM_RE_ENTRY  :  0

  RTM_ADD_FAIL     Total  Connect   Static    BGP
    RTM_RESET   :    2        1         0      1
    INVALID_COST:    4        1         1      2
    PATH_FAIL   :    5        1         2      2
    MALLOC_FAIL :   10        1         5      1
    FILTER_DENY :    5        1         2      2

  RTM_MOD_FAIL
    RTM_RESET   :   2        1         0      1
    INVALID_COST:   4        1         1      2
    MALLOC_FAIL :   7        1         3      2
    INVALID_ROUTE : 5        1         2      2
  RTM_DEL_FAIL
    INVALID_ROUTE : 2        1         0      1
```

RTM next-hop debug commands

This section describes the debug command that displays information about the RTM next hop and route update.

debug ipv6 rtm nexthop

Syntax: debug ipv6 rtm nexthop

This command enables debugging for the IPv6 nexthop table and displays information about various next-hop related events. Command output resembles the following example.

```
Brocade# debug ipv6 rtm nexthop
Aug  9 16:56:30.200 RTM6(default-vrf/0): allocate_nh id 1597, got id 1597, path 1
(::, ve 60)
Aug  9 16:56:30.200
RTM6: pack ipv6 nh entry: length of packet 55
Aug  9 16:56:30.200 RTM6(default-vrf/0): pack ipv6 nh entry, mode 4, id 1597, path
1 (::, ve 60)
Aug  9 16:56:30.200
RTM6: pack ipv6 route entry: length of packet 89 and each route size is 34
Aug  9 16:56:30.200 RTM6(default-vrf/0): pack ipv6 route 2001:DB8::/64
Aug 9 16:56:30.200RTM6(default-vrf/0): pack ipv6 nh entry to standby MP, mode 4,
id65536, path 1
```

debug ip rtm nexthop

Syntax: debug ip rtm nexthop

This command enables debugging for the IPv4 nexthop table and displays information about various next-hop related events. Command output resembles the following example.

```
Brocade# debug ip rtm nexthop
Aug 29 16:24:51.740 RTM(default-vrf/0): allocate_nh id 65535, got id 65537, path 1
(30.1.1.43, eth 2/1)
Aug 29 16:24:51.740 RTM(default-vrf/0): pack ip nh entry, mode 4, id 65537, path 1
(30.1.1.43, eth 2/1)
Aug 29 16:24:51.840 RTM(default-vrf/0): pack ip nh entry to standby MP, mode 4, id
65537, path 1
```

Chapter

ACL and QoS Diagnostics

7

In this chapter

This chapter provides diagnostic information for Access Control List (ACL) and Quality of Service (QoS) environments, including traffic management.

ACLs

Access Control List (ACL) debug and show commands help users to diagnose and determine the cause of faults for ACL-related features. For details on Layer 2 ACLs, refer to the Layer 2 ACL chapter in the *Brocade MLX Series and NetIron Family Configuration Guide*.

ACL show commands

This section describes the show commands that display ACL information.

show access-list

Syntax: show access-list *number* | **all**

This command displays the IPv4 ACLs configured on a Brocade device.

Enter the ACL number for the *number* variable:

- 1 through 99 for standard ACLs
- 100 through 199 for extended ACLs

Enter **all** to display information for all ACLs configured on the device.

For a numbered ACL, enter a command similar to the following example.

```
Brocade# show access-list 99
ACL configuration:
!
Standard IP access list 10
access-list 99 deny host 10.10.10.1
access-list 99 permit any
```

show access-list name

Syntax: show access-list name *acl-name*

The *acl-name* variable specifies the name of the ACL.

This command displays the named IPv4 ACLs configured on a Brocade device. Command output resembles the following example.

```
Brocade# show access-list name xGW_Filter2

Extended IP access list  xGW_Filter2
    0: permit ip host 10.33.44.55 any
    1: permit ip any any
```

When the display-config-format option is configured, the **show access-list name** command output resembles the following example

```
Brocade# show access-list name xGW_Filter2

ip access-list extended xGW_Filter2
 permit vlan 2405 ip host 10.33.44.55 any
permit vlan 3000 ip any any
```

show access-list accounting brief

Syntax: show access-list accounting brief [**l2** | **policy-based-routing** | **rate-limit**]

- **l2** - Limits the display to Layer 2 ACL accounting information.
- **policy-based-routing** - Limits the display to policy-based routing accounting information.
- **rate-limit** - Limits the display to rate limiting ACL accounting information.

If no option is specified, IPv4 ACL accounting statistics are displayed.

To display a brief summary of the number of hits in all ACLs on a device, enter the following command.

```
Brocade# show access-list accounting brief
Collecting ACL accounting summary for VE 1 ...  Completed successfully.
ACL Accounting Summary: (ac = accumulated since accounting started)
    Int    In ACL              Total In Hit   Out ACL             Total Out Hit
    VE 1    111                473963(1s)
                               25540391(1m)
                               87014178(5m)
                               112554569(ac)
```

show access-list accounting

Syntax: show access-list accounting [**ethernet** *slotnum/portnum* | **pos** *slotnum/portnum* | **ve** *ve-number*] [**in** | **out**] [**l2** | **policy-based-routing** | **rate-limit**]

- **ethernet** *slotnum/portnum* - Displays a report for an Ethernet interface.
- **pos** *slotnum/portnum* - Displays a report for a POS port.
- **ve** *ve-number* - Displays a report for the ports that are included in a virtual routing interface. For example, if ports 1/2, 1/4, and 1/6 are all members of ve 2, the report includes information for all three ports.
- **in** - Displays statistics for incoming traffic.
- **out** - Displays statistics for outgoing traffic.

- **l2** - Limits the display to Layer 2 ACL accounting information.
- **policy-based-routing** - Limits the display to policy-based routing accounting information. This option is only available for incoming traffic.
- **rate-limit** - Limits the display to rate limiting ACL accounting information.

This command displays statistics for an interface, as shown in the following example.

```
Brocade# show access-list accounting ve 1 in
Collecting ACL accounting for VE 1  ...  Completed successfully.
ACL Accounting Information:
Inbound: ACL 111
    1: deny tcp any any
       Hit count: (1 sec)              237000   (1 min)12502822
                  (5 min)            87014178   (accum) 99517000
    3: permit ip any any
       Hit count: (1 sec)              236961   (1 min) 13037569
                  (5 min)                   0   (accum) 13037569
    0: deny tcp 10.1.1.0 10.0.0.255 10.2.2.0 10.0.0.255
       Hit count: (1 sec)                   0   (1 min) 0
                  (5 min)                   0   (accum) 0
    2: deny udp any any
       Hit count: (1 sec)                   0   (1 min) 0
                  (5 min)                   0   (accum) 0
```

show access-list subnet-broadcast accounting

Syntax: **show access-list subnet-broadcast accounting** [**ethernet** *slotnum/portnum* | **pos** *slotnum/portnum* | **ve** *ve-number*]

- **ethernet** *slotnum/portnum* - Specifies the Ethernet interface for which you want to display the accounting information.
- **pos** *slotnum/portnum* - Specifies the POS interface for which you want to display the accounting information.
- **ve** *ve-number* - Specifies the VE interface for which you want to display the accounting information.

This command displays the accounting information for an IP broadcast ACL at the IP interface level, as shown in the following example.

```
Brocade# show access-list subnet-broadcast accounting ethernet 4/1
Subnet broadcast ACL 120
    0: permit udp host 10.10.10.1 host 10.20.20.255
       Hit count: (1 sec)                   0   (1 min)                   0
                  (5 min)                   0   (accum)                   0
    1: permit tcp host 10.10.10.1 host 10.20.20.255
       Hit count: (1 sec)                  10   (1 min)                  67
                  (5 min)                   0   (accum)                  67
    2: deny ip any any
       Hit count: (1 sec)                   0   (1 min)                   0
                  (5 min)                   0   (accum)                   0
```

show access-list subnet-broadcast accounting global

Syntax: **show access-list subnet-broadcast accounting global**

This command displays the accounting information for an IP broadcast ACL at the global level, as shown in the following example.

```
Brocade# show access-list subnet-broadcast accounting global
```

```
Subnet broadcast ACL 1
    0: permit enable-accounting host 10.1.1.2
       Hit count: (1 sec)                 2 (1 min)               150
                  (5 min)                 0 (accum)               384
```

show ipv6 access-list

Syntax: show ipv6 access-list

This command displays the IPv6 ACLs configured on a Brocade device. Command output resembles the following example.

```
Brocade# show ipv6 access-list Test_v6

ipv6 access-list Test_v6: 2 entries
10: permit ipv6 2001:DB8::/64 any
20: deny ipv6 any any
```

When the display-config-format option is configured, the **show ipv6 access-list** command output resembles the following example

```
Brocade# show ipv6 access-list Test_v6

ip access-list extended Test_filter1
permit vlan 112 ip host 10.100.50.1 any
permit vlan 114 udp any eq 2075 any
```

Clearing ACL statistics

Statistics on the ACL account report can be cleared in the following situations:

- When a software reload occurs
- When the ACL is bound to or unbound from an interface
- When you enter the **clear access-list** command

clear access-list

Syntax: clear access-list [**all** | **ethernet** *slotnum/portnum* | **pos** *slotnum/portnum* | **ve** *ve-num*]

- **all** - Clears all statistics for all ACLs.
- **ethernet** *slotnum/portnum* - Clears statistics for ACLs on an Ethernet port.
- **pos** *slotnum/portnum* - Clears statistics for ACLs on a POS port.
- **ve** *ve-num* - Clears statistics for all ACLs bound to ports that are members of a virtual routing interface.

clear access-list subnet-broadcast accounting global

Syntax: clear access-list subnet-broadcast accounting global

This command clears the accounting information for an IP broadcast ACL at the global level.

clear access-list subnet-broadcast accounting

Syntax: clear access-list subnet-broadcast accounting [**ethernet** *slotnum/portnum* | **pos** *slotnum/portnum* | **ve** *ve-number*]

This command clears the accounting information for an IP broadcast ACL at the IP interface level.

The **ethernet**, **pos**, and **ve** keywords specify the interfaces for which you want to clear the accounting information.

ACL debug commands

This section describes how to use diagnostic debug commands to monitor Layer 2 and Layer 4 ACLs for Brocade NetIron XMR series and Brocade MLX series routers.

NOTE
To save space, date and time stamps have been removed from all output examples.

In most Layer 4 instances, a debug command must be accompanied by a user activity, such as binding or unbinding an ACL. In a few instances, background activity for Layer 4 ACLs is displayed simply by enabling the debug function. Many of the debug ACL commands work in conjunction with related show commands, as described in the examples in this chapter.

debug access-list

Syntax: [**no**] **debug access-list** [**accounting** | **ipv4** | **ipv6** | **l2** | **mirror** | **policy-based-routing** | **rate-limit** | **receive** | **lsp-out-acl**]

- **accounting** - Displays ACL accounting statistics.
- **ipv4** - Displays information about IPv4 ACLs.
- **ipv6** - Displays information about IPv6 ACLs.
- **l2** - Displays information about Layer 2 ACLs.
- **mirror** - Displays ACL-based mirroring.
- **policy-based-routing** - Displays information about policy-based routing activity.
- **rate-limit** - Displays information about ACL-based rate limiting.
- **receive** - Displays information about IP receive ACLs.
- **lsp-out-acl** - Displays information about Label Switched Path (LSP) accounting.

debug access-list accounting

Syntax: [**no**] **debug access-list accounting**

This command displays information about inbound or outbound ACL accounting activity. The following example shows inbound rate limit accounting for port 4/2.

```
Brocade# debug access-list accounting
Brocade# show access-list accounting ethernet 4/2 in rate-limit
RL ACL accounting: retrieve for one interface
RL ACL accounting: retrieve for port 4/2, acl 101, outbound 0
RL ACL accounting: retrieve for port 4/2, acl 102, outbound 0
RL ACL Accounting Information:
Inbound: ACL 101
0: permit tcp any any
Hit count: (1 sec) 70 (1 min) 0 (5 min) 0 (accum) 0
1: deny ip any any
Hit count: (1 sec) 0 (1 min) 0(5 min) 0 (accum) 0
ACL 102
0: permit ip 192.168.2.0 10.0.0.255 10.1.1.0 10.0.0.255
Hit count: (1 sec) 0 (1 min) 0 (5 min) 0 (accum) 0
1: deny ip 192.168.2.0 10.0.0.255 10.1.1.0 10.0.0.255
Hit count: (1 sec) 0 (1 min) 0 (5 min) 0 (accum) 0
2: deny ip any any
Hit count: (1 sec) 0 (1 min) 0 (5 min) 0 (accum) 0
```

debug access-list ipv4

Syntax: [no] debug access-list ipv4

This command generates information about IPv4 access list activity. The following example displays information about inbound activity for access group 101.

```
Brocade# debug access-list ipv4
Brocade# ip access-gr 101 in
Bind/Unbind ACL: ACL 101, port 4/2, add 1, outbound 0
Send ITC ACL bind/unbind message: ACL_ID=101, name=, port=121, add=1, dir=in,
mask=
Received ITC ACL bind/unbind message: ACL type 0, ACL 101, add 1, outbound 0
COMMAND << ip access-group 101 in>>
Generated ACL binding command for LP: ip access-group 101 in
```

debug access-list l2

Syntax: [no] debug access-list l2

This command generates information about Layer 2 ACL activity, as shown in the following example, which shows Layer 2 inbound accounting information for interface 4/2.

```
Brocade# debug access-list l2
Brocade# show access-list accounting ethernet 4/2 in l2
L2 ACL accounting: retrieve for one interface
L2 ACL accounting: retrieve for port 4/2, acl 410, outbound 0
Collecting L2 ACL accounting for 410 on port 4/2 ... Completed successfully.
L2 ACL Accounting Information:
Inbound: ACL 410
0: permit 0000.0000.0001 0000.00ff.ffff any any
Hit count: (1 sec) 70 (1 min) 0 (5 min) 0 (accum) 0
1: deny any any any
Hit count: (1 sec) 0 (1 min) 0 (5 min) 0 (accum) 0
```

debug access-list policy-based-routing

Syntax: [no] debug access-list policy-based-routing

This command generates information about access list policy-based routing, as shown in the following example.

```
Brocade# debug access-list policy-based-routing
Policy-Based Routing:  debugging is on
PBR: Routemap refresh timer callback
IPv6 PBR: next_hop 2001:DB8::1 is resolved to port 1/6(5), VLAN 50, DA
0000.008c.ff00
IPv6 PBR: Routemap ipv6_pbr_map: ipv6 nexthop 2001:DB8::1 - resolved
IPV6 PBR: Routemap ipv6_pbr_map instance 10 - number of nexthops 1
IPv6 PBR: PPCR 3:1: Update nexthop entries for ipv6_pbr_map
IPv6 PBR: PPCR 3:1: 3/1: Update nexthop entries for ipv6_pbr_map
IPv6 PBR: PPCR 3:2: Update nexthop entries for ipv6_pbr_map
```

debug access-list rate-limit

Syntax: [no] debug access-list rate-limit

This command generates information about rate limiting for IPv4 access lists. To establish a rate limit for the access group you want to observe, enter commands similar to the following example.

```
Brocade# debug access-list rate-limit
Brocade# rate-limit in access-group 101 500000000 750000000
Brocade# ip access-gr 101 in
Bind/Unbind ACL: ACL 101, port 4/2, add 1, outbound 0
Send ITC ACL bind/unbind message: ACL_ID=101, name=, port=121, add=1, dir=in, mask=
Received ITC ACL bind/unbind message: ACL type 0, ACL 101, add 1, outbound 0
COMMAND << ip access-group 101 in>>
Generated ACL binding command for LP: ip access-group 101 in
ACL-based Rate-Limiting: debugging is ON
```

debug access-list receive generic

Syntax: [no] debug access-list receive generic

This command generates information about generic access list receive activity, as shown in the following example for access-list 101, sequence 10, which shows an ACL bind/unbind message being sent and received.

```
Brocade# debug access-list receive generic
Brocade# ip receive access-list 101 sequence 10
Send ITC Receive ACL bind/unbind message:
    ACL ID 101
    Sequence num 10
    Policy name
    strict acl enabled FALSE
    add 1
    Received ITC Receive-ACL bind/unbind message:
    ACL ID 101
    Sequence num 10
    Policy name
    strict acl enabled FALSE
    add 1
IP Receive ACL: Set global ACL 101 sequence 10 add 1
IP Receive ACL: Create/update ACL 101
Generated IP Receive ACL binding command for LP: ip receive access-list
```

debug access-list lsp-out-acl

Syntax: [**no**] **debug access-list lsp-out-acl** [**cam** | **generic**]

- **cam** - Displays information about LSP CAM programming for IPv4 access lists.
- **generic** - Displays generic LSP accounting information for IPv4 access lists.

The **debug access-list lsp-out-acl cam** command output resembles the following example.

```
Brocade# debug access-list lsp-out-acl cam
LSP outbound ACL CAM/PRAM programming:  debugging is on
May 02  15:22:15 LSP-OUT-ACL: PPCR 4:1: init CAM entry - ppcr hw port index
0x00000000  outer label id 0x00100001
May 02  15:22:15 LSP-OUT-ACL: PPCR 4:1: tunnel vif index 1  egress port id 4/1
May 02  15:22:15     CAM(SW CAM index:0x00017f9d  HW CAM index:0x000c40c4):
May 02  15:22:15     Dest MAC             0000.00f5.de30  (MASK: 0000.00ff.ffff )
May 02  15:22:15     PTYPE[4:0]            0x00000000      (MASK: 0x00000000)
May 02  15:22:15     PPCR port index       0x00000000      (MASK: 0x0000001f)
May 02  15:22:15     Outer Label ID        0x00100001      (MASK: 0x001fffff)
May 02  15:22:15     Inner Label ID        0x00000000      (MASK: 0x00000000)
May 02  15:22:15     PRAM(0x001840c4):
    00000001-00000000-00000000-00000000
    00000000-00000000-00000000-00000000
May 02  15:22:15     FORWARD PACKET    TRUE
    USE QOS ID        FALSE
    QOS ID            0x00000000
May 02  15:22:15 LSP-OUT-ACL: PPCR 4:1: init CAM entry - ppcr hw port index
0x00000000  outer label id 0x00100001
May 02  15:22:15 LSP-OUT-ACL: PPCR 4:1: tunnel vif index 1  egress port id 4/1
May 02  15:22:15     CAM(SW CAM index:0x00017f9d  HW CAM index:0x000c40c4):
May 02  15:22:15     Dest MAC             0000.00f5.de30  (MASK: 0000.00ff.ffff )
May 02  15:22:15     PTYPE[4:0]            0x00000000      (MASK: 0x00000000)
May 02  15:22:15     PPCR port index       0x00000000      (MASK: 0x0000001f)
May 02  15:22:15     Outer Label ID        0x00100001      (MASK: 0x001fffff)
May 02  15:22:15     Inner Label ID        0x00000000      (MASK: 0x00000000)
May 02  15:22:15 LSP-OUT-ACL: PPCR 4:1: init CAM entry - ppcr hw port index
0x00000000  outer label id 0x00100001
May 02  15:22:15 LSP-OUT-ACL: PPCR 4:1: tunnel vif index 1  egress port id 4/1
May 02  15:22:15     CAM(SW CAM index:0x00017f9d  HW CAM index:0x000c40c4):
May 02  15:22:15     Dest MAC             0000.00f5.de30  (MASK: 0000.00ff.ffff )
May 02  15:22:15     PTYPE[4:0]            0x00000000      (MASK: 0x00000000)
May 02  15:22:15     PPCR port index       0x00000000      (MASK: 0x0000001f)
May 02  15:22:15     Outer Label ID        0x00100001      (MASK: 0x001fffff)
May 02  15:22:15     Inner Label ID        0x00000000      (MASK: 0x00000000)
May 02  15:22:45 LSP-OUT-ACL: PPCR 4:1: init CAM entry - ppcr hw port index
0x00000000  outer label id 0x00100001
May 02  15:22:45 LSP-OUT-ACL: PPCR 4:1: tunnel vif index 1  egress port id 4/1
May 02  15:22:45     CAM(SW CAM index:0x00017f9e  HW CAM index:0x000c40c2):
May 02  15:22:45     Dest MAC             0000.00f5.de30  (MASK: 0000.00ff.ffff )
May 02  15:22:45     PTYPE[4:0]            0x00000000      (MASK: 0x00000000)
May 02  15:22:45     PPCR port index       0x00000000      (MASK: 0x0000001f)
May 02  15:22:45     Outer Label ID        0x00100001      (MASK: 0x001fffff)
May 02  15:22:45     Inner Label ID        0x00000000      (MASK: 0x00000000)
May 02  15:22:45     PRAM(0x001840c2):
    00000001-00000000-00000000-00000000
    00000000-00000000-00000000-00000000
May 02  15:22:45     FORWARD PACKET    TRUE
    USE QOS ID        FALSE
    QOS ID            0x00000000
```

```
May 02  15:22:45 LSP-OUT-ACL: PPCR 4:1: init CAM entry - ppcr hw port index
0x00000000  outer label id 0x00100001
May 02  15:22:45 LSP-OUT-ACL: PPCR 4:1: tunnel vif index 1  egress port id 4/1
May 02  15:22:45     CAM(SW CAM index:0x00017f9e  HW CAM index:0x000c40c2):
May 02  15:22:45     Dest MAC             0000.00f5.de30  (MASK: 0000.00ff.ffff )
May 02  15:22:45     PTYPE[4:0]            0x00000000      (MASK: 0x00000000)
May 02  15:22:45     PPCR port index       0x00000000      (MASK: 0x0000001f)
May 02  15:22:45     Outer Label ID        0x00100001      (MASK: 0x001fffff)
May 02  15:22:45     Inner Label ID        0x00000000      (MASK: 0x00000000)
May 02  15:22:45 LSP-OUT-ACL: PPCR 4:1: init CAM entry - ppcr hw port index
0x00000000  outer label id 0x00100001
May 02  15:22:45 LSP-OUT-ACL: PPCR 4:1: tunnel vif index 1  egress port id 4/1
May 02  15:22:45     CAM(SW CAM index:0x00017f9e  HW CAM index:0x000c40c2):
May 02  15:22:45     Dest MAC             0000.00f5.de30  (MASK: 0000.00ff.ffff )
May 02  15:22:45     PTYPE[4:0]            0x00000000      (MASK: 0x00000000)
May 02  15:22:45     PPCR port index       0x00000000      (MASK: 0x0000001f)
May 02  15:22:45     Outer Label ID        0x00100001      (MASK: 0x001fffff)
May 02  15:22:45     Inner Label ID        0x00000000      (MASK: 0x00000000)
```

The **debug access-list lsp-out-acl generic** command output resembles the following example.

```
Brocade# debug access-list lsp-out-acl generic
LSP outbound ACL Generic:  debugging is on
May 02  15:26:35 LSP-OUT-ACL: delete ACL CAM/PRAM entry - tunnel vif index 1
May 02  15:26:35 LSP-OUT-ACL: PPCR 4:1: tnnl vif index 1: delete CAM entry
0x00017f9f
May 02  15:26:35 LSP-OUT-ACL: PPCR 4:1: reset accounting entry at index 0
May 02  15:26:35 LSP-OUT-ACL: PPCR 4:1: updated highest possible valid accounting
index to 0
May 02  15:26:35 LSP-OUT-ACL: delete ACL CAM/PRAM entry - tunnel vif index 1
May 02  15:26:38 LSP-OUT-ACL: nht entry index 0 change notification
May 02  15:26:38 LSP-OUT-ACL: Scheduled timer to update CAM/PRAM entries.
May 02  15:26:38 LSP-OUT-ACL: update load-interval to 300
May 02  15:26:38 LSP-OUT-ACL: add/update ACL CAM/PRAM entry for tunnel vif index 1
May 02  15:26:38 LSP-OUT-ACL: setting BYPASS TUNNEL PKT FLAG TO FALSE bypass_label
4294967295 NHT valid index 0 is_bypass_tnnl_pkt 0
May 02  15:26:38 LSP-OUT-ACL: PPCR 4:1: CAM entries reqd = 1  CAM entries
available = 100
May 02  15:26:38 LSP-OUT-ACL: PPCR 4:1: program CAM entries for egress port 4/1
May 02  15:26:38 LSP-OUT-ACL: PPCR 4:1: no entry found for tunnel vif index 1
egress port 4/1 => create new entry
May 02  15:26:38 LSP-OUT-ACL: PPCR 4:1: found free accounting entry at index 0
May 02  15:26:38 LSP-OUT-ACL: PPCR 4:1: added tunnel vif index 1 egres port id 4/1
to accounting entry 0
May 02  15:26:38 LSP-OUT-ACL: PPCR 4:1: updated highest possible valid accounting
index to 0
May 02  15:26:38 LSP-OUT-ACL: allocate 60 rate buckets for LSP tunnel vif index 1
May 02  15:26:38 LSP-OUT-ACL: reset HW counters for tunnel vif index 1
May 02  15:26:38 Created new LSP tunnel vif index 1 entry - add/update outbound
label ACL entry
May 02  15:26:38 LSP-OUT-ACL: add/update ACL CAM/PRAM entry for tunnel vif index 1
May 02  15:26:38 LSP-OUT-ACL: setting BYPASS TUNNEL PKT FLAG TO FALSE bypass_label
4294967295 NHT valid index 0 is_bypass_tnnl_pkt 0
May 02  15:26:38 LSP-OUT-ACL: PPCR 4:1: CAM entries reqd = 1  CAM entries
available = 99
May 02  15:26:38 LSP-OUT-ACL: PPCR 4:1: program CAM entries for egress port 4/1
May 02  15:26:38 LSP-OUT-ACL: PPCR 4:1: found existing accounting entry 0 for
tunnel vif index 1  egress port 4/1
May 02  15:26:38 LSP-OUT-ACL: PPCR 4:1: updated ACL CAM/PRAM entry - tunnel vif
index 1
```

```
May 02  15:26:38 LSP-OUT-ACL: PPCR 4:1: tunnel vif index 1: update accounting
entry index 0
May 02  15:26:38 LSP-OUT-ACL: PPCR 4:1: updated highest possible valid accounting
index to 0
May 02  15:26:38 LSP-OUT-ACL: Update LSP CAM entries
May 02  15:26:38 LSP-OUT-ACL: tnnl vif index 1 is already programmed
May 02  15:26:38 LSP-OUT-ACL: add/update ACL CAM/PRAM entry for tunnel vif index 1
May 02  15:26:38 LSP-OUT-ACL: setting BYPASS TUNNEL PKT FLAG TO FALSE bypass_label
4294967295 NHT valid index 0 is_bypass_tnnl_pkt 0
May 02  15:26:38 LSP-OUT-ACL: PPCR 4:1: CAM entries reqd = 1  CAM entries
available = 99
May 02  15:26:38 LSP-OUT-ACL: PPCR 4:1: program CAM entries for egress port 4/1
May 02  15:26:38 LSP-OUT-ACL: PPCR 4:1: found existing accounting entry 0 for
tunnel vif index 1  egress port 4/1
May 02  15:26:38 LSP-OUT-ACL: PPCR 4:1: updated ACL CAM/PRAM entry - tunnel vif
index 1
May 02  15:26:38 LSP-OUT-ACL: PPCR 4:1: tunnel vif index 1: update accounting
entry index 0
May 02  15:26:38 LSP-OUT-ACL: PPCR 4:1: updated highest possible valid accounting
index to 0
May 02  15:26:38 LSP-OUT-ACL: Finished searching for all LSPs to be updated or
programmed.
```

debug ipv6 access-list

Syntax: [**no**] **debug ipv6 access-list** [**error** | **ipv6** | **ipv6-cam** | **rate-limit** | **receive** | **stats**]

- **error** - Displays error information about the IPv6 access list.
- **ipv6** - Displays information about the IPv6 access list activity.
- **ipv6-cam** - Displays CAM information about the IPv6 access list.
- **rate-limit** - Enables debugging traces for the IPv6 ACL-based rate limiting.
- **receive** - Enables debugging traces for the IPv6 receive ACLs.
- **stats** - Displays statistical information about the IPv6 access list.

debug ipv6 access-list error

Syntax: [**no**] **debug ipv6 access-list error**

This command generates error information about the IPv6 access list.

Command output resembles the following example.

```
Brocade# debug ipv6 access-list error
Nov 02  17:01:10: PPCR 2:1: L4_CU_ERRNO_OUT_OF_RESOURCE:
actual_ipv6_in_acl_rule_cam_sz 3072, MAX_IPV6_IN_ACL_RULE_CAM_SZ 3072
Nov 02  17:05:10: Send L4 status to MP: port id 2/3, acl id 10, acl name ipv6-pbr,
msg type 6
```

debug ipv6 access-list ipv6

Syntax: [**no**] **debug ipv6 access-list ipv6**

This command generates information about the IPv6 access list activity. The following example assumes an IPv6 traffic filter on incoming traffic to a virtual interface abc10.

Command output resembles the following example.

```
Brocade# debug ipv6 access-list ipv6
Brocade# ipv6 traffic-filter abc10 in
Send ITC ACL bind/unbind message: ACL_ID=0, name=abc10, port=651, add=1, dir=in,
mask=
Received ITC IPv6 ACL bind/unbind message: ACL type 2, ACL abc10, add 1, outbound
0
Set IPv6 ACL abc10 in internal: port number 651, enable 1
COMMAND(b) <<ipv6 traffic-filter abc10 in>>
```

debug ipv6 access-list ipv6-cam

Syntax: [**no**] **debug ipv6 access-list ipv6-cam**

This command generates CAM information about the IPv6 access list.

Command output resembles the following example.

```
Brocade# debug ipv6 access-list ipv6-cam
IPv6:  ACL CAM entry debugging is on
Nov 02  17:01:10
PPCR 2:1: port 2/3: IPv6 ACL CAM sharing disabled, hence group id not allocated
Nov 02  17:01:10 PPCR 2:1: IPv6 inbound ACL CAM entry type 7: ACL v6acl(10) vid
4096 port id 2/3
Nov 02  17:01:10 Src 2001:DB8::/ffff:ffff:ffff:ffff::
    Src port       0x0000/0x0000
    Dst            ::/::
    Dst port       0x0000/0x0000
    Next Header    0x00/0x00
    Est            0x0/0x0
    SYN            0x0/0x0
    Traffic Class  0x00/0x00
    Group id       0x00000002/0x0000001f
    Default VRF    0x00000000/0x00000000
    VLAN           0x00000000/0x00000000
Nov 02  17:01:10     PRAM(0x000000aa):
    001b0200-04000004-00000000-40000003
    edb50104-0000001b-00000000-00000000
Nov 02  17:01:10     OUTER_LABEL           0
Nov 02  17:01:10     PUSH_OUTER_LABEL      0
    INNER_LABEL           0
    DROP_PRECEDENCE[2:0]  0         Bit[2]:If 1, force PRAM drp into packet
    PRAM_HVALID           0x000     HPORT[11:0] Per-port entry valid
    DA HIGH               0x0104
    DA LOW                0x001bedb5
    SUPPRESS RPF DROP     0
    REPLACE_DA            1
    ENFORCE_TNNL_MTU_CHK  0
    NEXT_HOP_INDEX        0
    MULTICAST_VLAN        0
    PRAM SFLOW            0
Nov 02  17:01:10     VLAN_ID               0001
    REPLACE_VLAN          1
    PRAM_ENTRY_TYPE       0
    USE_ALT_SRC_PORT      0
    MONITOR               0
    CPU                   0
    DISCARD INVLD         0
    DISCARD PACKET        0
    USE FID               1
    USE QOS ID            0
```

```
    PRAM_LVALID           0x0       LPORT[11:10] Per-port entry valid
    QOS ID                0x000
    PRAM_LVALID           0x004     LPORT[9:0] Per-port entry valid
    FID                   0x001b
    TRUNK ADJUST          0
    DISABLE_QOS_OVERRIDE  0
    PRIORITY_FORCE        0
    PRIORITY              0
    FORCE_FASTPATH        1
    IGNORE_TTL_CHK        0
        PBR_VLAN_PRESERVE         0
    USE TOS ID            0
    TOS ID                0x000
Nov 02  17:01:10
PPCR 2:1: IPv6 inbound ACL CAM entry type 7: ACL v6acl(20) vid 4096 port id 2/3
Nov 02  17:01:10     Src 2001:DB8::/ffff:ffff:ffff:ffff::
    Src port       0x0000/0x0000
    Dst            ::/::
    Dst port       0x0000/0x0000
    Next Header    0x00/0x00
    Est            0x0/0x0
    SYN            0x0/0x0
    Traffic Class  0x00/0x00
    Group id       0x00000002/0x0000001f
    Default VRF    0x00000000/0x00000000
    VLAN           0x00000000/0x00000000
Nov 02  17:01:10     PRAM(0x000000ab):
    001b0200-04000004-00000000-40000003
    edb50104-0000001b-00000000-00000000
Nov 02  17:01:10     OUTER_LABEL           0
Nov 02  17:01:10     PUSH_OUTER_LABEL      0
    INNER_LABEL           0
    DROP_PRECEDENCE[2:0]  0         Bit[2]:If 1, force PRAM drp into packet
    PRAM_HVALID           0x000     HPORT[11:0] Per-port entry valid
    DA HIGH               0x0104
    DA LOW                0x001bedb5
    SUPPRESS RPF DROP     0
    REPLACE_DA            1
    ENFORCE_TNNL_MTU_CHK  0
    NEXT_HOP_INDEX        0
    MULTICAST_VLAN        0
    PRAM SFLOW            0
Nov 02  17:01:10     VLAN_ID               0001
    REPLACE_VLAN          1
    PRAM_ENTRY_TYPE       0
    USE_ALT_SRC_PORT      0
    MONITOR               0
    CPU                   0
    DISCARD INVLD         0
    DISCARD PACKET        0
    USE FID               1
    USE QOS ID            0
    PRAM_LVALID           0x0       LPORT[11:10] Per-port entry valid
    QOS ID                0x000
    PRAM_LVALID           0x004     LPORT[9:0] Per-port entry valid
    FID                   0x001b
    TRUNK ADJUST          0
    DISABLE_QOS_OVERRIDE  0
    PRIORITY_FORCE        0
    PRIORITY              0
```

```
    FORCE_FASTPATH        1
    IGNORE_TTL_CHK        0
        PBR_VLAN_PRESERVE         0
    USE TOS ID            0
    TOS ID                0x000
Nov 02  17:01:10 PPCR 2:1: port 2/3: IPv6 ACL CAM sharing disabled, hence group id
not allocated
Nov 02  17:01:10
PPCR 2:1: IPv6 inbound ACL CAM entry type 7: ACL v6acl-1(10) vid 4096 port id 2/3
Nov 02  17:01:10     Src 2001:DB8::/ffff:ffff:ffff:ffff::
    Src port       0x0000/0x0000
    Dst            ::/::
    Dst port       0x0000/0x0000
    Next Header    0x00/0x00
    Est            0x0/0x0
    SYN            0x0/0x0
    Traffic Class  0x00/0x00
    Group id       0x00000002/0x0000001f
    Default VRF    0x00000000/0x00000000
    VLAN           0x00000000/0x00000000
Nov 02  17:01:10     PRAM(0x000000ac):
    06f00000-00000004-00000000-00000000
    00000000-00000000-00000000-00000000
Nov 02  17:01:10     OUTER_LABEL           0
Nov 02  17:01:10     PUSH_OUTER_LABEL      0
    INNER_LABEL           0
    DROP_PRECEDENCE[2:0]  0         Bit[2]:If 1, force PRAM drp into packet
    PRAM_HVALID           0x000     HPORT[11:0] Per-port entry valid
    DA HIGH               0x0000
    DA LOW                0x00000000
    SUPPRESS RPF DROP     0
    REPLACE_DA            0
    ENFORCE_TNNL_MTU_CHK  0
    NEXT_HOP_INDEX        0
    MULTICAST_VLAN        0
    PRAM SFLOW            0
Nov 02  17:01:10     VLAN_ID               0000
    REPLACE_VLAN          0
    PRAM_ENTRY_TYPE       0
    USE_ALT_SRC_PORT      0
    MONITOR               0
    CPU                   0
    DISCARD INVLD         0
    DISCARD PACKET        0
    USE FID               0
    USE QOS ID            0
    PRAM_LVALID           0x0       LPORT[11:10] Per-port entry valid
    QOS ID                0x000
    PRAM_LVALID           0x004     LPORT[9:0] Per-port entry valid
    FID                   0x06f0
    TRUNK ADJUST          0
    DISABLE_QOS_OVERRIDE  0
    PRIORITY_FORCE        0
    PRIORITY              0
    FORCE_FASTPATH        0
    IGNORE_TTL_CHK        0
        PBR_VLAN_PRESERVE         0
    USE TOS ID            0
    TOS ID                0x000
```

debug ipv6 access-list rate-limit

Syntax: [no] debug ipv6 access-list rate-limit

This command enables debugging for the IPv6 ACL-based rate limiting. Command output resembles the following example for a configuration flow when the rate limiting feature is enabled.

```
Brocade# debug ipv6 access-list rate-limit
IPv6:  ACL-based Rate-Limiting debugging is on

Dec 17 12:17:22.250 RL: port 4/8: allocate temp_rl_config
Dec 17 12:17:22.250 RL-ACL: port 4/8: init class_id[0] to RL_INVALID_CLASS
Dec 17 12:17:22.250 rl_lp_xpp_add: rate 195456, mxa_burst 10000
Dec 17 12:17:22.250 rl_lp_xpp_add: cir 20, cbs 1250
Dec 17 12:17:22.250 RL-ACL: port 4/8: set rx acl hw config
Dec 17 12:17:22.250 RL-ACL: port 4/8: ACL rl-perm new class needed = TRUE
Dec 17 12:17:22.250 RL-ACL: port 4/8: MAX IN UNIQUE CLASSES 1024, Use priority for
in RL 1
          in_class_pool[3] 35, pri_flag 0
          rl_config->class_id[0] 4096, next_avail_class 35
Dec 17 12:17:22.275 RL-ACL: port 4/8: ACL rl-perm priority 0x00
          init rx class index 35
          cir 20
          eir 0
          mark_value 0
Dec 17 12:17:22.275 RL-ACL: , cbs 1250, ebs 0
Dec 17 12:17:22.275
 rl_lp_set_rx_hw_config : ppcr = 9 class_index = 35  mar_value = 0
Dec 17 12:17:22.275 rl_lp_set_rx_hw_config: , cir 20, eir 0
Dec 17 12:17:22.275 rl_lp_set_rx_hw_config: , cbs 9240, ebs 9240
Dec 17 12:17:22.275 RL-ACL: port 4/8: rl_config->class_id[] = {35, 1059, 2083,
3107}
Dec 17 12:17:22.275 RL-ACL: port 4/8: Schedule callback function to update inbound
rule ACL entries
PPCR 4:1: Cleaning inbound IPv6 rule CAM count...
Dec 17 12:17:24.275 RL-ACL: PPCR 4:1: add ACL policy in CAM
Dec 17 12:17:24.275 RL-ACL: port 4/1: construct list of unique ACLs
Dec 17 12:17:24.275 RL-ACL: port 4/2: construct list of unique ACLs
Dec 17 12:17:24.275 RL-ACL: port 4/3: construct list of unique ACLs
Dec 17 12:17:24.275 RL-ACL: port 4/4: construct list of unique ACLs
Dec 17 12:17:24.275 RL-ACL: port 4/5: construct list of unique ACLs
Dec 17 12:17:24.275 RL-ACL: port 4/5: add ACL policy rl-perm in CAM
Dec 17 12:17:24.275 RL-ACL: port 4/5: ACL rl-perm not defined
Dec 17 12:17:24.275 RL-ACL: port 4/6: construct list of unique ACLs
Dec 17 12:17:24.275 RL-ACL: port 4/6: add ACL policy rl-perm in CAM
Dec 17 12:17:24.275 RL-ACL: port 4/6: ACL rl-perm not defined
Dec 17 12:17:24.275 RL-ACL: port 4/7: construct list of unique ACLs
Dec 17 12:17:24.275 RL-ACL: port 4/7: add ACL policy rl-perm in CAM
Dec 17 12:17:24.275 RL-ACL: port 4/7: ACL rl-perm not defined
Dec 17 12:17:24.275 RL-ACL: port 4/8: construct list of unique ACLs
Dec 17 12:17:24.275           added ACL rl-perm VRF 0, count = 1
Dec 17 12:17:24.275 RL-ACL: port 4/8: add ACL policy rl-perm in CAM
Dec 17 12:17:24.275 RL-ACL: port 4/8: ACL rl-perm not defined
Dec 17 12:17:24.275 RL-ACL: port 4/9: construct list of unique ACLs
Dec 17 12:17:24.275 RL-ACL: port 4/10: construct list of unique ACLs
Dec 17 12:17:24.275 RL-ACL: port 4/11: construct list of unique ACLs
Dec 17 12:17:24.275 RL-ACL: port 4/12: construct list of unique ACLs
Dec 17 12:17:24.275 RL-ACL: port 4/13: construct list of unique ACLs
Dec 17 12:17:24.275 RL-ACL: port 4/14: construct list of unique ACLs
Dec 17 12:17:24.275 RL-ACL: port 4/15: construct list of unique ACLs
```

```
Dec 17 12:17:24.275 RL-ACL: port 4/16: construct list of unique ACLs
Dec 17 12:17:24.275 RL-ACL: port 4/17: construct list of unique ACLs
Dec 17 12:17:24.275 RL-ACL: port 4/18: construct list of unique ACLs
Dec 17 12:17:24.275 RL-ACL: port 4/19: construct list of unique ACLs
Dec 17 12:17:24.275 RL-ACL: port 4/20: construct list of unique ACLs
Dec 17 12:17:24.275 RL-ACL: port 4/21: construct list of unique ACLs
Dec 17 12:17:24.275 RL-ACL: port 4/22: construct list of unique ACLs
Dec 17 12:17:24.275 RL-ACL: port 4/23: construct list of unique ACLs
Dec 17 12:17:24.275 RL-ACL: port 4/24: construct list of unique ACLs
```

debug ipv6 access-list receive

Syntax: [**no**] **debug ipv6 access-list receive** [**time** | **cam** | **generic** | **packet**]

- **time** - Enables debugging traces for the IPv6 receive ACL binding time.
- **cam** - Enables debugging traces for the IPv6 receive ACL CAM and PRAM programming.
- **generic** - Enables debugging traces for the IPv6 receive ACL generic events.
- **packet** - Enables debugging traces for the IPv6 receive ACL related packet.

The following is the sample output from the **debug ipv6 access-list receive generic** command.

```
Brocade# debug ipv6 access-list receive generic
Brocade(config)# ipv6 receive access-list b1 sequence 10
Dec 18 11:51:42.021 Send ITC Receive ACL bind/unbind message:
  ACL b1
  Sequence num 10
  Policy name
  strict acl enabled FALSE
  add 1
Dec 18 11:51:42.021 Received ITC Receive-ACL bind/unbind message:
  ACL b1
  Sequence num 10
  Policy name
  strict acl enabled FALSE
  add 1
Dec 18 11:51:42.021 IPv6 Receive ACL: Set global IPv6 ACL b1 sequence 10 add 1
Dec 18 11:51:42.021 IPv6 Receive ACL: Create/update ACL b1
Dec 18 11:51:42.021 Generated IPv6 Receive ACL binding command for LP: ipv6
receive access-list b1 sequence 10 , mode 14
```

debug ipv6 access-list stats

Syntax: [**no**] **debug ipv6 access-list stats**

This command generates statistical information about the IPv6 access lists. In the following example, with this command enabled, the **show ipv6 access-list accounting brief** command generates a brief accounting summary.

```
Brocade# debug ipv6 access-list stats
Brocade# show ipv6 access-list accouting ethernet 4/2
IPv6 ACL accounting: interface 122, port id 121
IPv6 ACL accounting: retrieve for port 4/2, acl abc10, outbound 0
IPv6 inbound ACL accounting: abc10 filter from 0, num 1
Collecting IPv6 ACL accounting for 4/2  ...  Completed successfully.
IPv6 ACL Accounting Information:
Inbound: IPv6 ACL abc10
   10:  permit tcp any any
       Hit count: (1 sec)                   99   (1 min)                 1515
                  (5 min)                    0   (accum)                 1515
Brocade# show ipv6 access-list accounting brief
IPv6 in/out ACL accounting: retrieve brief information: Max IPv6 interfaces 1193
Collecting IPv6 ACL accounting summary for 4/2 ...  Completed successfully.
IPv6 ACL Accounting Summary: (ac = accumulated since accounting started)
    Int    In ACL             Total In Hit   Out ACL            Total Out Hit
    4/2    abc10                   99(1s)
                                 1515(1m)
                                    0(5m)
                                 1515(ac)
```

Configuration notes

Do not apply an empty ACL (an ACL ID without any corresponding entries) to an interface. If you accidentally do this, the software applies the default ACL action (deny all) to the interface and denies all traffic.

Considerations when implementing ACL CAM sharing

The following considerations apply when implementing the ACL CAM sharing feature:

- If you enable ACL CAM sharing, ACL statistics will be generated per-packet processor (PPCR) instead of per-port. If you require per-port granularity statistics for your application, you cannot use this feature.
- This feature is only applicable for inbound IPv4 ACLs, IPv6 ACLs, VPNv4 ACLs, Layer 2 ACLs, and global Policy Based Routing (PBR) policies.
- This feature is not applicable for ACL-based rate limiting and interface-level PBR policies.
- This feature cannot be applied to a virtual interface.
- CAM entry matching within this feature is based on the ACL group ID.

ACL deny logging

Carefully consider the following guidelines before configuring the ACL deny logging feature on your router:

- The ACL deny logging feature cannot be used in conjunction with the deny traffic redirection feature (the **ip access-group redirect-deny-to-interf** command). If you configure both features on the same interface, the ACL deny logging feature will take precedence and the deny traffic redirection will be disabled. Although disabled, deny traffic redirection will still be shown in the running configuration.
- ACL deny logging is a CPU-based feature. Consequently, to maintain maximum performance, Brocade recommends that you selectively enable the logging option only on the deny filters where you are interested in seeing the logs.

- ACL deny logging generates syslog entries only. No SNMP traps are issued.
- The ACL deny logging feature is supported for inbound ACLs only.
- You can configure the maximum number of ACL session entries using the **system-max session-limit** command as described in the *Brocade MLX Series and NetIron Family Configuration Guide*.
- ACL deny logging is applicable only for traffic matching ACL deny clauses. It is not applicable for traffic matching ACL permit clauses.

Common diagnostic scenarios

- Interrupted policy routing causes severe network problems.

  This issue is resolved by upgrading the software version to include the latest patches. It is recommended that customers always update software to reflect the latest patches and versions. If you have questions about your software version, contact Brocade Technical Support for assistance.

- You need to enable ACL filtering based on VLAN membership or VE port.

  Before you can bind an ACL to specific VLAN members on a port, you must first enable support for the ACL per-port per-VLAN feature. If this feature is not already enabled on your device, enable it using the following commands.

  ```
  Brocade(config)# enable acl-per-port-per-vlan
  Brocade(config)# write memory
  Brocade(config)# exit
  ```

- When an ACL is removed from a port with port mapping (ACL-based rate-limiting) configured, the Brocade MLX series device stops all traffic on this port.

  If you make an ACL configuration change, you must reapply the ACLs to their interfaces for the change to take effect. An ACL configuration change includes any of the following actions:

  - Adding, changing, or removing an ACL or an entry in an ACL
  - Changing a PBR policy
  - Changing ToS-based QoS mappings

  To reapply ACLs following an ACL configuration change, enter the following command at the global CONFIG level of the CLI:

  ```
  Brocade(config)# ip rebind-acl all
  ```

QoS

Quality of Service (QoS) features prioritize the use of bandwidth in a router. When QoS features are enabled, traffic is classified as it arrives at the router, and processed on the basis of configured priorities. Traffic can be dropped, prioritized for guaranteed delivery, or subject to limited delivery options, depending on how you configure QoS features.

QoS show commands

This section describes the show commands that display QoS information.

show qos-tos

Syntax: show qos-tos

This command displays QoS and Type of Service (ToS) configuration information, as shown in the following example.

```
ospf_r_calc 5     wait  0000d89c  2089ff10  16384   0           0        1
  isis_task 5     wait  0000d89c  208a3628  16384   0           0        1
   isis_spf 5     wait  0000d89c  208a8f10  16384   0           0        1
      mcast 5     wait  0000d89c  208ac628  16384   0           0        1
       vrrp 5     wait  0000d89c  208b4628  16384   0           0        1
      ripng 5     wait  0000d89c  208b9628  16384   0           0        1
      ospf6 5     wait  0000d89c  208c3628  16384   0           0        1
   ospf6_rt 5     wait  0000d89c  208c7f08  16384   0           0        1
     mcast6 5     wait  0000d89c  208cb628  16384   0           0        1
         l4 5     wait  0000d89c  208cf620  16384   0           0        1
        stp 5     wait  0000d89c  209a7620  16384   0           0        1
       snmp 5     wait  0000d89c  209c3628  32768   0           0        1
       rmon 5     wait  0000d89c  209cc628  32768   0           0        1
        web 5     wait  0000d89c  209d6628  32768   0           0        1
       lacp 5     wait  0000d89c  209da628  16384   0           0        1
      dot1x 5     wait  0000d89c  209e0620  16384   0           0        1
  hw_access 5     wait  0000d89c  209e6628  16384   0           0        1

DSCP-DSCP map: (dscp = d1d2)
     d2| 0 1 2 3 4 5 6 7 8 9
  d1   |
  -----+----------------------------------------
     0 | 0 1 2 3 4 5 6 7 8 9
     1 | 10 11 12 13 14 15 16 17 18 19
     2 | 20 21 22 23 24 25 26 27 28 29
     3 | 30 31 32 33 34 35 36 37 38 39
     4 | 40 41 42 43 44 45 46 47 48 49
     5 | 50 51 52 53 54 55 56 57 58 59
     6 | 60 61 62 63
```

show qos wred

Syntax: show qos wred

This command displays information about Weighted Random Early Detection (WRED), as shown in the following example.

```
Brocade# show qos wred
QType Enable AverWt    MaxQSz DropPrec MinAvgQSz MaxAvgQSz MaxDropProb MaxPktSz
0     Yes    9(0.19%)  16384  0             5696     16384          2%      16384
                              1             4864     16384          4%      16384
                              2             4096     16384          9%      16384
                              3             3264     16384         10%      16384
1     No
2     No
3     Yes    9(0.19%)  16384  0             6528     16384          2%      16384
                              1             5696     16384          4%      16384
                              2             4864     16384          9%      16384
                              3             4096     16384          9%      16384
4     No
5     No
6     No
7     No
```

show qos-map binding

Syntax: show qos-map binding *slot/port*

This command displays the QoS policy maps that are bounded to the specified port.

```
Brocade# show qos-map binding 4/1
qos pcp encode-policy off
qos exp encode-policy off
qos dscp encode-policy off
```

show qos-map binding global

Syntax: show qos-map binding global

This command displays the QoS policy maps that are bind ed globally on the device, as shown in the following example.

```
Brocade# show qos-map binding global
qos pcp decode-policy default-map
qos pcp encode-policy default-map
qos exp decode-policy default-map
qos exp encode-policy default-map
qos dscp decode-policy default-map
qos dscp encode-policy default-map
```

show qos-map

Syntax: show qos-map [**dscp** | **exp** | **pcp**] **decode-map** [*map-name* | **all-zero-map** | **default-map**]

This command displays information about the QoS ingress decode policy map configuration.

- **dscp** - Displays configuration information for the Differentiated Services Code Point (DSCP) policy map.
- **exp** - Displays configuration information for the Experimental (EXP) policy map.
- **pcp** - Displays configuration information for the Priority Code Point (PCP) policy map.
- *map-name* - Specifies the name of the policy map whose configuration you want to display.
- **all-zero-map** - Displays the all-zero-map configuration information for the specified ingress policy map.
- **default-map** - Displays the default configuration information for the specified ingress policy map.

The following example displays information about the default PCP decode policy map.

```
Brocade# show qos-map pcp decode-map default-map
PCP decode default-map
  PCP  0 to priority  0 drop-precedence  0
  PCP  1 to priority  1 drop-precedence  0
  PCP  2 to priority  2 drop-precedence  0
  PCP  3 to priority  3 drop-precedence  0
  PCP  4 to priority  4 drop-precedence  0
  PCP  5 to priority  5 drop-precedence  0
  PCP  6 to priority  6 drop-precedence  0
  PCP  7 to priority  7 drop-precedence  0
```

show qos-map

Syntax: show qos-map [**dscp** | **exp** | **pcp**] **encode-map** [*map-name* | **all-zero-map** | **default-map**]

This command displays information about the QoS egress encode policy map configuration.

- **dscp** - Displays configuration information for the DSCP policy map.
- **exp** - Displays configuration information for the EXP policy map.
- **pcp** - Displays configuration information for the PCP policy map.
- *map-name* - Specifies the name of the policy map whose configuration you want to display.
- **all-zero-map** - Displays the all-zero-map configuration information for the specified egress policy map.
- **default-map** - Displays the default configuration information for the specified egress policy map.

Command output resembles the following example.

```
Brocade# show qos-map dscp encode-map test2
DSCP encode map test2
Priority 0 drop-precedence 0 to DSCP 0
Priority 0 drop-precedence 1 to DSCP 2
Priority 0 drop-precedence 2 to DSCP 4
Priority 0 drop-precedence 3 to DSCP 6
Priority 1 drop-precedence 0 to DSCP 44
Priority 1 drop-precedence 1 to DSCP 44
Priority 1 drop-precedence 2 to DSCP 44
Priority 1 drop-precedence 3 to DSCP 44
Priority 2 drop-precedence 0 to DSCP 20
Priority 2 drop-precedence 1 to DSCP 25
Priority 2 drop-precedence 2 to DSCP 20
Priority 2 drop-precedence 3 to DSCP 20
Priority 3 drop-precedence 0 to DSCP 55
Priority 3 drop-precedence 1 to DSCP 55
Priority 3 drop-precedence 2 to DSCP 55
Priority 3 drop-precedence 3 to DSCP 55
Priority 4 drop-precedence 0 to DSCP 32
Priority 4 drop-precedence 1 to DSCP 34
Priority 4 drop-precedence 2 to DSCP 36
Priority 4 drop-precedence 3 to DSCP 38
Priority 5 drop-precedence 0 to DSCP 54
Priority 5 drop-precedence 1 to DSCP 54
Priority 5 drop-precedence 2 to DSCP 54
Priority 5 drop-precedence 3 to DSCP 54
Priority 6 drop-precedence 0 to DSCP 48
Priority 6 drop-precedence 1 to DSCP 50
Priority 6 drop-precedence 2 to DSCP 52
Priority 6 drop-precedence 3 to DSCP 54
Priority 7 drop-precedence 0 to DSCP 27
Priority 7 drop-precedence 1 to DSCP 27
Priority 7 drop-precedence 2 to DSCP 27
Priority 7 drop-precedence 3 to DSCP 27
PCP decode default-map
  PCP  0 to priority  0 drop-precedence  0
  PCP  1 to priority  1 drop-precedence  0
  PCP  2 to priority  2 drop-precedence  0
  PCP  3 to priority  3 drop-precedence  0
  PCP  4 to priority  4 drop-precedence  0
  PCP  5 to priority  5 drop-precedence  0
  PCP  6 to priority  6 drop-precedence  0
  PCP  7 to priority  7 drop-precedence  0
```

show qos scheduler

Syntax: show qos scheduler

This command displays the type of QoS scheduler applied at each port. By default strict scheduler is applied at all the physical ports of the device.

```
Brocade# show qos scheduler
Port   | Scheme   Type   Pri7   Pri6   Pri5   Pri4   Pri3   Pri2   Pri1   Pri0
-------+----------------+------+------+------+------+------+------+------+----
   1/1 | strict
   1/2 | weighted Weight 100    10     10     10     10     10     10     10
   3/1 | strict
   3/2 | strict
   3/3 | strict
   3/4 | strict
   4/1 | weighted Weight 100    100    100    1      1      1      1      1
   4/2 | weighted Weight 100    100    100    1      1      1      1      1
   4/3 | strict
   4/4 | strict
   4/5 | strict
   4/6 | strict
   4/7 | strict
   4/8 | strict
```

show tm port-mapping

Syntax: show tm port-mapping

This command displays information about the Traffic Manager (TM) port to queue mapping, as shown in the following example.

```
Brocade# show tm port-mapping
Maximum total TM ports:         1008
Currently used TM ports:        16
Currently available TM ports:   992
```

show np statistics

Syntax: show np statistics

This command displays the Network Processor (NP) statistics for all the interface modules within a device.

NOTE
The display has been shortened for brevity.

```
Brocade# show np statistics
Port 1/1 RX
NP Rx Raw Good Packet          = (667856409355)
NP Rx Forward Packet           = (667823373386)
NP Rx Discard Packet           = (33035969)
NP Rx Misc Packet              = (0)
NP Rx Unicast Packet           = (0)
NP Rx Broadcast Packet         = (2)
NP Rx Multicast Packet         = (667856409353)
NP Rx Send to TM Packet        = (667823373386)
NP Rx Bad Packet               = (0)
NP Rx Lookup Unavailable       = (0)
NP Rx ACL Drop                 = (0)
```

```
NP Rx Priority 0/1 Drop        = (33035968)
NP Rx Priority 2/3 Drop        = (1)
NP Rx Priority 4/5 Drop        = (0)
NP Rx Priority 6/7 Drop        = (0)
NP Rx Suppress RPF Drop        = (0)
NP Rx RPF Drop                 = (0)
NP Rx IPv4 Packet              = (667856409353)
NP Rx IPv6 Packet              = (0)
NP Rx Route-only Drop          = (0)
NP Rx IPv6 Suppress RPF Drop   = (0)
NP Rx IPv6 RPF Drop Count      = (0)
NP Rx IPv4 Byte                = (184327873143207)
NP Rx IPv6 Byte                = (0)
NP Rx Routed Packet Drop       = (0)

Port 1/1 TX
NP Tx Sent to MAC Packet       = (10548)
NP Tx Raw Good Packet          = (10568)
NP Tx Source Port Supress Drop = (20)
NP Tx Bad Packet Count         = (0)
NP Tx Unicast Packet           = (1)
NP Tx Broadcast Packet         = (1)
NP Tx Multicast Packet         = (10546)
NP Tx Receive from TM          = (10568)
NP Tx ACL Drop                 = (0)
NP Tx IPv6 HW Forwarded Packet = (0)
NP Tx IPv4 Packet              = (10546)
NP Tx IPv6 Packet              = (0)
NP Tx IPv4 Byte                = (953896)
NP Tx IPv6 Byte                = (0)

Port 1/2 RX
NP Rx Raw Good Packet          = (119)
NP Rx Forward Packet           = (119)
NP Rx Discard Packet           = (0)
NP Rx Misc Packet              = (0)
NP Rx Unicast Packet           = (0)
NP Rx Broadcast Packet         = (4)
NP Rx Multicast Packet         = (115)
NP Rx Send to TM Packet        = (119)
NP Rx Bad Packet               = (0)
NP Rx Lookup Unavailable       = (0)
NP Rx ACL Drop                 = (0)
NP Rx Priority 0/1 Drop        = (0)
NP Rx Priority 2/3 Drop        = (0)
NP Rx Priority 4/5 Drop        = (0)
NP Rx Priority 6/7 Drop        = (0)
NP Rx Suppress RPF Drop        = (0)
NP Rx RPF Drop                 = (0)
NP Rx IPv4 Packet              = (115)
NP Rx IPv6 Packet              = (0)
NP Rx Route-only Drop          = (0)
NP Rx IPv6 Suppress RPF Drop   = (0)
NP Rx IPv6 RPF Drop Count      = (0)
NP Rx IPv4 Byte                = (9660)
NP Rx IPv6 Byte                = (0)
NP Rx Routed Packet Drop       = (0)
```

show np statistics slot

Syntax: show np statistics slot *slotnum*

This command displays the NP statistics for a specified slot, as shown in the following example.

```
Brocade# show np statistics slot 1
Port 1/1 RX
NP Rx Raw Good Packet          = (667856409355)
NP Rx Forward Packet           = (667823373386)
NP Rx Discard Packet           = (33035969)
NP Rx Misc Packet              = (0)
NP Rx Unicast Packet           = (0)
NP Rx Broadcast Packet         = (2)
NP Rx Multicast Packet         = (667856409353)
NP Rx Send to TM Packet        = (667823373386)
NP Rx Bad Packet               = (0)
NP Rx Lookup Unavailable       = (0)
NP Rx ACL Drop                 = (0)
NP Rx Priority 0/1 Drop        = (33035968)
NP Rx Priority 2/3 Drop        = (1)
NP Rx Priority 4/5 Drop        = (0)
NP Rx Priority 6/7 Drop        = (0)
NP Rx Suppress RPF Drop        = (0)
NP Rx RPF Drop                 = (0)
NP Rx IPv4 Packet              = (667856409353)
NP Rx IPv6 Packet              = (0)
NP Rx Route-only Drop          = (0)
NP Rx IPv6 Suppress RPF Drop   = (0)
NP Rx IPv6 RPF Drop Count      = (0)
NP Rx IPv4 Byte                = (184327873143207)
NP Rx IPv6 Byte                = (0)
NP Rx Routed Packet Drop       = (0)

Port 1/1 TX
NP Tx Sent to MAC Packet       = (10548)
NP Tx Raw Good Packet          = (10568)
NP Tx Source Port Supress Drop = (20)
NP Tx Bad Packet Count         = (0)
NP Tx Unicast Packet           = (1)
NP Tx Broadcast Packet         = (1)
NP Tx Multicast Packet         = (10546)
NP Tx Receive from TM          = (10568)
NP Tx ACL Drop                 = (0)
NP Tx IPv6 HW Forwarded Packet = (0)
NP Tx IPv4 Packet              = (10546)
NP Tx IPv6 Packet              = (0)
NP Tx IPv4 Byte                = (953896)
NP Tx IPv6 Byte                = (0)

Port 1/2 RX
NP Rx Raw Good Packet          = (119)
NP Rx Forward Packet           = (119)
NP Rx Discard Packet           = (0)
NP Rx Misc Packet              = (0)
NP Rx Unicast Packet           = (0)
NP Rx Broadcast Packet         = (4)
NP Rx Multicast Packet         = (115)
NP Rx Send to TM Packet        = (119)
NP Rx Bad Packet               = (0)
```

```
NP Rx Lookup Unavailable       = (0)
NP Rx ACL Drop                 = (0)
NP Rx Priority 0/1 Drop        = (0)
NP Rx Priority 2/3 Drop        = (0)
NP Rx Priority 4/5 Drop        = (0)
NP Rx Priority 6/7 Drop        = (0)
NP Rx Suppress RPF Drop        = (0)
NP Rx RPF Drop                 = (0)
NP Rx IPv4 Packet              = (115)
NP Rx IPv6 Packet              = (0)
NP Rx Route-only Drop          = (0)
NP Rx IPv6 Suppress RPF Drop   = (0)
NP Rx IPv6 RPF Drop Count      = (0)
NP Rx IPv4 Byte                = (9660)
NP Rx IPv6 Byte                = (0)
NP Rx Routed Packet Drop       = (0)

Port 1/2 TX
NP Tx Sent to MAC Packet       = (644213434074)
NP Tx Raw Good Packet          = (644215594740)
NP Tx Source Port Supress Drop = (0)
NP Tx Bad Packet Count         = (0)
NP Tx Unicast Packet           = (4)
NP Tx Broadcast Packet         = (3)
NP Tx Multicast Packet         = (644213434067)
NP Tx Receive from TM          = (644215594740)
NP Tx ACL Drop                 = (0)
NP Tx IPv6 HW Forwarded Packet = (0)
NP Tx IPv4 Packet              = (644213434067)
NP Tx IPv6 Packet              = (0)
NP Tx IPv4 Byte                = (177802905908772)
NP Tx IPv6 Byte                = (0)
```

show np statistics ethernet

Syntax: show np statistics ethernet *slot/port*

This command displays the NP statistics for the specified Ethernet interface.

```
Brocade# show np statistics ethernet 1/1
Port 1/1 RX
NP Rx Raw Good Packet          = (667856409355)
NP Rx Forward Packet           = (667823373386)
NP Rx Discard Packet           = (33035969)
NP Rx Misc Packet              = (0)
NP Rx Unicast Packet           = (0)
NP Rx Broadcast Packet         = (2)
NP Rx Multicast Packet         = (667856409353)
NP Rx Send to TM Packet        = (667823373386)
NP Rx Bad Packet               = (0)
NP Rx Lookup Unavailable       = (0)
NP Rx ACL Drop                 = (0)
NP Rx Priority 0/1 Drop        = (33035968)
NP Rx Priority 2/3 Drop        = (1)
NP Rx Priority 4/5 Drop        = (0)
NP Rx Priority 6/7 Drop        = (0)
NP Rx Suppress RPF Drop        = (0)
NP Rx RPF Drop                 = (0)
NP Rx IPv4 Packet              = (667856409353)
NP Rx IPv6 Packet              = (0)
```

```
NP Rx Route-only Drop          = (0)
NP Rx IPv6 Suppress RPF Drop   = (0)
NP Rx IPv6 RPF Drop Count      = (0)
NP Rx IPv4 Byte                = (184327873143207)
NP Rx IPv6 Byte                = (0)
NP Rx Routed Packet Drop       = (0)

Port 1/1 TX
NP Tx Sent to MAC Packet       = (10548)
NP Tx Raw Good Packet          = (10568)
NP Tx Source Port Supress Drop = (20)
NP Tx Bad Packet Count         = (0)
NP Tx Unicast Packet           = (1)
NP Tx Broadcast Packet         = (1)
NP Tx Multicast Packet         = (10546)
NP Tx Receive from TM          = (10568)
NP Tx ACL Drop                 = (0)
NP Tx IPv6 HW Forwarded Packet = (0)
NP Tx IPv4 Packet              = (10546)
NP Tx IPv6 Packet              = (0)
NP Tx IPv4 Byte                = (953896)
NP Tx IPv6 Byte                = (0)
```

QoS debug commands

There are no debug commands specific to QoS.

Configuration notes

- You cannot use advanced ToS-based QoS and other Layer 4 features, such as the following features:
  - IPv4 ACLs and IPv4 ACL-based rate limiting
  - Layer 2 ACLs and Layer 2 ACL-based rate limiting
  - PBR
  - VLAN ID and Inner VLAN ID translation on the same interface
- QoS mappings are globally configurable and apply to all interfaces.
- To place a QoS mapping change into effect, you must enter the **ip rebind-acl all** command at the global CONFIG level of the CLI after making the mapping change. This applies to mappings that are configured using the **qos-tos map** command.
- Because excess traffic is buffered, rate shaping must be used with caution. In general, it is not advisable to rate shape delay-sensitive traffic.

Traffic management

Traffic management show commands

The following sections describe the show commands you can use to display traffic management information.

show tm statistics

Syntax: show tm statistics

This command displays all traffic manager statistics for the port groups that belong to each traffic manager, as shown in the following example.

```
Brocade# show tm statistics
--------- Ports 2/1 - 2/20 ---------
Ingress Counters:
   Total Ingress Pkt Count:                 464418
   EnQue Pkt Count:                         464418
   EnQue Byte Count:                        51904240
   DeQue Pkt Count:                         464418
   DeQue Byte Count:                        51904240
   TotalQue Discard Pkt Count:              0
   TotalQue Discard Byte Count:             0
   Oldest Discard Pkt Count:                0
   Oldest Discard Byte Count:               0
Egress Counters:
   EnQue Pkt Count:                         701812
   EnQue Byte Count:                        78785888
   Discard Pkt Count:                       0
   Discard Byte Count:                      0
--------- Ports 4/1 - 4/20 ---------
Ingress Counters:
   Total Ingress Pkt Count:                 0
   EnQue Pkt Count:                         0
   EnQue Byte Count:                        0
   DeQue Pkt Count:                         0
   DeQue Byte Count:                        0
   TotalQue Discard Pkt Count:              0
   TotalQue Discard Byte Count:             0
   Oldest Discard Pkt Count:                0
   Oldest Discard Byte Count:               0
Egress Counters:
   EnQue Pkt Count:                         0
   EnQue Byte Count:                        0
   Discard Pkt Count:                       0
   Discard Byte Count:                      0
```

NOTE
The byte counts displayed from the **show tm statistics** command incorporate proprietary internal headers of various lengths.

show tm statistics ethernet

Syntax: show tm statistics ethernet *slotnum/portnum*

This command displays traffic manager statistics for a specified Ethernet port group (identified by a slot and port within the group), as shown in the following example.

```
Brocade# show tm statistics ethernet 2/1
--------- Ports 2/1 - 2/20 ---------
Ingress Counters:
   Total Ingress Pkt Count:                  464454
   EnQue Pkt Count:                          464454
   EnQue Byte Count:                         51907696
   DeQue Pkt Count:                          464454
   DeQue Byte Count:                         51907696
   TotalQue Discard Pkt Count:               0
   TotalQue Discard Byte Count:              0
   Oldest Discard Pkt Count:                 0
   Oldest Discard Byte Count:                0
Egress Counters:
   EnQue Pkt Count:                          701866
   EnQue Byte Count:                         78791072
   Discard Pkt Count:                        0
   Discard Byte Count:                       0
```

show tm statistics slot

Syntax: **show tm statistics slot** *slotnum*

This command displays all traffic manager statistics for an interface module identified by slot number, as shown in the following example.

```
Brocade# show tm statistics slot 4
--------- Ports 4/1 - 4/20 ---------
Ingress Counters:
   Total Ingress Pkt Count:                  0
   EnQue Pkt Count:                          0
   EnQue Byte Count:                         0
   DeQue Pkt Count:                          0
   DeQue Byte Count:                         0
   TotalQue Discard Pkt Count:               0
   TotalQue Discard Byte Count:              0
   Oldest Discard Pkt Count:                 0
   Oldest Discard Byte Count:                0
Egress Counters:
   EnQue Pkt Count:                          0
   EnQue Byte Count:                         0
   Discard Pkt Count:                        0
   Discard Byte Count:                       0
```

Clearing traffic management statistics

You can clear traffic management statistics selectively for a specified port group, an interface module, or for an entire Brocade NetIron XMR series and Brocade MLX series router using the **clear tm statistics** command.

clear tm statistics

Syntax: **clear tm statistics** [**ethernet** *slotnum/portnum* | **slot** *slotnum*]

- **ethernet** *slotnum/portnum* - Clears traffic manager statistics for a specific Ethernet interface.
- **slot** *slotnum* - Clears traffic manager statistics for a specific interface module.

Configuration notes

- A traffic manager contains a specific number of ports depending on the interface module. Specifying a particular port and slot gathers statistics for all ports that belong to the same port group.
- The Brocade device classifies packets into one of eight internal priorities. Traffic scheduling allows you to selectively forward traffic according to the forwarding queue that is mapped to one of the following schemes:
  - Strict priority-based scheduling – This scheme guarantees that higher-priority traffic is always serviced before lower-priority traffic. The disadvantage of strict priority-based scheduling is that lower-priority traffic can be starved of any access.
  - WFQ weight-based traffic scheduling – With Weighted Fair Queuing (WFQ) destination-based scheduling enabled, some weight-based bandwidth is allocated to all queues. With this scheme, the configured weight distribution is guaranteed across all traffic leaving an egress port, and an input port is guaranteed allocation in relationship to the configured weight distribution.
  - Mixed strict priority and weight-based scheduling – This scheme provides a mixture of strict priority for the three highest-priority queues and WFQ for the remaining priority queues.

Route map

A route map is a named set of match conditions and parameter settings that the Brocade device can use to modify route attributes and control redistribution of the routes. Route maps contain match statements and set statements. Each route map contains a permit or deny action for routes that match the match statements.

You can use the PBR route map feature to match the IP information based on the ACLs and set routing information in the IP traffic.

Route map show commands

This section describes the show commands that display route map information.

show pbr route-map

Syntax: **show pbr route-map** *map-name*

The *map-name* parameter specifies the name of the route map.

This command checks and displays whether the route map definitions are properly synchronized to the LP. Command output resembles the following example.

```
Brocade# show pbr route-map ipv6-pbr
Routemap ipv6-pbr: (next-hop-lsp defined = 0, next-hop-ip-tunnel defined = 0)
route-map ipv6-pbr permit 10
match ipv6 address  v6acl
set ipv6 next-hop 2001:DB8::1
set ipv6 next-hop 2001:DB8::1
route-map ipv6-pbr permit 20
match ipv6 address  v6acl-1
set ipv6 next-hop 2001:DB8::1
```

show route-map

Syntax: show route-map [*map-name* | **binding** *map-name*]

- *map-name* - Specifies the name of the route map.
- **binding** - Specifies the interface binding definitions.

This command checks and displays whether the route map and interface binding are properly synchronized to the LP.

Command output resembles the following example.

```
Brocade# show route-map binding ipv6-pbr
IPv6 Bindings of ipv6-pbr : 2/3
```

Telemetry solutions

Telemetry solutions provides for logical grouping of related route map instances and information related to them, such as interface to which the route maps are bound to, current output port or VLAN, and ACLs bound to the route map instances.

Telemetry solutions show commands

This section describes the show commands that display information related to route map instances.

show telemetry rule-name

Syntax: show telemetry [**detail**] **rule-name** *name*

[**detail**] - Displays detailed information about each rule name.

name - Specifies the rule name for which you want to display the route map information.

This command displays information related to route map instances that are either bound or not bound to an interface.

If a rule name is not bound to an interface, the command output resembles the following example.

```
Brocade# show telemetry rule-name
Paths with leading * are configured but disabled, entries with + are for IPv6
                               Route-map    ACL         Output   Output
Name              Input        Policy       Match       VLAN     Port(s)/IP
*example          N/A          test1        test2       N/A      N/A
*default-rulename N/A          example      test2       N/A      N/A
```

If a rule name configured on a route map is bound to an interface, the command output resembles the following example.

```
Brocade# show telemetry rule-name
Paths with leading * are configured but disabled, entries with + are for IPv6
                               Route-map    ACL         Output   Output
Name              Input        Policy       Match       VLAN     Port(s)/IP
example           2/1          test1        test2                9/9
default-rulename  2/2          example      test2
```

The **show telemetry detail rule-name** command displays detailed information about all the configured rule names, as shown in the following example.

```
Brocade# show telemetry detail rule-name
Rule name: example
Input: IPv4 - 2/1
Route-map Policy: test1
IPv4 ACL match:  test2
Output: IPv4 - 9/9 9/17 10/1 10/9

Rule name: default-rulename
Input: IPv4 - 2/2
Route-map Policy: example
IPv4 ACL match:  test2
Output:
```

The **show telemetry rule-name** *name* command displays information about the specified rule name, as shown in the following example.

```
Brocade# show telemetry rule-name example
Paths with leading * are configured but disabled, entries with + are for IPv6
                               Route-map    ACL         Output   Output
Name              Input        Policy       Match       VLAN     Port(s)/IP
example           2/1          test1        test2                9/9
```

The **show telemetry detail rule-name** *name* command displays detailed information about the specified rule name, as shown in the following example.

```
Brocade# show telemetry detail rule-name example
Rule name: example
Input: IPv4 - 2/1
Route-map Policy: test1
IPv4 ACL match:  test2
Output: IPv4 - 9/9 9/17 10/1 10/9
```

Chapter

Multicast Diagnostics 8

In this chapter

This chapter provides diagnostic information about IP multicast environments on Brocade NetIron XMR series and Brocade MLX series routers.

IP multicasting

Multicast protocols allow a group or channel to be accessed over different networks by multiple stations (clients) for the receipt and transmission of multicast data.

Brocade devices support two multicast routing protocols—Distance Vector Multicast Routing Protocol (DVMRP) and Protocol-Independent Multicast (PIM) protocol along with the Internet Group Management Protocol (IGMP).

PIM and DVMRP are broadcast and pruning multicast protocols that deliver IP multicast datagrams. DVMRP and PIM build a different multicast tree for each source and destination host group.

DVMRP

Brocade devices provide multicast routing with the Distance Vector Multicast Routing Protocol (DVMRP). DVMRP uses IGMP to manage the IP multicast groups.

DVMRP is a broadcast and pruning multicast protocol that delivers IP multicast datagrams to intended receivers. The receiver registers the interested groups using IGMP. DVMRP builds a multicast delivery tree with the sender forming the root. Initially, multicast datagrams are delivered to all nodes on the tree. Those leaves that do not have any group members send prune messages to the upstream router, noting the absence of a group. The upstream router maintains a prune state for this group for the given sender. A prune state is aged out after a given configurable interval, allowing multicasts to resume.

DVMRP show commands

You can use show commands to display the following DVMRP information:

- DVMRP group information
- DVMRP interface information
- DVMRP multicast cache information
- DVMRP neighbor information
- DVMRP active prune information
- Available multicast resources
- IP multicast route information
- Active multicast traffic information

show ip mcache

Syntax: show ip mcache

This command displays information about the DVMRP multicast cache, as shown in the following example.

```
Brocade# show ip mcache
Total 2 entries
1    (10.2.1.2, 226.1.1.1) in v20 (e2/2)
     L3 (HW) 1: e2/4(VL40)
     fast=1 slow=0 leaf=0 prun=0 frag=0 tag=0 tnnl=0 swL2=0 hwL2=0 swRepl=0
     age=0 fid: 8012,
2    (10.1.1.2, 225.1.1.1) in v10 (e2/1)
     L3 (HW) 1: e2/3(VL30)
     fast=1 slow=0 leaf=0 prun=0 frag=0 tag=0 tnnl=0 swL2=0 hwL2=0 swRepl=0
     age=0 fid: 8011,
Total number of mcache entries 2
```

show ip dvmrp group

Syntax: show ip dvmrp group

This command displays information about DVMRP groups, as shown in the following example.

```
Brocade# show ip dvmrp group
Total number of groups: 2
1    Group 225.1.1.1          Ports
     Group member at  e2/3: v30
2    Group 226.1.1.1          Ports
     Group member at  e2/4: v40
```

show ip dvmrp interface

Syntax: show ip dvmrp interface

This command displays DVMRP interface information, as shown in the following example.

```
Brocade# show ip dvmrp interface
Interface e5/2
TTL Threshold: 1, Enabled, Querier
Local Address: 10.5.1.1
```

```
DR: itself
Neighbor:
  10.5.1.2
Interface e8/1
TTL Threshold: 1, Enabled, Querier
Local Address: 10.8.1.1
DR: itself
Interface v10
TTL Threshold: 1, Enabled, Querier
Local Address: 10.1.1.1 logical Vid=1
DR: itself
Interface v20
TTL Threshold: 1, Enabled, Querier
Local Address: 10.2.1.1 logical Vid=2
DR: itself
Interface v30
TTL Threshold: 1, Enabled, Querier
Local Address: 10.3.1.1 logical Vid=3
DR: itself
Interface v40
TTL Threshold: 1, Enabled, Querier
Local Address: 10.4.1.1 logical Vid=4
DR: itself
```

show ip dvmrp nbr

Syntax: show ip dvmrp nbr

This command displays information about DVMRP neighbors, as shown in the following example.

```
Brocade# show ip dvmrp nbr
Port   Phy_p  Neighbor        GenId    Age   UpTime
e5/2   e5/2   10.5.1.2       00000000  170   440
```

show ip dvmrp prune

Syntax: show ip dvmrp prune

This command displays DVMRP prune information, as shown in the following example.

```
Brocade# show ip dvmrp prune
Port SourceNet       Group           Nbr             Age
                                                     sec
e5/2  10.2.1.2       226.1.1.1      10.5.1.1         60
e5/2  10.1.1.2       225.1.1.1      10.5.1.1         60
```

show ip dvmrp resource

Syntax: show ip dvmrp resource

This command displays information about available multicast resources, as shown in the following example.

```
Brocade# show ip dvmrp resource
                     allocated    in-use available allo-fail  up-limita
DVMRP route               2048        13      2035         0      2048
route interface           2048        13      2035         0      8192
NBR list                   128         1       127         0      1874
prune list                  64         2        62         0       256
graft list                  64         0        64         0       256
mcache                     128         2       126         0      4096
mcache hash link           547         2       545         0  no-limit
graft if no mcache         197         0       197         0  no-limit
IGMP group                 256         2       254         0      2048
pim/dvm intf. group        256         2       254         0  no-limit
pim/dvm global group       256         2       254         0      4096
HW replic vlan            2000         4      1996         0  no-limit
HW replic port            1024         2      1022         0  no-limit
```

show ip dvmrp route

Syntax: show ip dvmrp route

This command displays IP multicast route information, as shown in the following example.

```
Brocade# show ip dvmrp route
P:Parent M:Metric
Total Routes=1
   SourceNet             Mask        Gateway         P      M

   10.1.0.0              10.255.0.0     *           v153   1
 Int v153   phy_p e1/36  nbr        10.1.1.2 age=0 NOT downstream
```

show ip dvmrp traffic

Syntax: show ip dvmrp traffic

This command displays active multicast traffic information, as shown in the following example.

```
Brocade# show ip dvmrp traffic
Port          Probe                    Graft                    Prune
     [Rx      Tx      Dscrd] [Rx      Tx      Dscrd] [Rx      Tx      Dscrd]
e5/2   111     112     0       0       0       0       9       0       1
e8/1   0       220     0       0       0       0       0       0       0
v10    0       211     0       0       0       0       0       0       0
v20    0       210     0       0       0       0       0       0       0
Total 111     1718     0       0       0       0       9       0       1
IGMP Statistics:
  Total Discard/chksum  0/0
```

DVMRP debug commands

This section describes the debug commands used for monitoring the DVMRP environment.

debug ip pim-dvmrp

Syntax: [no] debug ip pim-dvmrp [add-del-oif | bootstrap | clear | event | group | ipc | join-prune | level | nbr-change | route-change | show | source | vlan-id | vpls-id]

- **add-del-oif** - Displays multicast cache (mcache) additions or deletions.
- **bootstrap** - Displays bootstrap messages in detail.
- **clear** - Clears PIM-DVMRP debug settings.
- **event** - Displays information about infrastructure events and callback handling.
- **group** - Displays activity for a specific group.
- **ipc** - Displays information about IPC messages between the management processor and a line processor.
- **join-prune** - Displays information about join or prune messages.
- **level** - Sets the level of debug information from 1 through 3 (3 generates the most detailed information).
- **nbr-change** - Displays information about neighbor port changes.
- **route-change** - Displays information about route change events.
- **show** - Shows PIM-DVMRP debug settings.
- **source** - Displays information about multicast traffic from a specific source.
- **vlan-id** - Displays information about a specified VLAN.
- **vpls-id** - Displays information about a specific VPLS ID.

debug ip pim-dvmrp add-del-oif

Syntax: [**no**] **debug ip pim-dvmrp add-del-oif [stack]**

This command monitors and displays instances of multicast cache activity, such as outbound interface (OIF) additions or deletions. When the **stack** option is selected, this command also generates a stack trace of the add or delete event. Command output is similar to the following example.

```
Brocade# debug ip pim-dvmrp add-del-oif
Added oif v10, e2/1 to (10.10.10.2, 224.225.0.1) entry
```

This example indicates that VLAN 10 on port e2/1 has been added to the OIF table for the multicast stream (10.10.10.2, 224.225.0.1).

debug ip pim-dvmrp clear

Syntax: [**no**] **debug ip pim-dvmrp clear**

This command clears all PIM-DVMRP debug settings.

debug ip pim-dvmrp ipc

Syntax: [**no**] **debug ip pim-dvmrp ipc**

This command displays IPC messages between the management processor and a line processor. Output is similar to the following example, which Indicates a line processor notification has been enabled for the stream (10.10.10.1, 224.255.0.1). This stream originates from port e2/1 on VLAN 10.

```
Brocade# debug ip pim-dvmrp ipc
receive slave messages S_G_CREAT_NOTIF, entry (10.10.10.1, 224.255.0.1)intf v10,
e2/1
```

debug ip pim-dvmrp join-prune

Syntax: [**no**] **debug ip pim-dvmrp join-prune**

This command displays information about join or prune activity. Command output resembles the following example.

```
Brocade# debug ip pim-dvmrp join-prune
PIMDM:Rx Join/Prune from 10.30.30.1, on intf v10, e2/1. RPF Addr 10.20.20.1.ToME 1
```

This example indicates that a join or prune message has been received from 10.30.30.1, on Ethernet port 2/1, VLAN 10 for RPF Address 10.20.20.1. ToME means this message must be processed.

debug ip pim-dvmrp level

Syntax: [**no**] **debug ip pim-dvmrp level** *num*

This command sets the level of detail for debug output. Levels range from 0 through 3, with 3 being the most detailed. The following levels are currently supported:

- 0 - Receive input or send output messages
- 1 - Process control message

debug ip pim-dvmrp nbr-change

Syntax: [**no**] **debug ip pim-dvmrp nbr-change**

This command displays information about neighbor port changes. Command output resembles the following example.

```
Brocade# debug ip pim-dvmrp nbr-change
nbr 10.30.30.1 phy change from e2/1 to e3/1
```

This output indicates that neighbor 10.30.30.1 has changed from port e2/1 to port e3/1.

debug ip pim-dvmrp show

Syntax: [**no**] **debug ip pim-dvmrp show**

This command displays current debug settings for PIM-DVMRP.

```
Brocade# debug ip pim-dvmrp show
debug ip pim is enabled
```

IGMP V2 and V3

The Internet Group Management Protocol (IGMP) allows an IPv4 system to communicate IP multicast group membership information to its neighboring routers. The routers in turn limit the multicast of IP packets with multicast destination addresses to only those interfaces on the router that are identified as IP multicast group members.

In IGMP V2, when a router sends a query to the interfaces, the clients on the interfaces respond with a membership report of multicast groups to the router. The router can then send traffic to these groups, regardless of the traffic source. When an interface no longer needs to receive traffic from a group, it sends a leave message to the router, which in turn sends a group-specific query to that interface to see if any other clients on the same interface are still active.

IGMP V3 provides selective filtering of traffic based on traffic source. A router running IGMP V3 sends queries to every multicast-enabled interface at the specified interval. These general queries determine if any interface wants to receive traffic from the router.

IGMP show commands

This section describes the show commands that display IGMP information.

show ip igmp group

Syntax: **show ip igmp** [**vrf** *vrf-name*] **group** [*group-address* [**detail**] [**tracking**]]

- **vrf** *vrf-name* - Specifies that you want to display IGMP group information for the VRF specified.
- *group-address* - Displays a report for a specific multicast group. Omit the *group-address* parameter for a report for all multicast groups.
- **detail** - Displays the source list of the multicast group.
- **tracking** - Displays information on interfaces that have tracking enabled.

This command displays the status of all IGMP multicast groups on a device, as shown in the following example.

```
Brocade# show ip igmp group
Total 2 entries
-----------------------------------------------------
Idx Group Address    Port   Intf   Mode    Timer Srcs
---+----------------+------+------+-------+-----+----
  1 232.0.0.1        e6/2   v30    include     0   7
  2 226.0.0.1        e6/2   v30    exclude   240   2
                     e6/3   e6/3   include     0   3
Total number of groups 2
```

show ip igmp group detail

Syntax: **show ip igmp group** *group-address* **detail**

This command displays the status of a specific IGMP multicast group, as shown in the following example.

```
Brocade# show ip igmp group 239.0.0.1 detail
Total 2 entries
-----------------------------------------------------
Idx Group Address    Port   Intf   Mode    Timer Srcs
---+----------------+------+------+-------+-----+----
  1 226.0.0.1        e6/2   v30    exclude   218   2
     S: 10.40.40.12
     S: 10.40.40.11
     S: 10.40.40.10
     S: 10.40.40.2          (Age: 218)
     S: 10.40.40.3          (Age: 218)
    226.0.0.1        e6/3   e6/3   include     0   3
     S: 10.30.30.3          (Age: 165)
     S: 10.30.30.2          (Age: 165)
     S: 10.30.30.1          (Age: 165)
```

show ip igmp group tracking

Syntax: show ip igmp group *group-address* **tracking**

If tracking and fast leave are enabled, you can display the list of clients that belong to a particular group by entering a command similar to the one in the following example.

```
Brocade# show ip igmp group 224.1.10.1 tracking
Total 2 entries
-----------------------------------------------------
Idx Group Address    Port   Intf   Mode    Timer Srcs
---+----------------+------+------+-------+-----+----
  1 10.0.0.1        e6/2   v30    exclude   253   3
     S: 10.40.40.12
     S: 10.40.40.11
     S: 10.40.40.10
     S: 10.40.40.2          (Age: 253)
                     C: 10.10.10.1       (Age: 253)
    S: 10.40.40.3          (Age: 253)
                     C: 10.10.10.1       (Age: 253)
    10.0.0.1        e6/3   e6/3   include     0   3
     S: 10.30.30.3          (Age: 196)
                     C: 10.2.0.1         (Age: 196)
     S: 10.30.30.2          (Age: 196)
                     C: 10.2.0.1         (Age: 196)
     S: 10.30.30.1          (Age: 196)
                     C: 10.2.0.1         (Age: 196)
```

show ip igmp static

Syntax: show ip igmp [**vrf** *vrf-name*] **static**

This command displays information about IGMP static memberships.

The **vrf** parameter with the *vrf-name* variable displays IGMP static information for a specified VRF.

```
Brocade# show ip igmp static
Group Address    Interface Port List
----------------+---------+---------
      229.1.0.12       4/1 ethe 4/1
      229.1.0.13       4/1 ethe 4/1
      229.1.0.14       4/1 ethe 4/1
      229.1.0.92       4/1 ethe 4/1
```

show ip igmp interface

Syntax: show ip igmp [**vrf** *vrf-name*] **interface** [**ethernet** *slotnum/portnum* | **pos** *slotnum/portnum* | **ve** *num*]

- **vrf** *vrf-name* - Displays IGMP interface information for the VRF specified by the *vrf-name* variable.
- **ethernet** *slotnum/portnum* - Displays information for a specific Ethernet interface.
- **pos** *slotnum/portnum* - Displays information for a specific POS interface.
- **ve** *num* - Displays information for a specific virtual routing interface.

This command displays the status of a multicast enabled port, as shown in the following example.

```
Brocade# show ip igmp interface
---------+------+---------+---------------+---------+-----+-----+---------
Intf/Port|Groups| Version |Querier        |  Timer  |V1Rtr|V2Rtr|Tracking
         |      |Oper  Cfg|               |OQrr GenQ|     |     |
---------+------+----+----+---------------+----+----+-----+-----+---------
v200           2    2    -                                       Disabled
    e2/15           2    - Self               0   59 No    No
     e2/1           2    - Self               0   59 No    Yes
     e1/5           2    - Self (MCT-Blk)     0   40 No    No
```

show ip igmp traffic

Syntax: show ip igmp [vrf *vrf-name*] traffic

The **vrf** parameter with the *vrf-name* variable displays IGMP traffic information for a specific VRF.

This command displays the traffic status on each virtual routing interface, as shown in the following example.

```
Brocade# show ip igmp traffic
Recv  QryV2 QryV3 G-Qry GSQry MbrV2 MbrV3 Leave  IsIN  IsEX ToIN ToEX ALLOW BLK
v5       29     0     0     0     0     0     0     0     0    0    0     0    0
v18      15     0     0     0     0    30     0    60     0    0    0     0    0
v110      0     0     0     0     0    97     0   142    37    2    2     3    2
Send  QryV1 QryV2 QryV3 G-Qry GSQry
v5        0     2     0     0     0
v18       0     0    30    30     0
v110      0     0    30    44    11
```

show ip igmp settings

Syntax: show ip igmp [vrf *vrf-name*] settings

The **vrf** parameter with the *vrf-name* variable displays IGMP setting information for a specific VRF.

This command displays global IGMP settings or IGMP settings for a specific VRF. Global IGMP settings are shown in the following example.

```
Brocade# show ip igmp  settings
IGMP Global Configuration
  Query Interval            : 125s
  Configured Query Interval : 125s
  Max Response Time         : 10s
  Group Membership Time     : 260s
  Configured Version        : 2
  Operating Version         : 2
```

Clearing the IGMP group membership table

To clear the IGMP group membership table, enter the following command.

clear ip igmp cache

Syntax: clear ip igmp [vrf *vrf-name*] cache

This command clears the IGMP membership for the default router instance or for a specified VRF. Use the **vrf** option with the *vrf-name* variable to clear the traffic information for a specific VRF instance.

Clearing IGMP traffic statistics

To clear statistics for IGMP traffic, enter the following command.

clear ip igmp traffic

Syntax: **clear ip igmp** [**vrf** *vrf-name*] **traffic**

This command clears all the multicast traffic information on all interfaces on the device.

Use the **vrf** option to clear the traffic information for a VRF instance specified by the *vrf-name* variable. This option became available in version 03.5.00 of the Multi-Service IronWare software.

Clearing IGMP group flows

To clear all the IGMP flows, enter the following command at the Privileged EXEC level of the CLI.

clear ip multicast all

Syntax: **clear ip multicast all**

IGMP debug commands

This section describes the debug commands used for monitoring the IGMP environment.

debug ip igmp

Syntax: [**no**] **debug ip igmp**

This command generates information about IGMP activity, including IGMP membership queries, membership responses, and the conversion of IGMP V2 to IGMP V3 through DNS lookup. Command output resembles the following example.

```
Brocade# debug ip igmp
IGMP.RCV: Type Query Port 0/0 PktLen 8. GrpAddr 0.0.0.0 Src: 10.28.172.53
IGMP.RCV: Type Query Port 1/1 PktLen 8. GrpAddr 0.0.0.0 Src: 10.28.172.110
```

debug ip igmp protocol query

Syntax: **debug ip igmp protocol query**

This command displays information about the Internet Group Management Protocol (IGMP) query suppression state on the Cluster Client Edge Ports (CCEPs). Command output resembles the following example.

```
Brocade# debug ip igmp protocol query
Jul  2 04:55:37.273 IGMP.VRF0: [ Port 2/15,v200 ] Sent General Query version 2
using src 10.2.2.1
Jul  2 04:56:03.273 IGMP.VRF0: [ Port 1/5,v200 ] Skipped General Query on CCEP
port
```

debug ip vrf

Syntax: [no] debug ip vrf

This command generates information about synchronization of VRF routing information to line cards, similar to the following example, which shows a download request from the line card to start the tree download for VRF 1.

```
Brocade# debug ip vrf
RTM (vrf): Processing tree download for vrf 1
```

Configuration notes

- Each of the multicast protocols uses IGMP. IGMP is automatically enabled on an interface when you configure PIM or DVMRP on an interface, and is disabled on the interface if you disable PIM or DVMRP on the interface.
- IGMP V3 does not support static IGMP group members.
- Static IGMP groups are supported only in Layer 3 mode.
- Because VLANs are not VRF-aware, any changes to the default VLAN or tagged port moves is counted by all VRFs in existence at the time, including the default VRF.

Common diagnostic scenarios

- With flow-control and IGMP enabled, performance is considerably slower than expected.

  This issue is resolved by upgrading the software version to include the latest patches. It is recommended that customers always update software to reflect the latest patches and versions. If you have questions about your software version, contact Brocade Technical Support for assistance.

- Sporadic difficulty in accessing the Web management console.

  This issue is resolved by upgrading the software version to reflect the latest patches and versions, and disabling the following IGMP settings:

  ```
  no ip igmp snooping
  no ip igmp snooping querier
  ```

- Even though multicast function is not configured, LP and MP CPU usage is high when the Brocade NetIron XMR device receives a large number of invalid IGMP packets.

  The following solutions are proposed to avoid this issue:

  - Enter the following commands to deny IGMP packets:

    ```
    Brocade# ip receive access-list 100 sequence 5
    Brocade# access-list 100 deny igmp any any
    Brocade# access-list 100 permit ip any any
    ```

    Control packets can be flooded to the router and cause high CPU because the CPU must process these packets. Brocade NetIron XMR series and Brocade MLX series devices use IP receive ACL (firewall) to filter these packets.

  - Enter the following commands to deny any IP fragment packets and prevent excessive LP and MP CPU usage:

    ```
    Brocade# ip receive access-list 100 sequence 5
    Brocade# access-list 100 deny ip any any fragment
    Brocade# access-list 100 permit ip any any
    ```

The router will bind this ACL to all interfaces.

Multicast

When you enable IP multicast traffic reduction, you also can configure the following features:

- IGMP mode – When you enable IP multicast traffic reduction, the device passively listens for IGMP Group Membership reports by default. If the multicast domain does not have a router to send IGMP queries to elicit these Group Membership reports, you can enable the device to actively send the IGMP queries.
- Query interval – The query interval specifies how often the device sends Group Membership queries. This query interval applies only to the active IGMP mode. The default is 60 seconds. You can change the interval to a value from 10 through 600 seconds.
- Age interval – The age interval specifies how long an IGMP group can remain in the IGMP group table without the device receiving a Group Membership report for the group. If the age interval expires before the device receives another Group Membership report for the group, the device removes the entry from the table. The default is 140 seconds. You can change the interval to a value from 10 through 1220 seconds.

Furthermore, when you enable IP multicast traffic reduction, the device forwards all IP multicast traffic by default, but you can enable the device to do the following:

- Forward IP multicast traffic only for groups for which the device has received a Group Membership report.
- Drop traffic for all other groups.

NOTE
If the “route-only” feature is enabled on the Brocade device, then IP multicast traffic reduction will not be supported. This feature is also not supported on the default VLAN of the Brocade device.

To verify that IP multicast traffic reduction is enabled, enter the **show ip multicast** command at any level of the CLI.

Multicast show commands

This section describes the show commands that display multicast information.

show ip multicast

Syntax: **show ip multicast**

This command displays IP multicast traffic reduction information, as shown in the following example.

```
Brocade# show ip multicast
IP multicast is enabled - Passive
IP pimsm snooping is enabled

VLAN ID 23
Active 10.10.10.10 Report ports: 1/1 7/1
Report FID 0X0400
Number of Multicast Groups: 2

1      Group: 225.1.0.291
       IGMP report ports  :
       Mapped mac address : 0100.5e01.001d Fid:0x041b
       PIMv2*G join ports : 1/1

2      Group: 225.1.0.24
       IGMP report ports : 4/48
       Mapped mac address : 0100.5e01.0018 Fid:0x041a
       PIMv2*G join ports : 1/1
```

show ip multicast pimsm-snooping

Syntax: show ip multicast pimsm-snooping

This command displays PIM SM information, as shown in the following example.

```
Brocade# show ip multicast pimsm-snooping
PIMSM snooping is enabled
VLAN ID 100
  PIMSM neighbor list:
          10.31.31.4 : 12/2 expires 142 s
          10.31.31.13 : 10/7 expires 136 s
          10.31.31.2 : 3/1 expires 172 s
Number of Multicast Groups: 2
1    Group: 239.255.162.4 Num SG 4
     Forwarding ports : 3/1 12/2
     PIMv2 *G join ports : 3/1 12/2
     1     Source: (10.165.165.165, 10/7) FID 0x0bb3
           SG join ports: 12/2 10/7
     2     Source: (10.161.161.161, 10/7) FID 0x0bb2
           SG join ports: 12/2 3/1
     3     Source: (10.158.158.158, 10/7) FID 0x0bb1
           SG join ports: 12/2 3/1
     4     Source: (10.170.170.170, 10/7) FID 0x0baf
           SG join ports: 3/1 10/7
           (S, G) age 0 s
2    Group: 239.255.163.2 Num SG 1
     Forwarding ports : 10/7 12/2
     PIMv2 *G join ports : 10/7 12/2
     1     Source: (10.165.165.165, 3/1) FID 0x0bb5
            SG join ports: 12/2 10/7
```

show ip multicast statistics

Syntax: show ip multicast statistics

This command displays IP multicast statistics, as shown in the following example, which shows statistics for two port-based VLANs.

```
Brocade# show ip multicast statistics
IP multicast is enabled - Passive

VLAN ID 1
Reports Received:              34
Leaves Received:               21
General Queries Received:      60
Group Specific Queries Received: 2
Others Received:                0
General Queries Sent:           0
Group Specific Queries Sent:    0

VLAN ID 2
Reports Received:               0
Leaves Received:                0
General Queries Received:      60
Group Specific Queries Received: 2
Others Received:                0
General Queries Sent:           0
Group Specific Queries Sent:    0
```

Multicast debug commands

This section describes the debug commands that generate multicast information.

debug [ip | ipv6] multicast add-del-oif

Syntax: **debug** [**ip** | **ipv6**] **multicast add-del-oif**

- **ip** - Displays debug information for IPv4 multicast.
- **ipv6** - Displays debug information for IPv6 multicast.

This command displays information about addition or deletion of the outgoing interfaces (OIFs) to or from the mcache entry.

```
Brocade# debug ipv6 multicast add-del-oif
L2MCASTV6.MLD.ADD_DEL: l2mcast_l2mdb_process_port_down
```

debug [ip | ipv6] multicast error

Syntax: **debug** [**ip** | **ipv6**] **multicast error**

- **ip** - Displays debug information for IPv4 multicast.
- **ipv6** - Displays debug information for IPv6 multicast.

This command displays information about any kind of unexpected error.

```
Brocade# debug ip multicast error
L2MCASTV4.IGMP.ERR: Rx packet has invalid checksum. Dropping packet
```

debug [ip | ipv6] multicast events

Syntax: **debug** [**ip** | **ipv6**] **multicast events**

- **ip** - Displays debug information for IPv4 multicast.
- **ipv6** - Displays debug information for IPv6 multicast.

This command displays information about system events such as VRF changes, interface changes, and so on.

```
Brocade# debug ip multicast events
L2MCAST IGMPV4: l2mcast_mct_itc_process_mdup_message - Received mdup with dest-ip
10.0.0.1 from peer
L2MCASTV4: l2mcast_mct_receive_igmp_mld_packet - Received packet from
MDUP_MSG_FROM_CCEP for dest-ip 10.0.0.1 on VLAN 21
```

debug [**ip** | **ipv6**] **multicast ipc**

Syntax: debug [**ip** | **ipv6**] **multicast ipc**

- **ip** - Displays debug information for IPv4 multicast.
- **ipv6** - Displays debug information for IPv6 multicast.

This command displays debug information about the IPC messages communicated between the MP and the LP.

```
Brocade# debug ipv6 multicast ipc
L2MCASTV6.MLD.IPC: mgmt_send_enable_disable
```

debug [**ip** | **ipv6**] **multicast protocol**

Syntax: debug [**ip** | **ipv6**] **multicast protocol**

- **ip** - Displays debug information for IPv4 multicast.
- **ipv6** - Displays debug information for IPv6 multicast.

This command displays information related to the processing and handling of multicast query or reports.

```
Brocade# debug ip multicast protocol
L2MCASTV4.IGMP: l2mcast_process_igmpv3_mldv2_report
L2MCASTV4.IGMP: Received IGMPV2 report for group 10.0.0.1 from 8/4 on vlan 21
```

Clearing IP multicast statistics

To clear IP multicast statistics on a device, enter the following command at the Privileged EXEC level of the CLI.

clear ip multicast

Syntax: clear ip multicast [**all** | **group** *group-id*]

- **all** - Clears the learned flows for all groups.
- **group** *group-id* - Clears the flows for the specified group, but does not clear the flows for other groups.

This command resets statistics counters to zero for all the statistics displayed by the **show ip multicast statistics** command.

The following example shows IGMP flows information listed by the **show ip multicast** command, followed by removal of the information by the **clear ip multicast all** command.

```
Brocade# show ip multicast
IP multicast is enabled - Active
VLAN ID 1
Active 192.168.2.30 Router Ports  4/13
Multicast Group: 239.255.162.5, Port: 4/4 4/13
Multicast Group: 239.255.162.4, Port: 4/10 4/13

Brocade# clear ip multicast all

Brocade# show ip multicast
IP multicast is enabled - Active
VLAN ID 1
Active 192.168.2.30 Router Ports  4/13
```

To clear the learned IGMP flows for a specific IP multicast group, use the **clear ip multicast group** command.

The following example shows how to clear the IGMP flows for a specific group and retain reports for other groups.

```
Brocade# show ip multicast
IP multicast is enabled - Active
VLAN ID 1
Active 192.168.2.30 Router Ports  4/13
Multicast Group: 239.255.162.5, Port: 4/4 4/13
Multicast Group: 239.255.162.4, Port: 4/10 4/13

Brocade# clear ip multicast group 239.255.162.5
Brocade# show ip multicast
IP multicast is enabled - Active
VLAN ID 1
Active 192.168.2.30 Router Ports  4/13
Multicast Group: 239.255.162.4, Port: 4/10 4/13
```

Configuration notes

You must enter the **ip multicast-routing** command before changing the global IP multicast parameters. Otherwise, the changes do not take effect and the software uses the default values. Also, entering **no ip multicast-routing** will reset all parameters to their default values.

Common diagnostic scenarios

- Multicast is not working.

  VLANs and route-only will not work together. Removing the route-only setting eliminates the problem.

- Multicast packets are not being transferred between two Brocade MLX series devices.

  The two Brocade MLX series devices have network address overlap. The problem is resolved when the network address of one of the devices is changed.

MSDP

The Multicast Source Discovery Protocol (MSDP) is used by Protocol-Independent Multicast (PIM) Sparse routers to exchange routing information for PIM Sparse multicast groups across PIM Sparse domains. Routers running MSDP can discover PIM Sparse sources that are in other PIM Sparse domains.

MSDP show commands

You can display the following MSDP information using show commands:

- Summary information – The IP addresses of the peers, the state of the Brocade device's MSDP session with each peer, and statistics for Keepalive, Source Active, and Notification messages sent to and received from each of the peers.
- Peer information – The IP address of the peer, along with detailed MSDP and TCP statistics.
- Source Active cache entries – The Source Active messages cached by the router.

show ip msdp summary

Syntax: show ip msdp summary

This command displays summary MSDP information, as shown in the following example.

```
Brocade# show ip msdp summary

MSDP Peer Status Summary
 KA: Keepalive SA:Source-Active NOT: Notification
 Peer Address     State        KA          SA          NOT
                           In     Out   In    Out   In   Out
10.251.17.30    ESTABLISH   3      3     0     640   0     0
10.251.17.41    ESTABLISH   0      3     651   0     0     0
```

show ip msdp peer

Syntax: show ip msdp peer

This command displays MSDP peer information, as shown in the following example.

```
Brocade# show ip msdp peer

        Total number of MSDP Peers: 2

        IP Address           State
1       10.251.17.30        ESTABLISHED
        Keep Alive Time  Hold Time
        60               90

                            Message Sent         Message Received
        Keep Alive          2                    3
        Notifications       0                    0
        Source-Active       0                    640
        Last Connection Reset Reason:Reason Unknown
        Notification Message Error Code Received:Unspecified
        Notification Message Error SubCode Received:Not Applicable
        Notification Message Error Code Transmitted:Unspecified
        Notification Message Error SubCode Transmitted:Not Applicable
```

```
TCP Connection state: ESTABLISHED
  Local host:  10.251.17.29, Local  Port: 8270
  Remote host: 10.251.17.30, Remote Port: 639
  ISentSeq:      16927  SendNext:     685654  TotUnAck:          0
  SendWnd:       16384  TotSent:      668727  ReTrans:           1
  IRcvSeq:    45252428  RcvNext:    45252438  RcvWnd:        16384
  TotalRcv:         10  RcvQue:            0  SendQue:           0
```

show ip msdp sa-cache

Syntax: show ip msdp sa-cache

This command displays multiple copies of the same Source Active (SA) entries originating from different Rendezvous Points (RPs) in a sorted order. Command output resembles the following example.

```
Brocade# show ip msdp sa-cache
Total of 10 SA cache entries
Index RP address    (Source,Group)              Orig Peer      Age/Uptime
1     10.4.4.4      (10.200.200.100,227.1.1.1)   10.4.4.4       32 /00:14:07
2     10.5.5.5      (10.200.200.100,227.1.1.1)   172.168.2.2    39 /00:14:07
3     10.6.6.6      (10.200.200.100,227.1.1.1)   172.168.2.2    39 /00:14:05
4     10.7.7.7      (10.200.200.100,227.1.1.1)   172.168.2.2    39 /00:13:39
5     10.4.4.4      (10.200.200.100,227.1.1.2)   10.4.4.4       32 /00:14:07
6     10.5.5.5      (10.200.200.100,227.1.1.2)   172.168.2.2    39 /00:14:07
7     10.6.6.6      (10.200.200.100,227.1.1.2)   172.168.2.2    39 /00:14:05
8     10.7.7.7      (10.200.200.100,227.1.1.2)   172.168.2.2    39 /00:13:39
9     10.4.4.4      (10.200.200.100,227.1.1.3)   10.4.4.4       32 /00:14:07
10    10.5.5.5      (10.200.200.100,227.1.1.3)   172.168.2.2    39 /00:14:07
Total number of matching entries:10
```

show ip msdp debug

Syntax: show ip msdp debug

This command displays information about internal MSDP activity, such as the number of peers, timer settings, internal clock ticks, and Source Active (SA) cache memory pool data. Command output resembles the following example.

```
Brocade# show ip msdp debug
[BEGIN] MSDP Debug Info
Oper is On
Max # of peers 1, # of peers 1
Srvr IP 0.0.0.0, sockInit Yes
Orig-id 0/1/0/0.0.0.0, SA filter-orig No, orig-rmap ""/00000000
entry-per-ticks 3200, adv-int 60, adv-entry-per-tick 35, state-ticks 533, start-
offset 10
SA agetime 6, Holddown 75, KA 60, hold-timer 90, conn-rety 30
SA Cache memory Pool Information:
pool: 25da9600, unit_size: 24, initial_number:256, upper_limit:32000
  total_number:256, allocated_number:10, alloc_failure 0
  flag: 0, pool_index:1, avail_data:25ef60f0
[END] MSDP Debug Info
```

MSDP debug commands

This section describes the debug commands that generate MSDP information.

debug ip msdp

Syntax: [**no**] **debug ip msdp** [**alarms** | **events** | **message**]

- **alarms** - Displays information about MSDP alarms.
- **events** - Displays information about MSDP events.
- **message** - Displays information about MSDP messages.

The **debug ip msdp** command generates information about Multicast Source Discovery Protocol (MSDP) alarms, events, and messages.

debug ip msdp alarms

Syntax: [**no**] **debug ip msdp alarms**

This command generates information about MSDP RX processing errors, such as invalid headers or incomplete or truncated information, errors during transmission of SA advertisement transmission, (for example, buffer unavailability), and peer connection socket errors and notification messages. Command output resembles the following example.

```
Brocade# debug ip msdp alarms
         MSDP:  alarms debugging is on
MSDP: S=xxxxxxx P=0 Initiate Transport Connection to MSDP peer
```

debug ip msdp events

Syntax: [**no**] **debug ip msdp events**

This command tracks originating SA advertisements, major peer events, and peer keepalive timer events. Command output resembles the following example.

```
Brocade# debug ip msdp events
MSDP: events debugging is on
MSDP: 192.1.1.2: Process START event, local = 10.1.1.2
MSDP: 10.1.1.2: TCP Connection to Remote Peer is Open
MSDP: 10.1.1.2: MSDP-TCP Connection opened
MSDP: 10.1.1.2: TCP_OPEN DONE, State 4
MSDP: 10.1.1.2: Originating SA
MSDP: 10.1.1.2: TX Keep Alive timer expired, send keep alive to peer
MSDP: 10.1.1.2: Originating SA
MSDP: 10.1.1.2: TX Keep Alive timer expired, send keep alive to peer
```

debug ip msdp message

Syntax: [**no**] **debug ip msdp message**

This command generates information (including message contents) about MSDP messages received, transmitted, and forwarded, and flag errors in MSDP messages. Command output resembles the following example.

```
Brocade# debug ip msdp message
MSDP: 192.1.1.2: Xmt SA
RP 10.1.1.1, SA count 10
(10.1.2.10,226.1.1.1) (10.1.2.10,226.1.1.2)
(10.1.2.10,226.1.1.3) (10.1.2.10,226.1.1.4)
(10.1.2.10,226.1.1.5) (10.1.2.10,226.1.1.6)
(10.1.2.10,226.1.1.7) (10.1.2.10,226.1.1.8)
MSDP: 10.1.1.2: State=4, Rcv SA
RP 2.2.2.2, SA count 10
(10.2.2.10,225.1.1.1) (10.2.2.10,225.1.1.2)
(10.2.2.10,225.1.1.3) (10.2.2.10,225.1.1.4)
(10.2.2.10,225.1.1.5) (10.2.2.10,225.1.1.6)
(10.2.2.10,225.1.1.7) (10.2.2.10,225.1.1.8)
MSDP: 10.1.1.2: State=4, Rcv KA
```

Clearing MSDP information

You can clear the following MSDP information:

- Peer information
- Source Active cache entries
- MSDP statistics

To clear MSDP peer information, enter the **clear ip msdp peer** command at the Privileged EXEC level of the CLI.

clear ip msdp peer

Syntax: **clear ip msdp peer** *ip-addr*

This command displays a message to indicate when the connection has been successfully closed. To clear all the peers, omit the *ip-addr* variable from the command.

To clear the Source Active cache, enter the **clear ip msdp sa-cache** command at the Privileged EXEC level of the CLI.

clear ip msdp sa-cache

Syntax: **clear ip msdp sa-cache** *ip-addr*

This command clears all of the cache entries. Use the *ip-addr* variable to clear only the entries matching either a source or a group.

To clear MSDP statistics, enter the **clear ip msdp statistics** command at the Privileged EXEC level of the CLI.

clear ip msdp statistics

Syntax: **clear ip msdp statistics** *ip-addr*

This command clears statistics for all the peers. To clear statistics for a specific peer, enter the peer's IP address using the *ip-addr* variable.

Configuration notes

- MSDP depends on BGP and Multiprotocol BGP (MBGP) for inter-domain operations.

- Routers that run MSDP usually also run BGP. The source address used by the MSDP router is normally configured to be the same source address used by BGP.
- For MSDP mesh groups, on each device that will be part of the mesh group, there must be a mesh group definition for all the peers in the mesh group.
- It is recommended that you use the **connect-source loopback** *num* parameter when issuing the **msdp-peer** command. If you do not use this parameter, the Brocade device uses the outgoing interface's IP address. You must also make sure the IP address of the **connect-source loopback** parameter is the source IP address used by the PIM-RP, and the BGP router.

Common diagnostic scenarios

- High CPU usage with MSDP traffic between two peers.

  This issue can be resolved by resetting both peers.
- PIM SM Multicast packets are not being successfully transmitted.

  A device downstream has multicast passive and PIM snooping enabled. Once PIM snooping is disabled on the downstream device, and added on MBGP links, the problem is resolved.
- Unable to remove an MSDP peer.

  This issue is resolved by upgrading the software version to reflect the latest patches and versions.

PIM DM and PIM SM

Protocol-Independent Multicast (PIM) helps to simplify the complexity of the routing protocol. PIM is similar to DVMRP in that PIM builds source-routed multicast delivery trees and employs reverse path check when forwarding multicast packets.

There are two PIM modes: Dense and Sparse. The Dense Mode (DM) is suitable for densely populated multicast groups, primarily in the LAN environment. The Sparse Mode (SM) is suitable for sparsely populated multicast groups with the focus on WAN.

The Brocade device supports PIM DM V1 and V2. The default is V2. You can specify the version on an individual interface basis.

The primary difference between PIM DM V1 and V2 is the methods the protocols use for messaging:

- PIM DM V1 – Uses the IGMP to send messages.
- PIM DM V2 – Sends messages to the multicast address 224.0.0.13 (ALL-PIM-ROUTERS) with protocol number 103.

The CLI commands for configuring and managing PIM DM are the same for V1 and V2. The only difference is the command you use to enable the protocol on an interface.

NOTE

If you want to continue to use PIM DM V1 on an interface, you must change the version, and then save the configuration. However, this does not mean you can run different PIM versions on devices that are connected to each other. The devices must run the same version of PIM. To connect a Brocade device running PIM to a device that is running PIM V1, you must change the PIM version on the Brocade device to V1 (or change the version on the device to V2, if supported).

PIM DM and PIM SM show commands

The following section describes show commands you can use to display PIM DM and PIM SM information.

show ip pim counter mct

Syntax: show ip pim counter mct

This command displays statistics and error counters for the Multicast MAC Database Update Protocol (MDUP) channel between the Multi-Chassis Trunking (MCT) Peers. Command output resembles the following example.

```
Brocade# show ip pim counter mct
  Multicast MCT Statistics for IPv4 (DN):
  Messages assembled into the send buffer  : 0
  Messages processed out of the recv buffer: 0
  Segments sent successfully to TCP        : 0
  Segments failed to be accepted by TCP    : 0
  Segments assembed into the receive buffer: 0
  Messages dropped because (size > 1500)   : 0
  Messages dropped because it won't fit into available space in send buffer   : 0
  Segments dropped because it won't fit into available space in receive buffer: 0
  Received messages dropped because of cluster-id mismatch                    : 0
  Received messages dropped because the peer was not recognized               : 0
  Received messages dropped because cluster not active                        : 0
  Received messages dropped because MCT VLAN unrecognized                     : 0
  Received messages dropped because of bad message type                       : 0
  Received messages dropped because of bad checksum                           : 0
  Received bytes skipped because of sync or checksum errors                   : 0
```

show ip pim counter nsr

Syntax: show ip pim counter nsr

This command displays Multicast NSR-related status counters from the management processor. Command output resembles the following example.

```
Brocade# show ip pim counter nsr
Mcache sync (entity id: 203)
  pack: 1078
  unpack: 0
  ack: 1078
RPset sync (entity id: 201)
  pack: 3
  unpack: 0
  ack: 3
BSR status (entity id: 202)
  pack: 3
  unpack: 0
  ack: 3
```

show ip pim counter

Syntax: show ip pim counter

This command displays the PIM counters from the line processor, as shown in the following example.

```
Brocade# show ip pim counter
```

```
Forward:
  Packets : 41649 Regsters: 0
Drops:
  RPF-Fail: 0 No-RP   : 0    IfMsmtch: 0
  OIFEmpty: 0 InvldIf : 0    TTLXpire: 0
  NoFwEntr: 0 TrkMove : 0    PortMove: 0
  NoCause : 0 FwEntrFl: 0    ResFail : 0
  SSMNoEnt: 0
IPC:
  MCCreate: 0 MCFirDta: 8000 SGAgeOut: 4000
  Register: 0 WGFirDta: 0    SGAbvThr: 4000
  WrongIf : 0
  NoSGFDta: 0
ISSU:
MVID Save: BufferFull(0); Num Travrse(4000);Num Savd(4000);Num MVID(1)
MVID Restore: Num Travrse(4000);Num Rest(4000);Num MVID(1)
MVID Restore: Invalid MVID(0);InvalidNumOif(0);MVID Resrv Fail(0)
CAM Restore: Total CAMs(4000);Unique CAMs(4000);Inv.VRF(0);
Inv.fwd.entry(0);Dup.CAM(0)
```

show ip pim neighbor

Syntax: show ip pim neighbor

This command displays PIM neighbor states, as shown in the following example.

```
Brocade# show ip pim neighbor
--------+--------+---------------+--------+-----+----------------+-----------+---
Port     PhyPort  Neighbor        Holdtime Age   UpTime           VRF        Prio
                                  sec      sec
--------+--------+---------------+--------+-----+----------------+-----------+---
e1/22    e1/22    10.5.1.1        105      0     00:04:40         default-vrf 1
v46      e3/4     10.5.1.1        105      10    00:04:30         default-vrf 1
v59      e1/1     10.10.10.1      105      0     00:04:40         default-vrf 1
v67      e1/1     10.3.1.1        105      0     00:04:40         default-vrf 1
v420     e1/1     10.100.100.1    105      0     00:04:40         default-vrf 1
```

show ip pim sparse

Syntax: show ip pim sparse

This command displays global PIM SM configuration information, as shown in the following example.

```
Brocade# show ip pim sparse
Global PIM Sparse Mode Settings
  Hello interval        : 30         Neighbor timeout                   : 105
  Bootstrap Msg interval: 60         Candidate-RP Advertisement interval: 60
  Join/Prune interval   : 60         SPT Threshold                      : 1
  Inactivity interval   : 180
  SSM Enabled: Yes
  SSM Group Range: 226.0.0.0/8
---------+----------------+----+---+---------------------+------
Interface|Local           |Mode|Ver|  Designated Router  |TTL
         |Address         |    |   |Address          Port |Thresh
---------+----------------+----+---+---------------------+------
     e6/3 10.2.0.10         SM  V2  Itself                      1
     e6/4 10.3.0.10         SM  V2  Itself                      1
      v30 10.10.10.10       SM  V2  Itself                      1
```

show ip pim group

Syntax: show ip pim group

This command displays PIM group configuration information, as shown in the following example.

```
Brocade# show ip pim group

Total number of Groups: 2
Index 1         Group 239.255.162.1       Ports e3/11
```

show ip pim bsr

Syntax: show ip pim bsr

This command displays PIM bootstrap router (BSR) information, as shown in the following example.

```
Brocade# show ip pim bsr
PIMv2 Bootstrap information
This system is the elected Bootstrap Router (BSR)
  BSR address: 10.95.7.1
  Uptime: 00:33:52, BSR priority: 5, Hash mask length: 32
  Next bootstrap message in 00:00:20

Next Candidate-RP-advertisement in 00:00:10
  RP: 10.95.7.1
    group prefixes:
    224.0.0.0 / 4
Candidate-RP-advertisement period: 60
```

This example shows information for a router that has been elected as the BSR. The following example shows information for a router that is not the BSR. Note that some fields shown in the first example do not appear in the second example.

```
Brocade# show ip pim bsr

PIMv2 Bootstrap information
 BSR address  = 10.95.7.1
 BSR priority = 5
```

show ip pim rp-candidate

Syntax: show ip pim rp-candidate

This command displays candidate RP information, as shown in the following example.

```
Brocade# show ip pim rp-candidate

Next Candidate-RP-advertisement in 00:00:10
  RP: 10.95.7.1
    group prefixes:
    224.0.0.0 / 4

  Candidate-RP-advertisement period: 60
```

This example shows information displayed on a candidate RP router. The following example shows the message displayed for a non-candidate RP router.

```
Brocade# show ip pim rp-candidate
This system is not a Candidate-RP.
```

show ip pim rp-map

Syntax: **show ip pim rp-map**

This command displays RP-to-group mappings, as shown in the following example.

```
Brocade# show ip pim rp-map
Number of group-to-RP mappings: 6

Group address        RP address
-------------------------------
1 239.255.163.1  10.99.99.5
2 239.255.163.2  10.99.99.5
3 239.255.163.3  10.99.99.5
4 239.255.162.1  10.99.99.5
5 239.255.162.2  10.43.43.1
6 239.255.162.3  10.99.99.5
```

show ip pim rp-hash

Syntax: **show ip pim rp-hash** *group-address*

The *group-address* parameter is the address of a PIM Sparse IP multicast group.

This command displays RP hash information for a PIM Sparse group (identified by the *group-address* variable), as shown in the following example.

```
Brocade# show ip pim rp-hash 239.255.162.1

  RP: 10.95.7.1, v2
    Info source: 10.95.7.1, via bootstrap
```

show ip pim rp-set

Syntax: **show ip pim rp-set**

To display the RP set list, enter this command at any CLI level.

```
Brocade# show ip pim rp-set
Group address Static-RP-address Override
---------------------------------------------------
Access-List 44 10.99.99.5 On
Number of group prefixes Learnt from BSR: 1
Group prefix = 239.255.162.0/24 # RPs expected: 1
# RPs received: 1
RP 1: 10.43.43.1 priority=0 age=0
```

show ip pim nbr

Syntax: **show ip pim nbr**

This command displays information about PIM neighbors, as shown in the following example.

```
Brocade# show ip pim nbr

Port   Neighbor        Holdtime   Age     UpTime
                       sec        sec     sec
e3/8   10.95.8.10      180        60      900
Port   Neighbor        Holdtime   Age     UpTime
                       sec        sec     sec
v1     10.95.6.2       180        60      900
```

show ip pim mcache

Syntax: show ip pim mcache *A.B.C.D A.B.C.D*

The *A.B.C.D* variable specifies the IPv4 source or group address.

This command displays information about the IPv4 PIM multicast cache. Command output resembles the following example.

```
Brocade# show ip pim mcache 10.198.198.111 232.26.26.8
IP Multicast Mcache Table
Entry Flags    : SM  - Sparse Mode, SSM - Source Specific Mutlicast, DM - Dense
Mode
                RPT    - RPT Bit, SPT - SPT Bit, LSRC - Local Source, LRCV - Local
Receiver, JOIN - Join Upstream
                HW - HW Forwarding Enabled, FAST - Resource Allocated, TAG - Need
For Replication Entry
                 REGPROB - Register In Progress, REGSUPP - Register Suppression
Timer
                 MSDPADV - Advertise MSDP, NEEDRTE - Route Required for Src/RP,
PRUN - DM Prune Upstream
Interface Flags: IM - Immediate, IH - Inherited, WA - Won Assert
                 MJ - Membership Join, MI - Membership Include, ME - Membership
Exclude
                 BR - Blocked RPT, BA - Blocked Assert, BF - Blocked Filter, BI -
Blocked IIF,
            BM - Blocked MCT
Total entries in mcache: 1

1     (10.2.2.101, 239.0.1.3) in v200 (e2/15), Uptime 00:01:08, Rate 42229 (SM)
      Source is directly connected. RP 10.2.2.1
      Flags (0x3046cec1) SM SPT L2REG LSRC LRCV JOIN HW FAST MSDPADV
      fast ports: ethe 2/1
      AgeSltMsk: 00000002,  FID: 0x8006, MVID: NotReq, RegPkt: 0, AvgRate: 41688,
profile: none
      Forwarding_oif: 1, Immediate_oif: 1, Blocked_oif: 1
      L2 (HW) 1:
          TR(e2/1,e2/1),  00:01:08/181, Flags: IM IH
      Blocked OIF 1:
          TR(e1/5,e1/5)(VL200), 00:01:08/0, Flags: MJ BM
Number of matching entries: 1
```

show ipv6 pim mcache

Syntax: show ipv6 pim mcache

This command displays information about the IPv6 PIM multicast cache. Command output resembles the following example.

```
Brocade# show ipv6 pim mcache
IP Multicast Mcache Table
```

```
Entry Flags    : SM  - Sparse Mode, SSM - Source Specific Mutlicast, DM - Dense
Mode
                RPT    - RPT Bit, SPT - SPT Bit, LSRC - Local Source, LRCV - Local
Receiver, JOIN - Join Upstream
                HW - HW Forwarding Enabled, FAST - Resource Allocated, TAG - Need
For Replication Entry
                 REGPROB - Register In Progress, REGSUPP - Register Suppression
Timer
                 MSDPADV - Advertise MSDP, NEEDRTE - Route Required for Src/RP,
PRUN - DM Prune Upstream
Interface Flags: IM - Immediate, IH - Inherited, WA - Won Assert
                 MJ - Membership Join, MI - Membership Include, ME - Membership
Exclude
                BR - Blocked RPT, BA - Blocked Assert, BF - Blocked Filter, BI -
Blocked IIF,
            BM - Blocked MCT
Total entries in mcache: 1
1 (2001:DB8:62:62::11, 2001:DB8::2) in v62 (tag e1/1), Uptime 00:00:58
(SM)      upstream neighbor is L2 fe80::21b:edff:fea4:a441. RP
2001:DB8:2:3:4::b      Flags (0x304680c1) SM SPT LSRC LRCV HW FAST      fast
ports:       AgeSltMsk: 00000003, FID: 0xffff (D), DIT: NotReq, profile: none,
KAT Timer value: 240      Forwarding_oif: 0, Immediate_oif: 0, Blocked_oif:
1       Blocked OIF 1:
             TR(e1/39,e1/39)(VL62), 00:00:01/0, Flags: MJ BM

Number of matching entries: 1
```

show ip pim traffic

Syntax: show ip pim traffic

This command displays PIM traffic statistics, as shown in the following example.

```
Brocade# show ip pim traffic

Port    Hello          J/P           Register       RegStop          Assert
     [Rx     Tx]    [Rx     Tx]    [Rx     Tx]    [Rx     Tx]    [Rx     Tx]
e3/8   19     19      32     0       0      0       37     0       0      0
v1     18     19      0      20      0      0       0      0       0      0
v2     0      19      0      0       0      16      0      0       0      0

Total 37      57     32      0       0      0       0      0       0      0
IGMP Statistics:
  Total Recv/Xmit 85/110
  Total Discard/chksum  0/0
```

NOTE
If you have configured interfaces for standard PIM (Dense Mode) on your router, statistics for these interfaces are listed first in the display.

Clearing the PIM forwarding cache

You can clear the PIM forwarding cache by using the following command.

clear pim-cache

Syntax: clear pim-cache

PIM debug commands

The following section describes the debug commands used to display PIM DM and PIM SM related debug information. For debug commands for PIM-DVMRP, refer to "DVMRP debug commands" on page 340.

debug ip pim oif

Syntax: [**no**] **debug** [**ip** | **ipv6**] **pim oif** [**add-del** | **fsm** | **timer**]

- **add-del** - Displays debugs related to the addition and deletion of outbound interfaces (OIFs) for a forwarding entry.
- **fsm** - Displays debugs related to OIF FSM related information.
- **timer** - Displays debugs related to the periodic OIF timers.

The following is the sample output from the **debug ip pim oif** command.

```
Brocade# debug ip pim oif
        PIM:  OIF FSM debugging is on
        PIM:  OIF timer debugging is on
        PIM:  OIF External add-del debugging is on
```

debug ip pim bootstrap

Syntax: [**no**] **debug** [**ip** | **ipv6**] **pim bootstrap**

This command displays bootstrap messages in detail as shown in the following example.

```
Brocade# debug ip pim bootstrap
PIM:  bootstrap debugging is on
Brocade# May 14 14:05:21.640 PIM-BSR.VRF0: Prefer BSR 10.11.11.11(Pr 1) over
current BSR 10.11.11.11(Pr 1)
May 14 14:05:21.640 PIM-BSR.VRF0: Intf 2/4 - accept BSM from BSR 10.11.11.11(Pr
1), local state AccPref, curr BSR 10.11.11.11 (Pr 1)
May 14 14:05:21.640 PIMv4-BSR.VRF0: Intf 2/16 - Ignoring BSRMsg from 10.11.11.11
due to RPF-failure. RPFIfId e2/4
May 14 14:05:21.640 PIMv4-BSR.VRF0: Intf tn2 - Ignoring BSRMsg from 10.11.11.11
due to RPF-failure. RPFIfId e2/4
May 14 14:05:21.641 PIMv4-BSR.VRF0: Intf 2/12 - Ignoring BSRMsg from 10.11.11.11
due to RPF-failure. RPFIfId e2/4
```

debug ip pim clear

Syntax: [**no**] **debug** [**ip** | **ipv6**] **pim clear**

This command clears all PIM related debug settings.

debug ip pim entry

Syntax: [**no**] **debug** [**ip** | **ipv6**] **pim entry** [**add-del** | **hw** | **kat**]

- **add-del** - Displays debugs related to the addition and deletion of software forwarding entries.
- **hw** - Displays debugs related to programming of forwarding entries in harwdare.
- **kat** - Displays debugs related to programming of the KAT timer that is used to age out forwarding entries (for PIMSM only).

The following is the sample output from the **debug ip pim entry** command.

```
Brocade# debug ip pim entry
        PIM:  KAT timer debugging is on
        PIM:  Entry Hw programming debugging is on
        PIM:  Entry creation/deletion debugging is on
```

debug ip pim event

Syntax: [**no**] **debug** [**ip** | **ipv6**] **pim event**

This command displays debugging information about infrastructure events and callback handling. The following is the sample output from the **debug ip pim event** command.

```
Brocade# debug ip pim event
Aug 17 03:53:42.646 MCAST_MCT: mcast_mct_send_message_to_peer: msg_type 2,
msg_len 32, client 300, vlan 200
Aug 17 03:53:42.653 MCAST_MCT: mcast_mct_send_message_on_mdup: app_id 64,
seg_size 44, seg_base 0 returned 0
Aug 17 03:53:48.149 MCAST_MCT: mcast_mct_send_message_to_peer: msg_type 2,
msg_len 32, client 300, vlan 200
Aug 17 03:53:48.153 MCAST_MCT: mcast_mct_send_message_on_mdup: app_id 64,
seg_size 44, seg_base 0 returned 0
```

debug ip pim filter

Syntax: [**no**] **debug** [**ip** | **ipv6**] **pim filter** [**group-prefix** | **level** | **source-prefix** | **stack** | **vrf-index**]

- **group-prefix** - Filters and displays debugs relevant to groups that are within the group prefix range.
- **level** - Displays additional information including data structure logs.
- **source-prefix** - Filters and displays debugs relevant to sources that are within the source prefix range.
- **stack** - Displays function call stack information along with other debugging information.
- **vrf-index** - Filters and displays debugs relevant to the particular VRF index.

The following is the sample output from the **debug ip pim filter group-prefix** command.

```
Brocade# debug ip pim filter group-prefix 225.0.0.1/32
        PIM:  Filter Group prefix debugging is on, value = 225.0.0.1
```

The following is the sample output from the **debug ip pim filter level** command.

```
Brocade# debug ip pim filter level 2
        PIM:  Debug Level debugging is on, value = 2
```

debug ip pim ipc

Syntax: [**no**] **debug** [**ip** | **ipv6**] **pim ipc**

This command displays information about the IPC messages that are exchanged between the MP and the LP.

```
Brocade# debug ip pim ipc
PIM:  ipc debugging is on
```

debug ip pim join-prune

Syntax: [**no**] **debug** [**ip** | **ipv6**] **pim join-prune**

This command displays information about the PIM joins and prunes that received and sent by the router.

```
Brocade# debug ip pim join-prune
PIM:  join-prune debugging is on

Brocade# May 14 14:15:16.117 PIMSM.VRF0: BEGIN J/P proc: rpf_nbr 10.1.1.4, to_me=1
from 10.1.1.2, intf tn2 ----
May 14 14:15:16.117   Group=225.0.0.1. Join list: 1 srcs
May 14 14:15:16.117     J-Src=10.15.15.100, wc=0, rpt=0, SM=1
May 14 14:15:16.117      PIM.VRF0: Processing (S,G) Join (10.15.15.100 225.0.0.1)
from 10.1.1.2, intf tn2,2/16
May 14 14:15:16.117   Group=225.0.0.1. Prune list: 0 srcs
May 14 14:15:16.117   Group=225.0.0.2. Join list: 1 srcs
May 14 14:15:16.117     J-Src=10.15.15.100, wc=0, rpt=0, SM=1
May 14 14:15:16.117      PIM.VRF0: Processing (S,G) Join (10.15.15.100 225.0.0.2)
from 10.1.1.2, intf tn2,2/16
May 14 14:15:16.117   Group=225.0.0.2. Prune list: 0 srcs
May 14 14:15:16.117   Group=225.0.0.3. Join list: 1 srcs
May 14 14:15:16.117     J-Src=10.15.15.100, wc=0, rpt=0, SM=1
May 14 14:15:16.117      PIM.VRF0: Processing (S,G) Join (10.15.15.100 225.0.0.3)
from 10.1.1.2, intf tn2,2/16
May 14 14:15:16.117   Group=225.0.0.3. Prune list: 0 srcs
May 14 14:15:16.117   Group=225.0.0.4. Join list: 1 srcs
May 14 14:15:16.117     J-Src=10.15.15.100, wc=0, rpt=0, SM=1
May 14 14:15:16.117      PIM.VRF0: Processing (S,G) Join (10.15.15.100 225.0.0.4)
from 10.1.1.2, intf tn2,2/16
May 14 14:15:16.117   Group=225.0.0.4. Prune list: 0 srcs
May 14 14:15:16.117 PIMSM: END J/P proc. ---------------------
May 14 14:15:16.634 PIMSM.VRF0: BEGIN J/P proc: rpf_nbr 192.168.4.4, to_me=1 from
192.168.4.1, intf e2/4
```

debug ip pim nbr-change

Syntax: [**no**] **debug** [**ip** | **ipv6**] **pim nbr-change**

This command displays information about preventing PIM IPv4 neighbors on Cluster Client Edge Ports (CCEPs). Command output resembles the following example.

```
Brocade# debug ip pim nbr-change
PIM:  nbr-change  debugging is on

Apr 9 17:20:02.668 PIM.VRF0: Rx Hello msg from 10.2.2.101 on intf v62, 1/1Apr 9
17:20:02.938 PIM.VRF0: Rx Hello msg from 10.2.2.103 on intf v62, 1/3Apr 9
17:20:02.938 PIM.VRF0: Rx Hello msg from 10.2.2.103 on intf v62, 1/3 is dropped:
Neighbors not allowed on CCEPs
```

debug ip pim optimization

Syntax: [**no**] **debug** [**ip** | **ipv6**] **pim optimization**

This command displays information about the hardware forwarding resources used by all multicast flows in the system.

```
Brocade# debug ip pim optimization
PIM: optimization debugging is on
```

debug ip pim rate-update

Syntax: [**no**] **debug** [**ip** | **ipv6**] **pim rate-update**

This command displays the IPC messages exchanged between the LP and the MP to communicate rates of PIM streams.

```
Brocade# debug ip pim rate-update
PIM-INFO(HW rate update).VRF 0: (10.1.1.1, 225.0.0.1) Total pkts: 100
```

debug ip pim reg-proc

Syntax: [no] debug [ip | ipv6] pim reg-proc

This command displays information about the sending and processing of PIM register messages at the first hop and the Rendezvouz Point (RP). Also displays information about the periodic null-registers.

```
Brocade# debug ip pim reg-proc
PIM:  regproc debugging is on

RegSupp expires. RegSupp 1 RegProbe 0, send null reg to RP: 10.11.11.11
May 14 14:21:42.076 PIMSM-REG.VRF0: (10.15.15.100 225.0.0.2) RegSupp expires.
RegSupp 1 RegProbe 0, send null reg to RP: 10.11.11.11
May 14 14:21:42.076 PIMSM-REG.VRF0: (10.15.15.100 225.0.0.3) RegSupp expires.
RegSupp 1 RegProbe 0, send null reg to RP: 10.11.11.11
May 14 14:21:42.076 PIMSM-REG.VRF0: (10.15.15.100 225.0.0.4) RegSupp expires.
RegSupp 1 RegProbe 0, send null reg to RP: 10.11.11.11
May 14 14:21:42.078 PIM.0: rcvd reg stop (10.15.15.100 225.0.0.1) from 10.11.11.11
May 14 14:21:42.078 PIM vrf 0 rx RegStop. Reg Suppr Timer restarted for
(10.15.15.100 225.0.0.1)
May 14 14:21:42.078 PIM.0: rcvd reg stop (10.15.15.100 225.0.0.2) from 10.11.11.11
May 14 14:21:42.078 PIM vrf 0 rx RegStop. Reg Suppr Timer restarted for
(10.15.15.100 225.0.0.2)
May 14 14:21:42.078 PIM.0: rcvd reg stop (10.15.15.100 225.0.0.3) from 10.11.11.11
May 14 14:21:42.078 PIM vrf 0 rx RegStop. Reg Suppr Timer restarted for
(10.15.15.100 225.0.0.3)
May 14 14:21:42.078 PIM.0: rcvd reg stop (10.15.15.100 225.0.0.4) from 10.11.11.11
May 14 14:21:42.078 PIM vrf 0 rx RegStop. Reg Suppr Timer
```

debug ip pim route-change

Syntax: [no] debug [ip | ipv6] pim route-change

This command displays information about route change events.

```
Brocade# debug ip pim route-change
PIM:  route-change debugging is on
Brocade# May 14 14:24:41.077 PIM-RtChg-VRF0: RPF Lookup for Dest:10.11.11.11 safi
URTM, Mcast ECMP Paths 1
```

debug ip pim rp

Syntax: [no] debug [ip | ipv6] pim rp

This command displays information about the impact on the multicast flows when the RP address changes for a particular group prefix.

```
Brocade# debug ip pim rp
PIM:  rp  debugging is on
```

debug ip pim show

Syntax: [no] debug [ip | ipv6] pim show

This command displays PIM related debug settings.

```
Brocade# debug ip pim show
PIM debugging is enabled
PIM:  OIF FSM debugging is on
PIM:  OIF timer debugging is on
PIM:  bootstrap debugging is on
PIM:  rp  debugging is on
PIM:  KAT timer debugging is on
PIM:  Entry Hw programming debugging is on
PIM:  Entry creation/deletion debugging is on
```

debug ip pim sync-lib

Syntax: [**no**] **debug** [**ip** | **ipv6**] **pim sync-lib**

This command displays information about the impact that hitless upgrade and MP switchover have on multicast flows.

```
Brocade# debug ip pim sync-lib
PIM:  sync-lib debugging is on
```

debug ip pim timer-type

Syntax: [**no**] **debug** [**ip** | **ipv6**] **pim timer-type** *decimal*

The *decimal* variable specifies the type of timer to be enabled.

This command displays information regarding the events associated with the multicast protocol and hardware timers in the system. Note that only one of the timer types can be enabled at the same time.

```
Brocade# debug ip pim timer-type 5
        PIM:  timer-type PIMTMR_HELLO debugging is on
```

Configuration notes

The following limitations apply to implementation of PIM SM:

- PIM Border Routers (PMBRs) are not supported. You cannot configure a routing interface as a PMBR interface for PIM SM.
- PIM SM and regular PIM (Dense Mode) cannot be used on the same interface.
- You cannot configure or display PIM SM information using the Web management interface. (You can display some general PIM information, but not specific PIM SM information.)
- If you want to continue to use PIM DM V1 on an interface, you must change the version, and then save the configuration. This does not mean you can run different PIM versions on devices that are connected to each other. The devices must run the same version of PIM. If you want to connect a device running PIM to a device that is running PIM V1, you must change the PIM version on the device to V1 (or change the version on the device to V2, if supported).
- When PIM routing is enabled, the line rate for receive traffic is reduced by about 5 percent. The reduction occurs due to overhead from the VLAN multicasting feature, which PIM routing uses. This behavior is normal and does not indicate a problem with the device.
- You do not need to enable IP multicast routing globally when configuring PIM SM.
- It is recommended that you configure the same device as both the BSR and the RP.

- It is possible to configure the device as only a candidate BSR or RP, but it is recommended that you configure the same interface on the same device as both a BSR and an RP.
- Specify the same IP address as the RP on all PIM SM routers within the PIM SM domain. Make sure the router is on the backbone or is otherwise well connected to the rest of the network.
- Use the same join or prune message interval on all the PIM SM routers in the PIM SM domain. If the routers do not all use the same timer interval, the performance of PIM SM can be adversely affected.

Common diagnostic scenarios

- The following PIM SM error appears:

  ```
  ERROR - cannot enable PIM sparse or DVMRP on an interface.
  ```
  This issue is resolved by upgrading the software version to reflect the latest patches and versions.
- Specific multicast sources are not being forwarded properly.

  The dmzo router was directed to an incorrect rpf. After modifying the configuration, the problem is resolved.
- Enabling IP PIM-SPARSE caused high CPU usage.

  This can occur when the device runs out of CAM due to large amounts of multicast traffic flooding the CAM. Make sure the device is using the most current patch of the software, and contact Brocade Technical Support for assistance.

MLD

Multicast Listener Discovery (MLD) discovers multicast listeners, or nodes that want to receive multicast packets on directly attached links. MLDv2 supports source filtering, the ability of a node to send reports on traffic that is from a specific address source or from all multicast addresses except the specified address sources. The information is then provided to the source-specific multicast (SSM) routing protocols such as PIM-SSM.

MLD show commands

The following section describes show commands you can use to display MLD information.

show ipv6 mld interface

Syntax: show ipv6 mld interface

This command displays the MLD definitions for an interface. Command output resembles the following example.

```
Brocade# show ipv6 mld interface
---------+------+---------+---------------------------------------+---------+----
-+---------
Intf/Port|Groups| Version |Querier                                |  Timer
|V1Rtr|Tracking
        |      |Oper  Cfg|                                 |OQrr GenQ|     |
---------+------+----+----+---------------------------------------+----+----+----
-+---------
```

```
v62             0    2    2
Disabled
     e2/1            2    - Self (MCT-Blk)                                    0   79 No
    e1/37            2    - Self                                              0  108 No
    e1/33            2    - Self                                              0  108 No
```

MLD debug commands

This section describes the debug commands that generate MLD information.

debug ipv6 mld protocol query

Syntax: debug ipv6 mld protocol query

This command displays information about MLD query suppression state on the CCEPs. Command output resembles the following example.

```
Brocade# debug ipv6 mld protocol query
Apr  9 18:11:56.102 MLD.VRF0: [ Port 1/37,v62 ] Sent General Query version 2 using
src fe80::211:ff:fe04:1001
Apr  9 18:11:56.102 MLD.VRF0: [ Port 1/33,v62 ] Sent General Query version 2 using
src fe80::211:ff:fe04:1001
Apr  9 18:11:27.103 MLD.VRF0: [ Port 2/1,v62 ] Skipped General Query on CCEP port
```

Chapter

Security Diagnostics 9

In this chapter

This chapter describes diagnostic information for security environments on Brocade NetIron XMR series and Brocade MLX series routers.

802.1x

802.1x port security allows you to configure a Brocade device to grant access to a port based on information supplied by a client to an authentication server.

This section describes how to use show commands and debug commands to monitor 802.1x configurations and activity on Brocade NetIron XMR series and Brocade MLX series routers.

CAUTION

Enabling diagnostic commands may degrade system performance. These commands are best used to troubleshoot specific problems while working with qualified Brocade service technicians. Whenever possible, troubleshoot your system during periods of low network traffic and user activity to preserve system performance.

NOTE
To save space, date and time stamps have been removed from all output examples.

802.1x show commands

You can display the following 802.1x-related information:

- Information about the 802.1x configuration on the device and on individual ports
- Statistics about the EAPOL frames passing through the device
- Information about 802.1x-enabled ports dynamically assigned to a VLAN
- Information about the user-defined and dynamically applied MAC address and IP ACLs currently active on the device
- Information about the 802.1x multiple client configuration

show dot1x

Syntax: **show dot1x**

This command displays information about the 802.1x configuration, as shown in the following example.

```
Brocade# show dot1x
PAE Capability                : Authenticator Only
system-auth-control           : Enable
Number of ports enabled       : 25
re-authentication             : Disable
global-filter-strict-security: Enable
quiet-period                  : 60 Seconds
tx-period                     : 30 Seconds
supptimeout                   : 30 Seconds
servertimeout                 : 30 Seconds
maxreq                        : 3
re-authperiod                 : 3600 Seconds
Protocol Version              : 1
auth-fail-action              : Block Traffic
MAC Session Aging             : All
MAC Session Max Age           : 120 Seconds
Maximum Failed Attempts       : 3
```

show dot1x configuration

Syntax: **show dot1x configuration** [**all** | **ethernet** *slotnum/portnum*]

This command displays information about the 802.1x configuration on all ports, or on a specified port. The following example shows the output from this command when issued for a specific port.

```
Brocade# show dot1x configuration ethernet 1/3

Port 1/3 Configuration:
AuthControlledPortControl    : Auto
max-clients                  : 32
multiple-clients             : Enable
filter-strict-security       : Enable
```

show dot1x statistics

Syntax: **show dot1x statistics** [**all** | **ethernet** *slotnum/portnum*]

This command displays 802.1x statistics for all ports, or for a specified port. The following example shows the output from this command when issued for a specific port.

```
Brocade# show dot1x statistics ethernet 3/3
Port 1/3 Statistics:
RX EAPOL Start:                0
RX EAPOL Logoff:               0
RX EAPOL Invalid:              0
RX EAPOL Total:                2
RX EAP Resp/Id:                1
RX EAP Resp other than Resp/Id: 1
RX EAP Length Error:           0
Last EAPOL Version:            1
Last EAPOL Source:             0000.000b.8bef
TX EAPOL Total:                3
TX EAP Req/Id:                 1
TX EAP Req other than Req/Id:  1
Num Sessions:                  1
Num Restricted Sessions:       0
Num Authorized Sessions:       1
```

show interfaces

Syntax: **show interfaces ethernet** *slotnum/portnum*

The **show interface** command displays the VLAN to which an 802.1x-enabled port has been dynamically assigned, as well as the port from which it was moved (that is, the port's default VLAN).

The following example indicates the port's dynamically assigned VLAN. Information about the dynamically assigned VLAN is shown in bold type. In this example, the 802.1x-enabled port has been moved from VLAN 1 to VLAN 4094. When the client disconnects, the port will be moved back to VLAN 1.

```
Brocade# show interfaces ethernet 12/2
GigabitEthernet1/3 is up, line protocol is up
  Hardware is GigabitEthernet, address is 0000.00e2.5800 (bia 0000.00e2.5800)
  Configured speed auto, actual 100Mbit, configured duplex fdx, actual fdx
  Configured mdi mode AUTO, actual MDIX
  Member of L2 VLAN ID 4094 (dot1x-RADIUS assigned), original L2 VLAN ID is 1,
  port is untagged, port state is Forwarding
  STP configured to ON, Priority is level0, flow control enabled
  Force-DSCP disabled
  mirror disabled, monitor disabled
  Not member of any active trunks
  Not member of any configured trunks
  No port name
  Internet address is 10.12.12.250/24, MTU 1522 bytes, encapsulation ethernet
  300 second input rate: 810 bits/sec, 0 packets/sec, 0.00% utilization
  300 second output rate: 1253 bits/sec, 1 packets/sec, 0.00% utilization
  70178 packets input, 7148796 bytes, 0 no buffer
  Received 0 broadcasts, 0 multicasts, 70178 unicasts
  0 input errors, 0 CRC, 0 frame, 0 ignored
  0 runts, 0 giants, DMA received 70178 packets
  91892 packets output, 10081165 bytes, 0 underruns
  Transmitted 9853 broadcasts, 13330 multicasts, 68709 unicasts
  0 output errors, 0 collisions, DMA transmitted 91892 packets
```

show dot1x mac-address-filter

Syntax: **show dot1x mac-address-filter** [**all** | **ethernet** *slotnum/portnum*]

- The **all** keyword displays all dynamically applied MAC address filters active on the device.
- Use the **ethernet** *slotnum/portnum* parameter to display information for one port.

This command displays information about MAC filters. If you specify a specific port, and if the MAC address filter is dynamically assigned by 802.1x, the command output resembles the following example.

```
Brocade# show dot1x mac-address-filter ethernet 1/1
Port 1/1 MAC Address Filter information:
  802.1x dynamic MAC Filter (user defined) :
    mac access-list 401 in
  Port default MAC Filter :
    mac access-list 400 in
```

"Port default MAC Filter" appears if a default MAC filter has been configured on the port. This is the filter that will be applied to the port once the dynamically assigned MAC filter is removed. If a default MAC filter has not been configured, the message "No Port default MAC" is displayed.

When the dynamically assigned MAC address filter is removed, output resembles the following example.

```
Brocade# show dot1x mac-address-filter ethernet 1/1
Port 1/1 MAC Address Filter information:
  Port default MAC Filter :
    mac access-list 400 in
```

show dot1x ip-acl

Syntax: show dot1x ip-acl [**all** | **ethernet** *slotnum/portnum*]

- **all** - Displays all dynamically applied IP ACLs active on the device.
- **ethernet** *slotnum/portnum* - Displays information for one port.

This command displays information about what IP ACLs have been applied to an 802.1x-enabled port. If an IP ACL was dynamically applied by 802.1x, output from this command resembles the following example.

```
Brocade# show dot1x ip-acl ethernet 1/1
Port 1/1 IP ACL information:
  802.1x dynamic IP ACL (user defined) in:
    ip access-list extended Port_1/1_E_IN in
  Port default IP ACL in:
    ip access-list 100 in
  No outbound ip access-list is set
```

"Port default IP ACL" appears if a default IP ACL has been configured on the port. This is the IP ACL that will be applied to the port once the dynamically assigned IP ACL is removed. If a default IP ACL has not been configured, the message "No Port default IP ACL" is displayed.

When the dynamically assigned IP ACL is removed from the port, the command output shows the following information.

```
Brocade# show dot1x ip-acl ethernet 1/1
Port 1/1 IP ACL information:
  Port default IP ACL in:
    ip access-list 100 in
  No outbound ip access-list is set
```

show dot1x mac-sessions

Syntax: show dot1x mac-sessions

This command displays information about the 802.1x MAC sessions on all ports, as shown in the following example.

```
Brocade# show dot1x mac-sessions
Port  MAC             Username                      VLAN Auth State ACL|MAC  Age
                                                                    i|o|f
--------------------------------------------------------------------------------
1/1   0000.000b.8cd7  Mary M                           1    DENIED  n|n|n   0
1/2   0000.000b.8cb3  adminmorn                     4094 PERMITTED  y|n|n   0
1/3   0000.000b.8bef  reports                       4094 PERMITTED  y|n|n   0
1/4   0000.001f.6a63  testgroup                     4094 PERMITTED  y|n|n   0
1/5   0000.001a.ff7e  admineve                      4094 PERMITTED  y|n|n   0
```

show dot1x mac-sessions brief

Syntax: show dot1x mac-sessions brief

This command displays information about the 802.1x MAC sessions in brief, as shown in the following example.

```
Brocade# show dot1x mac-sessions brief
Port             Number of users      Dynamic Dynamic     Dynamic
          Restricted Authorized Total VLAN    ACL (In/Out)MAC-Filt
---------+----------+----------+-----+-------+-----------+--------
1/1              0          0     1 no          no/no   no
1/2              0          1     1 yes        yes/no   no
1/3              0          1     1 yes        yes/no   no
1/4              0          1     1 yes        yes/no   no
1/5              0          1     1 yes        yes/no   no
```

Clearing 802.1x statistics

You can clear the 802.1x statistics counters on all interfaces at once, on individual interfaces, or on a range of interfaces.

To clear the 802.1x statistics counters on all interfaces on the device, enter the following command.

clear dot1x statistics all

Syntax: clear dot1x statistics all

Use the following command to clear 802.1x statistics for a specific Ethernet port.

clear dot1x statistics ethernet

Syntax: clear dot1x statistics ethernet *slotnum/portnum*

802.1x debug commands

This section describes the 802.1x-related debug commands.

debug dot1x

Syntax: [no] **debug dot1x** [**all** | **dumpclass** | **events** | **fault** | **packets** | **port** | **state** | **timers**]

This command displays information about 802.1x authentication events, activity, and settings.

- **all** - Displays general information about 802.1x activity for all ports.
- **dumpclass** - Displays the internal data structure.
- **events** - Displays significant events for all ports.
- **fault** - Displays any internal errors for all ports.
- **packets** - Displays information about 802.1x packets.
- **port** - Displays information about 802.1x events and timers for specified ports.
- **state** - Displays 802.1x port state information.
- **timers** - Displays 802.1x timer information.

debug dot1x all

Syntax: [**no**] **debug dot1x all**

This command provides a complete profile of the authentication process, including events, faults, timers, and packets. This command works globally across all ports. Command output resembles the following example.

```
Brocade# debug dot1x all
802.1X All debugging ON
Dec 21 17:11:48 : 802.1X: Timer tick expired
: 802.1X 3/16 Tx EAPOL on vlan_id 1 for dst 0000.0000.0003, len 8:
EAP PACKET - EAPCode: FAILURE
dump TX 802.1X: 8 bytes|
01000004 04000004                      ........
: 802.1X: port 3/16 Tx (OK) EAPOL-FAIL Pkt (EAPId: 0)
: 802.1X: Port 3/16 txEAP timer expired. Transmitting an EAP ReqId
: 802.1X 3/16 Tx EAPOL on vlan_id 1 for dst 0000.0000.0003, len 9:
EAP PACKET - EAPCode: REQUEST  EAPType: IDENTITY
dump TX 802.1X: 9 bytes
01000005 01000005 010000fc             .........
: 802.1X: port 3/16 Tx (OK) EAPOL-EAP-REQUEST-ID Pkt (EAPId: 0)
: 802.1X: Timer tick expired
: 802.1X: Timer tick expired
: 802.1X: Timer tick expired
: 802.1X: Timer tick expired
: 802.1X: port 4/1 Rx EAPOL Pkt SA=0000.00aa.0001, DA=0000.0000.0003, len=0
EAP START   Dec 21 17:12:02   dump RX 802.1X: 18 bytes
0180c200 00030000 00aa0001 888e0101    ................
000071ae                               ..
: 802.1X: port 4/1 Rx EAPOL_START
: 802.1X: port 4/1 BkEnd BKEND_INVALID --> BKEND_INIT
: 802.1X: port 4/1 BkEnd BKEND_INIT --> BKEND_IDLE
: 802.1X: port 4/1 AuthPAE AUTH_INVALID --> AUTH_INIT
: 802.1X: port 4/1 AuthPAE AUTH_INIT --> AUTH_DISCONCTED
: 802.1X: port 4/1 AuthPAE AUTH_DISCONCTED --> AUTH_CONNTING
: 802.1X 4/1 Tx EAPOL on vlan_id 1 for dst 0000.00aa.0001, len 9:
EAP PACKET - EAPCode: REQUEST  EAPType: IDENTITY
dump TX 802.1X: 9 bytes
01000005 01010005 01011607             .........
: 802.1X: port 4/1 Tx (OK) EAPOL-EAP-REQUEST-ID Pkt (EAPId: 1)
```

```
: 802.1X 4/1 Tx EAPOL on vlan_id 1 for dst 0000.00aa.0001, len 9:
EAP PACKET - EAPCode: REQUEST  EAPType: IDENTITY
dump TX 802.1X: 9 bytes
01000005 01010005 01011607             .........
: 802.1X: port 4/1 Tx (OK) EAPOL-EAP-REQUEST-ID Pkt (EAPId: 1)
: 802.1X: port 4/1 Rx EAPOL Pkt SA=0000.00aa.0001, DA=0000.0000.0003, len=6
EAP PACKET - EAPCode: RESPONSE  EAPType: IDENTITY
dump RX 802.1X: 24 bytes
0180c200 00030000 00aa0001 888e0100    ................
00060201 00060161                      .......a
: 802.1X: port 4/1 Rx EAPOL-EAP-RESPONSE-ID Pkt (EAPId: 1)
: 802.1X: port 4/1 AuthPAE AUTH_CONNTING --> AUTH_ENTCATING
: 802.1X: port 4/1 BkEnd BKEND_IDLE --> BKEND_RESPONSE
: 802.1X: port 4/1 aWhile timer (AuthServer) started for 30 secs
: 802.1X: port 4/1 Tx EAP PDU (EAPId: 1) to AuthServer
: 802.1X: port 4/1 Rx EAPOL Pkt SA=0000.00aa.0001, DA=0000.0000.0003, len=6
EAP PACKET - EAPCode: RESPONSE  EAPType: IDENTITY
dump RX 802.1X: 24 bytes
0180c200 00030000 00aa0001 888e0100    ................
00060201 00060161                      .......a
: 802.1X: port 4/1 Rx EAPOL-EAP-RESPONSE-ID Pkt (EAPId: 1)
: 802.1X: port 4/1 Rx AAA_INTERACTIVE from AuthServer
: 802.1X: port 4/1 Rx from Server: EAP-REQUEST-MD5 (EAPId: 2) Len 22
: 802.1X 4/1 Tx EAPOL on vlan_id 1 for dst 0000.00aa.0001, len 26:
EAP PACKET - EAPCode: REQUEST  EAPType: MD5
dump TX 802.1X: 26 bytes
01000016 01020016 04102fb9 32b79134    ........../.2..4
17b5ea71 89b9eb79 b42b0b04             ...q...y.+
: 802.1X: port 4/1 BkEnd BKEND_RESPONSE --> BKEND_REQUEST
: 802.1X: port 4/1 Rx AAA_INTERACTIVE from AuthServer
: 802.1X: port 4/1 Rx from Server: EAP-REQUEST-MD5 (EAPId: 2) Len 22
: 802.1X 4/1 Tx EAPOL on vlan_id 1 for dst 0000.00aa.0001, len 26:
EAP PACKET - EAPCode: REQUEST  EAPType: MD5
dump TX 802.1X: 26 bytes
01000016 01020016 04102fb9 32b79134    ........../.2..4
17b5ea71 89b9eb79 b42b0b04             ...q...y.+
: 802.1X: port 4/1 BkEnd BKEND_RESPONSE --> BKEND_REQUEST
: 802.1X: port 4/1 Fwd (OK) EAPOL-EAP Pkt (EAPId: 2) from AuthServer to Supplicant
: 802.1X: port 4/1 aWhile timer (Supplicant) started for 30 secs
: 802.1X: port 4/1 Rx EAPOL Pkt SA=0000.00aa.0001, DA=0000.0000.0003, len=24
EAP PACKET - EAPCode: RESPONSE  EAPType: MD5
dump RX 802.1X: 42 bytes
0180c200 00030000 00aa0001 888e0100    ................
00180202 00180410 0462366a 6cb18e4a    .........b6jl..J
e739af16 80a64756 61000000             .9....GVa.
: 802.1X: port 4/1 Rx EAP-RESPONSE-MD5 Pkt (EAPId: 2) Len 24
: 802.1X: port 4/1 BkEnd BKEND_REQUEST --> BKEND_RESPONSE
: 802.1X: port 4/1 aWhile timer (AuthServer) started for 30 secs
: 802.1X: port 4/1 Tx EAP PDU (EAPId: 2) to AuthServer
: 802.1X: port 4/1 Rx AAA_ACCEPT from AuthServer
: 802.1X: Port 4/1 Created user-defined mac filter 400.
: 802.1X: Port 4/1 Binding with the RADIUS assigned MAC ACL ID: 400
: 802.1X: port 4/1 Rx Tunnel Data (Type=0, Medium_Type=0, PvtGrpId=NULL)
: 802.1X 4/1 Tx EAPOL on vlan_id 1 for dst 0000.00aa.0001, len 8:
EAP PACKET - EAPCode: SUCCESS
dump TX 802.1X: 8 bytes
01000004 03020004                      ........
: 802.1X: port 4/1 Tx (OK) EAPOL-SUCCESS Pkt (EAPId: 2)
: 802.1X: port 4/1 BkEnd BKEND_RESPONSE -->
BKEND_SUCCESS
```

```
: 802.1X: port 4/1 BkEnd BKEND_SUCCESS --> BKEND_IDLE
: 802.1X: port 4/1 AuthPAE AUTH_ENTCATING --> AUTH_ENTCATED
: 802.1X: Port 4/1 Programming Permitted MAC 0000.00aa.0001 on VLAN 1
: 802.1X: Timer tick expired
: 802.1X: Timer tick expired
```

debug dot1x dumpclass

Syntax: [**no**] **debug dot1x dumpclass**

This command displays the internal data structure. Command output resembles the following example.

```
Brocade# debug dot1x dumpclass
DOT1X Class: 0x2085e280
 Flags:  EnabAll: 0 ReAuthEnab/ReAuthMax: 0/3 MaxReq: 3 SysAuthEnab: 0
         AuthFailAction/Vlanid: Restricted/99 Aging/Age: All/120
 Timers: SecHold: 60 Quiet: 60 TxWhen: 30 ReAuth: 3600 SuppTmO: 30 SrvrTmO: 30
```

debug dot1x events

Syntax: [**no**] **debug dot1x events**

This command displays authentications that have failed or succeeded, the application of VLAN and ACLs requested by RADIUS, and so on. This command works globally across all ports. Command output resembles the following example.

```
Brocade# debug dot1x events
        events:  debugging is on
: 802.1X: port 4/1 Rx EAPOL_START
: 802.1X: port 4/1 BkEnd BKEND_INVALID --> BKEND_INIT
: 802.1X: port 4/1 BkEnd BKEND_INIT --> BKEND_IDLE
: 802.1X: port 4/1 AuthPAE AUTH_INVALID --> AUTH_INIT
: 802.1X: port 4/1 AuthPAE AUTH_INIT --> AUTH_DISCONCTED
: 802.1X: port 4/1 AuthPAE AUTH_DISCONCTED --> AUTH_CONNTING
: 802.1X: port 4/1 Tx (OK) EAPOL-EAP-REQUEST-ID Pkt (EAPId: 1)
: 802.1X: port 4/1 Tx (OK) EAPOL-EAP-REQUEST-ID Pkt (EAPId: 1)
: 802.1X: port 4/1 Rx EAPOL-EAP-RESPONSE-ID Pkt (EAPId: 1)
: 802.1X: port 4/1 AuthPAE AUTH_CONNTING --> AUTH_ENTCATING
: 802.1X: port 4/1 BkEnd BKEND_IDLE --> BKEND_RESPONSE
: 802.1X: port 4/1 Tx EAP PDU (EAPId: 1) to AuthServer
: 802.1X: port 4/1 Rx EAPOL-EAP-RESPONSE-ID Pkt (EAPId: 1)
: 802.1X: port 4/1 Rx AAA_INTERACTIVE from AuthServer
: 802.1X: port 4/1 Rx from Server: EAP-REQUEST-MD5 (EAPId: 2) Len 22
: 802.1X: port 4/1 BkEnd BKEND_RESPONSE --> BKEND_REQUEST
: 802.1X: port 4/1 Fwd (OK) EAPOL-EAP Pkt (EAPId: 2) from AuthServer to Supplicant
: 802.1X: port 4/1 Rx EAP-RESPONSE-MD5 Pkt (EAPId: 2) Len 24
: 802.1X: port 4/1 BkEnd BKEND_REQUEST --> BKEND_RESPONSE
: 802.1X: port 4/1 Tx EAP PDU (EAPId: 2) to AuthServer
: 802.1X: port 4/1 Rx AAA_ACCEPT from AuthServer
: 802.1X: Port 4/1 Created user-defined mac filter 400.
: 802.1X: Port 4/1 Binding with the RADIUS assigned MAC ACL ID: 400
: 802.1X: port 4/1 Rx Tunnel Data (Type=0, Medium_Type=0, PvtGrpId=NULL)
: 802.1X: port 4/1 Tx (OK) EAPOL-SUCCESS Pkt (EAPId: 2)
: 802.1X: port 4/1 BkEnd BKEND_RESPONSE --> BKEND_SUCCESS
: 802.1X: port 4/1 BkEnd BKEND_SUCCESS --> BKEND_IDLE
: 802.1X: port 4/1 AuthPAE AUTH_ENTCATING --> AUTH_ENTCATED
: 802.1X: Port 4/1 Programming Permitted MAC 0000.00aa.0001 on VLAN 1
: 802.1X: port 4/1 Rx EAPOL_LOGOFF
: 802.1X: port 4/1 AuthPAE AUTH_ENTCATED --> AUTH_DISCONCTED
```

```
: 802.1X: Port 4/1 Deleting Permitted MAC 0000.00aa.0001 on VLAN 1
: 802.1X: Port 4/1 Unbinding with the dynamic assigned L2 ACL: 400
: 802.1X: Port 4/1 Deleting the user defined L2 ACL: 400
: 802.1X: port 4/1 Tx (OK) EAPOL-FAIL Pkt (EAPId: 3)
: 802.1X: port 4/1 Tx (OK) EAPOL-EAP-REQUEST-ID Pkt (EAPId: 0)
```

debug dot1x fault

Syntax: [**no**] **debug dot1x fault**

This command reports any kind of internal errors, such as out-of-memory, invalid RADIUS-response, and so on. This command works globally across all ports. Command output resembles the following example.

```
Brocade# debug dot1x fault
fault:  debugging is on
: 802.1X: Port 4/1 Preprocess of user-defined L2-ACL
failed due to invalid ACL
: 802.1X: Port 4/1 Preprocess of user-defined L2-ACL failed due to invalid ACL
: 802.1X: Port 4/1 Preprocess of user-defined L2-ACL failed due to invalid ACL
: 802.1X: Port 4/1 Preprocess of user-defined L2-ACL failed due to invalid ACL
: 802.1X: Port 4/1 Preprocess of user-defined L2-ACL failed due to invalid ACL
```

debug dot1x packets

Syntax: [**no**] **debug dot1x packets**

This command displays information about 802.1x packets. Command output resembles the following example.

```
Brocade# debug dot1x packets
   packets:  debugging is on
: 802.1X: port 4/1 Rx EAPOL Pkt SA=0000.00aa.0001, DA=0000.0000.0003, len=0
EAP START   Dec 21 17:26:04   dump RX 802.1X: 18 bytes
0180c200 00030000 00aa0001 888e0101    ................
00004ef5                               ..
: 802.1X 4/1 Tx EAPOL on vlan_id 1 for dst 0000.00aa.0001, len 9:
EAP PACKET - EAPCode: REQUEST  EAPType: IDENTITY
dump TX 802.1X: 9 bytes
01000005 01010005 01000000             .........
802.1X 4/1 Tx EAPOL on vlan_id 1 for dst 0000.00aa.0001, len 9:
EAP PACKET - EAPCode: REQUEST  EAPType: IDENTITY
dump TX 802.1X: 9 bytes
01000005 01010005 01000000             .........
: 802.1X: port 4/1 Rx EAPOL Pkt SA=0000.00aa.0001, DA=0000.0000.0003, len=6
EAP PACKET - EAPCode: RESPONSE  EAPType: IDENTITY
dump RX 802.1X: 24 bytes
0180c200 00030000 00aa0001 888e0100    ................
: 802.1X: port 4/1 Rx EAPOL Pkt SA=0000.00aa.0001, DA=0000.0000.0003, len=6
```

debug dot1x port

Syntax: [**no**] **debug dot1x port** [**all** | **event** | **timer**] *slotnum/portnum*

- **all** *slotnum/portnum* - Displays 802.1x event and timer information for a specified port.
- **event** *slotnum/portnum* - Displays 802.1x event information for a specified port.
- **timer** *slotnum/portnum* - Displays 802.1x timer settings for a specified port.

This command displays event and timer information for a specified port, as shown in the following example.

```
Brocade# debug dot1x port all 3/15
802.1X Events debugging on port 94 ON
802.1X Timers debugging on port 94 ON

SYSLOG: Nov 7 17:39:22:<12>MLX_4K, DOT1X: Port 3/15, MAC Address 0000.00cd.5f4e
Access: unauthenticated
: 802.1X: port 3/15 Rx EAPOL-EAP-RESPONSE-ID Pkt (EAPId: 1)
: 802.1X: port 3/15 AuthPAE AUTH_CONNTING --> AUTH_ENTCATING
: 802.1X: port 3/15 BkEnd BKEND_IDLE --> BKEND_RESPONSE
: 802.1X: port 3/15 aWhile timer (AuthServer) started for 35 secs
: 802.1X: port 3/15 Tx EAP PDU (EAPId: 1) to AuthServer
: 802.1X: port 3/15 Rx AAA_INTERACTIVE from AuthServer
: 802.1X: port 3/15 Rx from Server: EAP-REQUEST-PEAP(EAPId: 237)Len 6
: 802.1X: port 3/15 BkEnd BKEND_RESPONSE --> BKEND_REQUEST
: 802.1X: port 3/15 Fwd(OK) EAPOL-EAP Pkt (EAPId:237) from AuthServer to
Supplicant
: 802.1X: port 3/15 aWhile timer (Supplicant) started for 30 secs
: 802.1X: port 3/15 Rx EAP-RESPONSE-NAK Pkt (EAPId: 237) Len 6
: 802.1X: port 3/15 BkEnd BKEND_REQUEST --> BKEND_RESPONSE
: 802.1X: port 3/15 aWhile timer (AuthServer) started for 35 secs
: 802.1X: port 3/15 Tx EAP PDU (EAPId: 237) to AuthServer
: 802.1X: port 3/15 Rx AAA_INTERACTIVE from AuthServer
: 802.1X: port 3/15 Rx from Server: EAP-REQUEST-MD5 (EAPId: 238) Len 30
: 802.1X: port 3/15 BkEnd BKEND_RESPONSE --> BKEND_REQUEST
: 802.1X: port 3/15 Fwd(OK) EAPOL-EAP Pkt (EAPId: 238) from AuthServer to
Supplicant
: 802.1X: port 3/15 aWhile timer (Supplicant) started for 30 secs
: 802.1X: port 3/15 Rx EAP-RESPONSE-MD5 Pkt (EAPId: 238) Len 23
: 802.1X: port 3/15 BkEnd BKEND_REQUEST --> BKEND_RESPONSE
: 802.1X: port 3/15 aWhile timer (AuthServer) started for 35 secs
: 802.1X: port 3/15 Tx EAP PDU (EAPId: 238) to AuthServer
: 802.1X: port 3/15 Rx AAA_ACCEPT from AuthServer
: 802.1X: port 3/15 Tx (OK) EAPOL-SUCCESS Pkt (EAPId: 238)
: 802.1X: port 3/15 BkEnd BKEND_RESPONSE --> BKEND_SUCCESS
: 802.1X: port 3/15 BkEnd BKEND_SUCCESS --> BKEND_IDLE
: 802.1X: port 3/15 AuthPAE AUTH_ENTCATING --> AUTH_ENTCATED
: SYSLOG: Nov 7 17:39:22:<14>MLX_4K, DOT1X: Port 3/15, MAC Address 0000.00cd.5f4e
802.1X: Port 3/15 Programming Permitted MAC 0000.00cd.5f4e on VLAN 1
Access: authorized
```

debug dot1x state

Syntax: [**no**] **debug dot1x state** *slotnum/portnum*

This command displays information about the 802.1x state of a specified port. Command output resembles the following example.

```
Brocade# debug dot1x state 4/1
Port 4/1. #Sess 0 #RestSess 0 #AuthSess 0. #DeniedSess 0
DfACLIn 0 DfACLOut 0 DfL2ACL 0. DfVLAN 4096 InVLAN 4096(UserCfg)
Flags pEnab pCtrl reAut ACLSS MClnt |X| Aging RvtVl DynVl OvRst DoSPr
1    AUTO   0     1     1      None     0     1     1     0
Timers TxEAP 23. EAPTries: 0
```

debug dot1x timers

Syntax: [no] debug dot1x timers

This command enables debugging and displays information about 802.1x timers. Command output resembles the following example.

```
Brocade# debug dot1x timers
timers:  debugging is on
802.1X: Timer tick expired
802.1X: Timer tick expired
802.1X: Timer tick expired
802.1X: Timer tick expired
802.1X: Port 4/1 txEAP timer expired. Transmitting an EAP ReqId
802.1X: Timer tick expired
802.1X: Timer tick expired
802.1X: Timer tick expired
802.1X: port 4/1 aWhile timer (Supplicant) started for 30 secs
802.1X: port 4/1 aWhile timer (AuthServer) started for 30 secs
802.1X: port 4/1 aWhile timer (AuthServer) started for 30 secs
802.1X: Timer tick expired
802.1X: port 4/1 aWhile timer (Supplicant) started for 30 secs
802.1X: port 4/1 aWhile timer (AuthServer) started for 30 secs
802.1X: Timer tick expired
802.1X: Port 3/15 txEAP timer expired. Transmitting an EAP ReqId
802.1X: Port 3/17 txEAP timer expired. Transmitting an EAP ReqId
802.1X: Port 3/18 txEAP timer expired. Transmitting an EAP ReqId
802.1X: Port 3/19 txEAP timer expired. Transmitting an EAP ReqId
802.1X: Port 3/20 txEAP timer expired. Transmitting an EAP ReqId
802.1X: port 4/1 aWhile timer (AuthServer) started for 30 secs
802.1X: port 4/1 aWhile timer (Supplicant) started for 30 secs
802.1X: port 4/1 aWhile timer (AuthServer) started for 30 secs
802.1X: Timer tick expired
802.1X: port 4/1 aWhile timer (AuthServer) started for 30 secs
802.1X: port 4/1 aWhile timer (Supplicant) started for 30 secs
802.1X: port 4/1 aWhile timer (AuthServer) started for 30 secs
802.1X: Timer tick expired
```

Configuration notes

- The client’s 802.1x MAC session establishes a relationship between the user name and the MAC address used for authentication. If a user attempts to gain access from different clients (with different MAC addresses), the user must be authenticated from each client.
- If a client has been denied access to the network (that is, the client’s 802.1x MAC session is set to “access-denied”), then you can cause the client to be re-authenticated by manually disconnecting the client from the network, or by using the **clear dot1x mac-session** *mac-address* command.
- When a client has been denied access to the network, the 802.1x MAC session is aged out if no traffic is received from the client’s MAC address over a fixed hardware aging period (70 seconds), plus a configurable software aging period. You can optionally change the software aging period for 802.1x MAC sessions or disable aging altogether. After the denied client’s 802.1x MAC session is aged out, traffic from that client is no longer blocked, and the client can be re-authenticated.
- To implement 802.1x port security, at least one of the RADIUS servers identified to the Brocade device must support the 802.1x standard.

Denial of Service attacks

In a Denial of Service (DoS) attack, a router is flooded with useless packets, hindering normal operation. Brocade devices include measures for defending against two types of DoS attacks: Smurf attacks and TCP SYN attacks.

A Smurf attack is a kind of DoS attack where an attacker causes a victim to be flooded with ICMP echo (Ping) replies sent from another (intermediary) network. For detailed information about how to prevent Smurf attacks, refer to the *Brocade MLX Series and NetIron Family Configuration Guide.*

In a TCP SYN attack, an attacker floods a host with TCP SYN packets that have random source IP addresses. For each of these TCP SYN packets, the destination host responds with a SYN ACK packet and adds information to the connection queue. However, because the source host does not exist, no ACK packet is sent back to the destination host, and an entry remains in the connection queue until it ages out (after about a minute). If the attacker sends enough TCP SYN packets, the connection queue can fill up, and service can be denied to legitimate TCP connections.

DoS show commands

This section describes the DoS-related show commands.

show statistics dos-attack

Syntax: **show statistics dos-attack**

This command displays information about ICMP and TCP SYN packets dropped, passed, and blocked because burst thresholds were exceeded. Command output resembles the following example.

```
Brocade# show statistics dos-attack
------ Local Attack Statistics ------
ICMP Drop Count     Port Block Count    SYN Drop Count    SYN Block Count
---------------     ----------------    --------------    ---------------
              0                    0                 0                  0
```

Clearing DoS attack statistics

To clear statistics about ICMP and TCP SYN packets dropped because burst thresholds were exceeded, enter the following command.

clear statistics dos-attack

Syntax: **clear statistics dos-attack**

DoS debug commands

There are no debug commands specific to DoS attacks.

Port loop detection

Brocade port loop detection allows the Brocade device to detect loops and disable a port that is on the receiving end of a loop. A loop is detected by sending test packet BPDUs. There are two modes of loop detection: Loose mode and Strict mode.

Port loop detection show commands

This section describes the show command that displays loop detection information.

show loop-detection

Syntax: **show loop-detection**

This command displays the loop detection configuration and current status, as shown in the following example.

```
Brocade# show loop-detection
loop detection packets interval: 10 (unit 100 msec)
loop detection disable duration: 10 (In minutes, 0 means permanently disabled)
Ports mode loop detection
=========================
port-num disable-count
1/12 0
1/11 0
Vlan mode loop detection
========================
vlan-id disable-count
100 2
10 0
200 0
Ports disabled by loop detection
================================
port age(minutes) disable cause
1/11 1 Disabled by VLAN: 100 loopdetect 1/11
1/12 1 Disabled by VLAN: 100 loopdetect 1/12
```

Port loop detection debug commands

This section describes the debug command that generates loop detection information.

debug loopdetect

Syntax: [**no**] **debug loopdetect** [**detail** | **error** | **info**]

This command enables loop detect debugging.

- **detail** - Displays loop detect detail messages.
- **error** - Displays loop detect error messages.
- **info** - Displays loop detect information messages.

The following command example displays loop detect information messages.

```
Brocade# debug loopdetect info
loop_detect_cu_enable: sending itc message for Inv VLAN: 13
ITC loop_detect_itc_enable rxd
```

```
set_port_or_vlan_loop_detection(port Inv VLAN: 13 , del=0)
[loop_detect_send_ipc_request]: sending ipc msg 1 for port Inv VLAN: 13  to fid
53283 enable 0 strict vlan VLAN: 0
loop_detect_clear_loop_detection: for port_id Inv VLAN: 13
Dec  1 01:21:00 loop_detect_port_event_handler - port 2/5 up event
Dec  1 01:21:00 loop_detect_port_event_handler - port 2/7 up event
loop
loop_detect_cu_enable: sending itc message for Inv VLAN: 13
ITC loop_detect_itc_enable rxd

set_port_or_vlan_loop_detection(port Inv VLAN: 13 , del=1)
[loop_detect_send_ipc_request]: sending ipc msg 0 for port Inv VLAN: 13  to fid
53283 enable 1 strict vlan VLAN: 0
[loop_detect_lp_ipc_control] rx opcode=3
loop_detect_process_loop_detected: (2/5 VLAN: 13 ) sending_port Inv type 2
Dec  1 01:21:08 loop_detect_process_loop_detected: loop detect is shutting port
2/5 VLAN: 13
[loop_detect_lp_ipc_control] rx opcode=3
loop_detect_process_loop_detected: (2/7 VLAN: 13 ) sending_port Inv type 2
Dec  1 01:21:08 loop_detect_process_loop_detected: loop detect is shutting port
2/7 VLAN: 13
```

The following command example displays loop detect detail messages.

```
Brocade# debug loopdetect detail
loop_detect_send_bpdu: i/f 2/5 stp state is blocking, do not send loop-detect
loop_detect_send_bpdu: send loop-detect to port Inv VLAN: 13
loop_detect_send_bpdu: i/f 2/5 stp state is blocking, do not send loop-detect
loop_detect_send_bpdu: send loop-detect to port Inv VLAN: 13
loop_detect_send_bpdu: i/f 2/5 stp state is blocking, do not send loop-detect
loop_detect_send_bpdu: send loop-detect to port Inv VLAN: 13
loop_detect_send_bpdu: i/f 2/5 stp state is blocking, do not send loop-detect
loop_detect_send_bpdu: send loop-detect to port Inv VLAN: 13
loop_detect_send_bpdu: i/f 2/5 stp state is blocking, do not send loop-detect
loop_detect_send_bpdu: send loop-detect to port Inv VLAN: 13
loop_detect_send_bpdu: i/f 2/5 stp state is blocking, do not send loop-det
```

Configuration notes

The following information applies to Loose mode loop detection:

- Loop detection is configured on the VLAN. Different VLANs may disable different ports.
- Loose mode can disable multiple ports of a loop. A disabled port affects every VLAN using it.
- Loose mode disables the receiving port if packets originate from any port or member port of a VLAN on the same device.
- The VLAN of the receiving port must be configured for loop detection in order to disable the port.
- Loose mode floods test packets to the entire VLAN. This can impact system performance if too many VLANs are configured for Loose mode loop detection.

The following information applies to Strict mode loop detection:

- A port is disabled only if a packet is looped back to that same port. Loop detection must be configured on the physical port.
- Strict mode overcomes specific hardware issues where packets are echoed back to the input port.

Port mirroring and monitoring

You can monitor the traffic on the Brocade ports by configuring another port to mirror the traffic on the ports you want to monitor. The port thus configured is called a mirror port. By attaching a protocol analyzer to the mirror port, you can observe the traffic on the monitored ports.

Port mirroring show commands

This section describes the show commands that display port mirroring information.

show monitor config

Syntax: show monitor config

This command displays the inbound and outbound traffic that is being mirrored to each mirror port, as shown in the following example.

```
Brocade# show monitor config
Monitored Port 3/1
    Input traffic mirrored to: 2/1
    Output traffic mirrored to: 1/1
Monitored Port 4/1
    Input traffic mirrored to: 1/2
    Output traffic mirrored to: 2/1
```

show monitor actual

Syntax: show monitor actual

This command displays the actual traffic being mirrored to each mirror port, as shown in the following example.

```
Brocade# show monitor actual
Monitored Port 3/1
    Output traffic mirrored to: 1/1
Monitored Port 4/1
    Input traffic mirrored to: 1/2
```

This command output displays the output traffic mirrored to mirror port 1/1 from port 3/1 and input traffic mirrored to mirror port 1/2 from port 4/1, which are explicitly configured.

Port mirroring debug commands

This section describes the debug command that generates port mirroring information.

debug access-list mirror generic

Syntax: [no] debug access-list mirror generic

This command displays information about generic access list mirroring activity. Command output resembles the following example.

```
Brocade# debug access-list mirror generic
        ACL based Mirroring-Generic:  debugging is on
Port24       7212413945                    0        0       0         0       0
Port47                0           6559006109        0       0         0       0
```

```
Port48               0           1363150585        0       0        0       0

Nov 30  18:28:14 trying port 1/10 for out ACL, ptr=24dd2112
Nov 30  18:28:14 trying port 1/10 for out ACL, ptr=24dd2112
PPCR 2:1: Reset IPv4/L2 outbound Rule ACL CAM count
PPCR 2:1: Free IPv4/L2 outbound Rule ACL CAM Partition
PPCR 1:1: Reset IPv4/L2 outbound Rule ACL CAM count
PPCR 1:1: Free IPv4/L2 outbound Rule ACL CAM Partition
PPCR 1:2: Reset IPv4/L2 outbound Rule ACL CAM count
PPCR 1:2: Free IPv4/L2 outbound Rule ACL CAM Partition
PPCR 2:1: Free IPv4/L2 inbound Rule ACL CAM Partition
PPCR 1:1: Reset IPv4/L2 inbound Rule ACL CAM count
PPCR 1:1: Free IPv4/L2 inbound Rule ACL CAM Partition
PPCR 1:2: Free IPv4/L2 inbound Rule ACL CAM Partition
Extended IP access list 100
    0: permit enable-accounting ip any any mirror
!
interface ethernet 1/24
enable
ip access-group 100 in
acl-mirror-port ethernet 1/48
```

Configuration notes

The following information must be considered when configuring ACL-based inbound mirroring:

- All ports using the same PPCR must have a common destination ACL mirror port when configuring ACL-based inbound mirroring.
- ACL-based inbound mirroring and port-based inbound mirroring are mutually exclusive on a per-port basis.

For ACL CAM sharing to function, either of the following conditions must be operative:

- All ports that belong to a PPCR must have the **acl-mirror-port** command configured to direct mirrored traffic to the same port.
- None of the ports that belong to the PPCR can have the **acl-mirror-port** command configured.

RADIUS

You can use a Remote Authentication Dial In User Service (RADIUS) server to secure the following types of access to the Brocade devices:

- Telnet access
- SSH access
- Web management access
- Access to the Privileged EXEC level and CONFIG level of the CLI

NOTE
Brocade devices do not support RADIUS security for SNMP (Brocade Network Advisor) access.

RADIUS show commands

This section describes the RADIUS-specific show commands.

show aaa

Syntax: show aaa

This command displays information about all RADIUS servers configured on the device. Command output resembles the following example.

```
Brocade# show aaa
Radius key: networks
Radius retries: 3
Radius timeout: 3 seconds
Radius dead-time: 3 minutes
Radius Server: 10.95.6.90 Auth Port=1645 Acct Port=1646:
opens=2 closes=1 timeouts=1 errors=0
packets in=1 packets out=4
health-check=enabled|disabled dead-time-interval=45m
auto-authenticate-time-interval=30m available|dead
```

show web

Syntax: show web

This command displays the privilege level of Web management interface users.

```
Brocade# show web
User                                Privilege       IP address
set                                         0       192.168.1.234
```

RADIUS debug commands

This section describes the debug command used for monitoring the RADIUS servers identified on the device.

debug ip aaa

Syntax: [no] debug ip aaa

This command enables authentication, authorization, and accounting (AAA) debugging and displays debug information about AAA or RADIUS authentication. Command output resembles the following example.

```
Brocade# debug ip aaa
IP:  aaa debugging is on
**Radius StartTimer cu_radius_start_health_check called


 **Radius StartTimer Success

Sep 30 06:26:38 RADIUS HCSM received event eHealthCheckPollTimerExpired for server
2001:DB8::7ae7:d1ff:fe8d:1b82:1 when in state sHealthCheckInit

Sep 30 06:26:38 AAA: Reuse RADIUS UDP port by health check

Sep 30 06:26:38 Tracing the outgoing Radius Authentication packet..
```

```
Sep 30 06:26:38 UDP packet source IP=2001:DB8::7ae7:d1ff:fe8d:1b81, port=1052,
destination IP=2001:DB8::7ae7:d1ff:fe8d:1b82, port=1646
Sep 30 06:26:38 RADIUS HCSM received event eHealthCheckPollTimerExpired for server
2001:DB8::7ae7:d1ff:fe8d:1b82:0 when in state sHealthCheckInit
```

Configuration notes

- You must deploy at least one RADIUS server in your network.
- Brocade devices support authentication using up to eight RADIUS servers. The device tries to use the servers in the order you add them to the device's configuration. If one RADIUS server is not responding, the Brocade device tries the next one in the list.
- You can select only one primary authentication method for each type of access to a device (CLI through Telnet, CLI Privileged EXEC and CONFIG levels). For example, you can select RADIUS as the primary authentication method for Telnet CLI access, but you cannot also select TACACS+ authentication as the primary method for the same type of access. However, you can configure backup authentication methods for each access type.

sFlow

sFlow is a system for observing traffic flow patterns and quantities within and among a set of Brocade devices. sFlow performs the following tasks:

- Samples packet flows
- Collects the packet headers from sampled packets and collects ingress and egress information on these packets
- Composes the collected information into flow sample messages
- Relays these messages to an external device known as a collector

Participating devices also relay byte and packet counter data (counter samples) for ports to the collector.

Displaying sFlow statistics

When traffic is received in the sFlow enabled interface, packets are sent to the LP CPU. The packets are processed by sFlow module by adding sFlow header along with the packet header and thereafter sent to the sFlow collector. The statistics of sFlow samples are maintained in the sFlow collector.

This section describes the show command that displays sFlow information.

show sflow

Syntax: **show sflow**

This command displays sFlow configuration information and statistics, as shown in the following example.

```
Brocade# show sflow
sFlow services are enabled.
sFlow agent IP address: 10.30.30.2
Collector IP 10.10.10.1, UDP 6343
Polling interval is 20 seconds.
```

```
Configured default sampling rate: 1 per 2048 packets.
0 UDP packets exported
0 sFlow samples collected.
sFlow ports   Global Sample Rate   Port Sample Rate   Hardware Sample Rate
        3/1                 2048                2048   2048
        3/2                 2048                2048   2048
        3/3                 2048                2048   2048
        3/4                 2048                2048   2048
```

show sflow statistics

Syntax: show sflow statistics

This command displays the total count per interface for both sFlow and ACL based samples in all the slots where sFlow is configured.

```
Brocade# show sflow statistics
Sflow Ports    Flow Samples count Acl Samples Count
1/1              800                 0
1/5              0                   900
2/1              600                 0

Brocade# show sflow statistics  ethernet 1/1
Sflow Ports   Flow Samples count Acl Samples Count
1/1              800                 0
```

Clearing sFlow statistics

To clear the UDP packet and sFlow sample counters in the show sflow display, enter the following command.

clear statistics

Syntax: clear statistics

This command clears the values in the following fields of the show sflow display:

- UDP packets exported
- sFlow samples collected

Use the **clear statistics sflow** command to clear all the statistics collected per interface.

```
Brocade# clear statistics sflow
Brocade# show sflow statistics  ethernet 1/1
Sflow Ports   Flow Samples count Acl Samples Count
1/1              0                   0
```

sFlow debug commands

There are no debug commands specific to sFlow.

Configuration notes

- Before you enable sFlow, make sure the device has an IP address that sFlow can use as its source address.
- It is recommended that you do not change the denominator to a value lower than the default. Sampling requires CPU resources. Using a low denominator for the sampling rate can cause high CPU utilization.
- sFlow uses CPU resources to send sFlow samples to the collector. If you set a low sampling value on a high rate interface (for example, 10 GE), the interface module CPU utilization can become high.
- If you change the router ID or other IP address value that sFlow uses for its agent address, you must disable and then re-enable sFlow to cause the feature to use the new source address.
- Sample data is collected from inbound traffic on ports enabled for sFlow. However, both traffic directions are counted for byte and packet counter statistics sent to the collector.
- Port mirroring and sFlow cannot be enabled simultaneously on the same port.

SNMP

The Simple Network Management Protocol (SNMP) forms part of the Internet Protocol (IP) suite as defined by the Internet Engineering Task Force (IETF). SNMP is used in network management systems to monitor network-attached devices administration and management.

SNMP is disabled by default on Brocade devices. SNMP must be enabled in order to manage a Brocade device using Brocade Network Advisor.

SNMP show commands

This section describes the show commands that display SNMP information.

show snmp server

Syntax: **show snmp server**

This command displays both the read-only and read-write community strings.

NOTE
If display of the strings is encrypted, the strings are not displayed. Encryption is enabled by default.

To display the SNMP community string, enter the following commands.

```
Brocade(config)# enable password-display
Brocade(config)# show snmp server
```

The **enable password-display** command allows the community string to be displayed, but only in the output of the **show snmp server** command. Display of the string is still encrypted in the startup configuration file and running configuration file. Enter the **enable password-display** command at the global CONFIG level of the CLI.

show snmp engineid

Syntax: **show snmp engineid**

This command displays the engine ID of a management module, as shown in the following example.

```
Brocade# show snmp engineid
Local SNMP Engine ID: 800007c70300e05290ab60
Engine Boots: 3
Engine time: 5
```

show snmp group

Syntax: show snmp group

This command displays the definition of an SNMP group, as shown in the following example.

```
Brocade# show snmp group
groupname = exceptifgrp
security model = v3
security level = authNoPriv
ACL id = 2
readview = exceptif
writeview = <none>
```

show snmp user

Syntax: show snmp user

This command displays the definition of an SNMP user account, as shown in the following example.

```
Brocade# show snmp user
username = bob
acl id = 2
group = admin
security model = v3
group acl id = 0
authtype = md5
authkey = 3aca18d90b8d172760e2dd2e8f59b7fe
privtype = des,  privkey = 1088359afb3701730173a6332d406eec
engine ID= 800007c70300e052ab0000
```

SNMP debug commands

There are no debug commands specific to SNMP.

Configuration notes

- SNMP read-only or read-write community strings are always required for SNMP access to the device.
- SNMP access is disabled by default.
- The syntax for using ACLs for SNMP access is different from the syntax for controlling Telnet, SSH, and Web management access using ACLs.
- When **snmp-server community** is configured, all incoming SNMP packets are validated first by their community strings and then by their bound ACLs. Packets are permitted if no filters are configured for an ACL.

- If you do not enable Telnet access, you can access the CLI using a serial connection to the management module. If you do not enable SNMP access, you will not be able to use Brocade Network Advisor or third-party SNMP management applications.
- You cannot authenticate Simple Network Management Protocol (SNMP) access to a Brocade device using TACACS or TACACS+.
- Brocade devices do not support RADIUS security for SNMP access.
- The TACACS or TACACS+, RADIUS, and Telnet login password authentication methods are not supported for SNMP access.
- For CLI access, you must configure authentication-method lists if you want the device to authenticate access using local user accounts or a RADIUS server. Otherwise, the device will authenticate using only the locally based password for the Super User privilege level.
- When no authentication-method list is configured specifically for Web management access, the device performs authentication using the SNMP community strings:
  - For read-only access, use the user name "get" and the password "public". The default read-only community string is "public".
  - There is no default read-write community string, which means you cannot open a read-write management session using the Web management interface. You first must configure a read-write community string using the CLI. Then you can log on using "set" as the user name and the read-write community string you configure as the password.
- If you configure an authentication-method list for Web management access and specify "local" as the primary authentication method, users who attempt to access the device using the Web management interface must supply a user name and password configured in one of the local user accounts on the device. You *cannot* access the device by entering "set" or "get" and the corresponding SNMP community string.
- For devices that can be managed using Brocade Network Advisor, the default authentication method (if no authentication-method list is configured for SNMP) is the CLI Super User level password. If no Super User level password is configured, then access through Brocade Network Advisor is not authenticated. To use local user accounts to authenticate access through Brocade Network Advisor, configure an authentication-method list for SNMP access and specify "local" as the primary authentication method.

TACACS and TACACAS+

TACACS+ is an enhancement to the TACACS security protocol. TACACS+ improves on TACACS by separating the functions of authentication, authorization, and accounting (AAA) and by encrypting all the traffic between the Brocade device and the TACACS+ server.

TACACS show commands

This section describes the show commands that display information about the TACACS servers.

show aaa

Syntax: show aaa

This command displays information about all the TACACS+ and RADIUS servers identified on the device.

```
Brocade# show aaa
Tacacs+ key: foundry
Tacacs+ retries: 1
Tacacs+ timeout: 15 seconds
Tacacs+ dead-time: 3 minutes
Tacacs+ Server: 10.95.6.90 Port:49:
                opens=6 closes=3 timeouts=3 errors=0
                packets in=4 packets out=4
no connection
Radius key: networks
Radius retries: 3
Radius timeout: 3 seconds
Radius dead-time: 3 minutes
Radius Server: 10.95.6.90 Auth Port=1645 Acct Port=1646:
                opens=2 closes=1 timeouts=1 errors=0
                packets in=1 packets out=4
no connection
```

show web

Syntax: show web

The **show web** command displays the privilege level of Web management interface users.

```
Brocade# show web
User                             Privilege       IP address
set                                     0        192.168.1.234
```

TACACS debug commands

This section describes the debug command used for monitoring the TACACS servers.

debug ip aaa

Syntax: [**no**] **debug ip aaa** [**event** | **ipc** | **itc** | **packet**]

- **event** - Displays information about ARP events.
- **ipc** - Displays information about ARP IPC messages.
- **itc** - Displays information about ARP ITC messages.
- **packet** - Displays information about ARP packets.

This command displays information about AAA or TACACS+ authentication, as the following example illustrates.

```
Brocade# debug ip aaa
COMMAND ACCOUNTING STARTS...
RADIUS accounting for context 2
Reseting RADIUS Client structure
RADIUS: Reset client 0, Total number of active clients=1

AAA: Open RADIUS UDP port

Tracing the outgoing Radius Accounting packet..
UDP packet source IP=172.20.1.2, port=1061, destination IP=172.2
0.1.1, port=1813
**Radius timer - 0 kicks in.
RADIUS retransmission for user lab, context=2, client=0
```

```
Tracing the outgoing Radius Accounting packet..
UDP packet source IP=172.20.1.2, port=1061, destination IP=172.20.1.1, port=1813
**Radius timer - 0 kicks in.
RADIUS retransmission for user lab, context=2, client=0
Tracing the outgoing Radius Accounting packet..
UDP packet source IP=172.20.1.2, port=1061, destination IP=172.20.1.1, port=1813
**Radius timer - 0 kicks in.
RADIUS timeout for user lab, context=2, client=0
RADIUS Timer cancelled for client 0.
Closing RADIUS UDP port
RADIUS: radius_authenticate_stop for client Idx 0. Actv Clients left 0
Reseting RADIUS Client structure
Accounting status - timeout.
Accounting status - reject.
aaa_send_aaa_response()..session 2, err_code=3
Unsuccessful accounting for session 2, code 3.
```

Configuration notes

- You must deploy at least one TACACS or TACACS+ server in your network.
- The Brocade device supports authentication using up to eight TACACS or TACACS+ servers. The device tries to use the servers in the order you add them to the device's configuration.
- You can select one primary authentication method for each type of access to a device (CLI through Telnet, CLI Privileged EXEC and CONFIG levels). For example, you can select TACACS+ as the primary authentication method for Telnet CLI access. You cannot also select RADIUS as a primary method for the same type of access, but you can configure backup authentication methods for each access application.
- You can configure the device to authenticate using a TACACS or TACACS+ server, but not both.
- TACACS+ provides for authentication, authorization, and accounting, but an implementation or configuration is not required to employ all three.
- If you erase a **tacacs-server** command (by entering the **no** form of the command), make sure you also erase the **aaa** commands that specify TACACS or TACACS+ as an authentication method. Otherwise, when you exit from the CONFIG mode or from a Telnet session, the system continues to believe it is TACACS or TACACS+ enabled and you will not be able to access the system.
- TACACS+ command authorization can be performed only for commands entered from Telnet or SSH sessions, or from the console. No authorization is performed for commands entered at the Web management interface or Brocade Network Advisor.

Common diagnostic scenarios

- During enable authentication with the login password, the password checked is the login password and not the enable password configured on the TACACS+ server.

  This problem is resolved when the "implicit-user" option is configured. The enable password then correctly checks against the enable password configured on the TACACS+ server.

- TACACS does not work after reboot. The key is still in configuration, but authentication does not work until the key is removed and added back to the configuration.

  This issue is resolved by upgrading the software version to reflect the latest patches and versions.

Telnet and SSH connections

The first time you log on to the console port, you must use a serial connection in order to assign an IP address to the port. Once an IP address is assigned, you can access the CLI through a local Telnet, SSH, or SNMP connection through the management port. When accessing the CLI through Telnet, you maybe prompted for a password. By default, the password required is the one you enter for general access at initial setup.

NOTE
Telnet, SSH, Web, and SNMP servers are disabled by default, and can be enabled selectively.

Telnet and SSH show commands

This section describes the show command that displays Telnet information.

show telnet

Syntax: **show telnet**

This command shows you the number of open Telnet sessions at any given time, including information about each session. Command output resembles the following example.

```
Brocade# show telnet
Console connections:
       established
       3 days 17 hours 31 minutes 27 seconds in idle
Telnet server status: Enabled
Telnet connections (inbound):
1      established, client ip address 10.53.1.65, privilege super-user
       you are connecting to this session
2      closed
3      closed
4      closed
5      closed
Telnet connections (outbound):
6      established, server ip address 10.47.2.200, from Telnet session 1
       4 seconds in idle
7      closed
8      closed
9      closed
10     closed
SSH server status: Enabled
SSH connections:
1      closed
```

Telnet and SSH debug commands

This section describes the debug commands that generate information about Telnet and SSH connections.

debug ip telnet

Syntax: [**no**] **debug ip telnet**

This command generates information about incoming Telnet connections, as shown in the following example.

```
Brocade# debug ip telnet
TELNET: Data is ready to receive
TELNET: Data is ready to receive
```

debug ip ssh

Syntax: [**no**] **debug ip ssh**

This command generates information about SSH connections.

The following example shows information about SSH connections for a user logging in.

```
Brocade# debug ip ssh
Nov 30 22:54:01 SSH[0]: ssh_is_connection_allowed..NEW CONNECTION
Nov 30 22:54:01 SSH[0]:ssh_socket_control: new connection
Nov 30 22:54:01 SSH[0]:ShtcpConnectionStatus: pending
Nov 30 22:54:01 SSH[0]:Awaiting child task init complete
Nov 30 22:54:01 SSH[0]:ssh_socket_control:Incoming connection ready
Nov 30 22:54:01 SSH:tcp incoming tcb 0x117ee46e, handle 0xa0000007
Nov 30 22:54:01 SSH[0]:ShtcpConnectionStatus: connection established
Nov 30 22:54:02 SSH[0]:ShtcpReceiveStatus: string length 47
Nov 30 22:54:02 SSH[0]:ShtcpSend: eSendComplete: sent 24/24
Nov 30 22:54:02 SSH[0]:ProcessPendingSend: send completion status=Complete,
result=0
Nov 30 22:54:02 SSH[0]:ShtcpSend: eSendComplete: sent 240/240
Nov 30 22:54:02 SSH[0]:ProcessPendingSend: send completion status=Complete,
result=0
Nov 30 22:54:02 SSH[0]:ShtcpReceiveStatus: string length 456
Nov 30 22:54:02 SSH[0]:ShtcpReceiveStatus: string length 272
Nov 30 22:54:04 SSH[0]:ShtcpSend: eSendComplete: sent 768/768
Nov 30 22:54:04 SSH[0]:ProcessPendingSend: send completion status=Complete,
result=0
Nov 30 22:54:05 SSH[0]:ShtcpReceiveStatus: string length 16
Nov 30 22:54:05 SSH[0]:ShtcpSend: eSendComplete: sent 16/16
Nov 30 22:54:05 SSH[0]:ProcessPendingSend: send completion status=Complete,
result=0
Nov 30 22:54:08 SSH[0]:ShtcpReceiveStatus: string length 52
Nov 30 22:54:08 SSH[0]:ShtcpSend: eSendComplete: sent 52/52
Nov 30 22:54:08 SSH[0]:ProcessPendingSend: send completion status=Complete,
result=0
Nov 30 22:54:10 SSH[0]:ShtcpReceiveStatus: string length 68
Nov 30 22:54:10 SSH[0]:ShtcpSend: eSendComplete: sent 84/84
Nov 30 22:54:10 SSH[0]:ProcessPendingSend: send completion status=Complete,
result=0
Nov 30 22:54:11 SSH[0]:ShtcpReceiveStatus: string length 116
Nov 30 22:54:11 SSH[0]:ShtcpSend: eSendComplete: sent 36/36
Nov 30 22:54:11 SSH[0]:ShtcpReceiveStatus: string length 68
Nov 30 22:54:11 SSH[0]:ProcessPendingSend: send completion status=Complete,
result=0
Nov 30 22:54:11 SSH[0]:ShtcpSend: eSendComplete: sent 52/52
Nov 30 22:54:11 SSH[0]:ShtcpReceiveStatus: string length 148
Nov 30 22:54:11 SSH[0]:ProcessPendingSend: send completion status=Complete,
result=0
Nov 30 22:54:11 SSH[0]:ShtcpSend: eSendComplete: sent 36/36
Nov 30 22:54:11 SSH[0]:ShtcpReceiveStatus: string length 68
Nov 30 22:54:11 SSH[0]:ProcessPendingSend: send completion status=Complete,
result=0
Nov 30 22:54:11 SSH[0]:ShtcpSend: eSendComplete: sent 36/36
```

```
Nov 30 22:54:11 SSH[0]:ShtcpReceiveStatus: string length 52
Nov 30 22:54:11 SSH[0]:ProcessPendingSend: send completion status=Complete,
result=0
Nov 30 22:54:11 SSH[0]:ssh_event_handler: listen event.
Nov 30 22:54:11 SSH[0]:***ssh_connection_task busy..return
Nov 30 22:54:11 SSH[0]:ssh_event_handler:sent banner and prompt
Nov 30 22:54:11 SSH[0]:ShtcpSend: eSendComplete: sent 36/36
Nov 30 22:54:11 SSH[0]:ProcessPendingSend: send completion status=Complete,
result=0
Nov 30 22:54:11 SSH[0]:ShtcpSend: eSendComplete: sent 68/68
Nov 30 22:54:11 SSH[0]:ShSendChannelData: eRpNoError
Nov 30 22:54:11 SSH[0]:ProcessPendingSend: send completion status=Complete,
result=0
Nov 30 22:54:11 SSH[0]:ssh_event_handler:Send event.
```

Configuration notes

- By default, a user logging in to the device through Telnet or SSH must first enter the User EXEC level. The user can enter the **enable** command to get to the Privileged EXEC level.
- You can select only one primary authentication method for each type of access to a device (CLI through Telnet, CLI Privileged EXEC and CONFIG levels). For example, you can select TACACS+ as the primary authentication method for Telnet CLI access, but you cannot also select RADIUS authentication as a primary method for the same type of access. However, you can configure backup authentication methods for each access type.

NTP

The Network Time Protocol (NTP) allows synchronization of system clocks among a set of distributed time servers and clients. The NTP transmits timekeeping information from the primary time servers to the secondary time servers and clients using both private networks and the public Internet.

NTP show commands

This section describes the show commands that display NTP information.

show ntp status

Syntax: **show ntp status**

This command displays information about the NTP status. The following is the command example, when the NTP panic mode is enabled.

```
Brocade# show ntp status
Clock is unsynchronized, no reference clock
NTP server mode is enabled, NTP client mode is enabled
NTP master mode is disabled, NTP master stratum is 8
NTP is in panic mode
```

You can disable the NTP panic mode by performing the following steps:

1. Remove the clock time zone and clock summer time configurations from the device by entering the following commands.

   ```
   Brocade(config)# no clock summer-time
   ```

```
Brocade(config)# no clock timezone us pacific
```

Syntax: [no] clock summer-time

Syntax: [no] clock timezone us *time-zone*

The *time-zone* variable specifies the timezone and can be any one of the following: alaska, aleutian, arizona, central, east-indiana, eastern, hawaii, michigan, mountain, pacific, samoa.

2. Exit the configuration mode by entering the following command.

```
Brocade(config)# exit
```

Syntax: exit

3. Set the clock using the current Coordinated Universal Time (UTC) time, by entering the following command.

```
Brocade# clock set 17:40:00 04-11-12
```

Syntax: [no] clock set *hh:mm:ss mm-dd-yy*

The *hh:mm:ss mm-dd-yy* variables specify the time to be set in timestamp format.

4. Wait for NTP to synchronize and then the syslog messages such as the following are displayed.

```
SYSLOG: <14>Apr 11 17:41:09 Brocade NTP: System clock is synchronized to
10.20.99.174. SYSLOG: <14>Apr 11 17:41:09 Brocade NTP: Stratum is changed to
4.
```

5. Reconfigure the clock time zone and clock summer time configurations that are removed in step 1.

```
Brocade# config terminal
Brocade(config)# clock timezone us pacific
Brocade(config)# show clock
09:41:27.182 Pacific Wed Apr 11 2012
Brocade(config)# clock summer-time
Brocade(config)# show clock
10:41:35.632 Pacific Wed Apr 11 2012
```

show ntp associations

Syntax: show ntp associations

This command displays information about NTP servers and peers associations, as shown in the following example.

```
Brocade# show ntp associations
address        ref clock    st  when poll reach delay offset  disp
*~172.19.69.1  172.24.114.33 3 25 64   3 2.89  0.234 39377
~2001:DB8::234
               INIT      16   -   64   0      0.00  0.000    15937

* synced, # selected, + candidate, - outlayer, x falseticker, ~ configured
```

show ntp associations detail

Syntax: show ntp associations detail

This command displays detailed association information of all the NTP servers and peers, as shown in the following example.

```
Brocade# show ntp association detail
```

```
10.216.1.101 configured server, sys peer,  stratum 2
 ref ID 10.123.2.5, time d21da706.1ed27000 (16:14:22.517107712 GMT+05:30 Fri Sep 16 2011)
 our mode client, peer mode server, our poll intvl 6, peer poll intvl 6,
 root delay 0.02256774 msec, root disp 0.01150512, reach 377, root dist 0.36969603
 delay 290.94232711 msec, offset -1.08355772 msec, dispersion 4.58729275,
 precision 2**-16, version 4
 org time d21da713.f8f25000 (16:14:35.4176629760 GMT+05:30 Fri Sep 16 2011)
 rcv time d21da714.2602742a (16:14:36.637695018 GMT+05:30 Fri Sep 16 2011)
 xmt time d21da713.d31f723f (16:14:35.3542053439 GMT+05:30 Fri Sep 16 2011)
 filter delay    296.5594  322.7792  323.5571  297.6697  290.9942  303.5554  305.9971  295.0019
 filter offset     0.4430  -13.4441  -14.2241   -4.0003   -1.0083   -1.4414   -3.0034    1.9941
 filter disp       1.9984    1.0025    0.0035    6.8889    5.8899    4.8895    3.9920    2.9944
 filter epoch        5779      5843      5909      5452      5518      5585      5650      5715
```

Syntax: show ntp associations detail *IPv4 address* | *IPv6 address*

- *IPv4 address* - Specifies the IPv4 address of the NTP peer.
- *IPv6 address* - Specifies the IPv6 address of the NTP peer.

This command displays all the NTP servers and peers association information for a specific IP address, as shown in the following example.

```
Brocade# show ntp association detail 10.99.40.1
10.99.40.1 configured server, candidate,  stratum 3
 ref ID 10.45.57.38, time d288de7d.690ca5c7 (10:33:33.1762436551 Pacific Tue Dec 06 2011)
 our mode client, peer mode server, our poll intvl 10, peer poll intvl 10,
 root delay 0.02618408 msec, root disp 0.10108947, reach 3, root dist 0.23610585
 delay 0.92163588 msec, offset 60.77749188 msec, dispersion 70.33842156,
 precision 2**-16, version 4
 org time d288defa.b260a71f (10:35:38.2992678687 Pacific Tue Dec 06 2011)
 rcv time d288defa.a2efbd41 (10:35:38.2733620545 Pacific Tue Dec 06 2011)
 xmt time d288defa.a2ae54f8 (10:35:38.2729334008 Pacific Tue Dec 06 2011)
 filter delay        0.000    6.7770    6.7773    6.7711    6.7720    6.7736    6.7700    0.9921
 filter offset      0.000   19.0047   19.1145   19.2245   19.3313   17.4410   15.4463   60.7777
 filter disp    16000.000   16.0005   15.9975   15.9945   15.9915   15.8885   15.8855    0.0030
 filter epoch       55683     55683     55685     55687     55689     55691     55693     56748
```

show ntp statistics

Syntax: show ntp statistics

This command displays NTP statistics, as shown in the following example.

```
Brocade# show ntp statistics
NTP statistics:
   Number of NTP packets recieved        = 139
   Number of NTP packets sent            = 239
   Number of NTP packets processed       = 130
   Number of NTP packets with bad format = 0
   Number of NTPv3 packets recieved      = 0
   Number of NTP packets with bad auth   = 0
   Number of NTP packets restricted      = 0
   Number of NTP KoD packets sent        = 0
   Number of NTP packets declined        = 0
```

show ntp statistics peer

Syntax: show ntp statistics peer

This command displays NTP server and peer statistics, as shown in the following example.

```
Brocade# show ntp statistics peer
Peer: 10.216.1.101
   Number of NTP packets received     = 32
   Number of NTP packets sent         = 32
   Number of NTP packets processed    = 32
   Number of NTP packets failed tests = 0
   Number of duplicate NTP packets    = 0
   Number of bogus NTP packets        = 0
   Number of bad length NTP packets   = 0
   Number of RATE KoD NTP packets     = 0
   The last packet received 77 seconds ago
   The last peer association struct updated 71 seconds ago
```

NTP debug commands

This section describes the debug commands that generate NTP information.

debug ip ntp

Syntax: [**no**] **debug ip ntp** [**algorithms** | **association** | **broadcast** | **clockadjust** | **errors** | **packet** | **server**]

- **algorithms** - Displays information about the NTP system algorithms.
- **association** - Displays information about the NTP server and peer association.
- **broadcast** - Displays information about the NTP broadcast server and client.
- **clockadjust** - Displays information about the NTP clock-adjust process.
- **errors** - Displays information about the NTP error events.
- **packet** - Displays information about the NTP input and output packets.
- **server** - Displays information about the NTP server.

debug ip ntp algorithms

Syntax: [**no**] **debug ip ntp algorithms**

This command displays information about the NTP system algorithms. Command output resembles the following example.

```
Brocade# debug ip ntp algorithms
Oct 19 18:18:08 NTP: ntp_clock_select: final survivor 10.45.57.38
Oct 19 18:18:08 NTP: ntp_clock_select: final survivor 10.167.160.102
Oct 19 18:18:08 NTP: ntp_clock_select: final survivor 10.164.222.108
Oct 19 18:18:08 NTP: ntp_clock_select: final survivor 10.189.94.4
Oct 19 18:18:08 NTP: ntp_clock_select: final survivor 10.20.99.139
Oct 19 18:18:08 NTP: ntp_clock_select: number of final survivors 5 and leap vote 0
Oct 19 18:18:08 NTP: ntp_clock_select: combine offset -0.00029317 jitter
0.01741329
Oct 19 18:18:11 NTP: ntp_clock_filter: Adding offset -0.00693410, delay
0.05996580, disp 0.00001627 to filter[4] for peer 10.167.160.102
```

debug ip ntp association

Syntax: [**no**] **debug ip ntp association**

This command displays information about the NTP server and peer association. Command output resembles the following example.

```
Brocade# debug ip ntp association
Oct 19 18:24:41 NTP: peer_clear: peer 10.167.160.102 next 1179 refid INIT
Oct 19 18:24:41 NTP: newpeer: 10.167.160.102  mode client vers 4 poll 6 10 key
00000000
```

debug ip ntp broadcast

Syntax: [no] debug ip ntp broadcast

This command displays information about the NTP broadcast server and client. Command output resembles the following example.

```
Brocade# debug ip ntp broadcast
Oct 19 18:32:46 NTP: ntp_timer: interface mgmt1 is up, we may send broadcast
packet
Oct 19 18:32:49 NTP: Sending NTP broadcast packet to subnet 10.20.111.255 via port
mgmt1
Oct 19 18:33:56 NTP: Sending NTP broadcast packet to subnet 10.20.111.255 via port
mgmt1
```

debug ip ntp clockadjust

Syntax: [no] debug ip ntp clockadjust

This command displays information about the NTP clock-adjust process. Command output resembles the following example.

```
Brocade# debug ip ntp clockadjust
Oct 19 18:34:33 NTP: ntp_clock_update: at 1767 sample 1571 associd 5
Oct 19 18:34:33 NTP: ntp_local_clock: hufbuf - ptr 2 mindly 0.00291813 huffpuff
correction 0.00419663
Oct 19 18:34:33 NTP: ntp_local_clock: clk offset -0.00287231 clk jit 0.01699589
clk stab 0.13831413 sys_poll 7
Oct 19 18:34:34 NTP: ntp_set_freq: drift 0.00000047, old freq 24999934
Oct 19 18:34:34 NTP: ntp_set_freq: new freq 24999923
Oct 19 18:34:34 NTP: ntp_adj_host_clock: new offset -0.00287231, freq 24999923
Oct 19 18:34:34 NTP: Adjusting the clock. offset -0.00287231, calib used 63564
```

debug ip ntp errors

Syntax: [no] debug ip ntp errors

This command displays information about the NTP error events.

debug ip ntp packet

Syntax: [no] debug ip ntp packet

This command displays information about the NTP input and output packets. Command output resembles the following example.

```
Brocade# debug ip ntp packet
Oct 19 18:37:49 NTP: Sending the NTP client packet to 10.167.160.102 port 123 via
port id Inv
Oct 19 18:37:49         Leap 0, Version 4, Mode client, Startum 3, Poll 7,
       Precision 2**-16, Root delay 1654, Root disp 4452, Ref Id 10.45.57.38,
       Ref time 3528038073.300212530 (18:34:33.300212530 GMT+00 Wed Oct 19 2011)
       Org 3528038138.260935663 (18:35:38.260935663 GMT+00 Wed Oct 19 2011)
       Rec 3528038138.497315593 (18:35:38.497315593 GMT+00 Wed Oct 19 2011)
```

```
       Xmt 3528038269.241951685 (18:37:49.241951685 GMT+00 Wed Oct 19 2011) pkt
len = 48 key 0

Oct 19 18:37:49 NTP: Received NTP server packet from 10.167.160.102 on port 123
via port id mgmt1 at 18:37:49.500184983 GMT+00 Wed Oct 19 2011
Oct 19 18:37:49         Leap 0, Version 4, Mode server, Startum 2, Poll 7,
       Precision 2**-23, Root delay 533, Root disp 2382, Ref Id 10.9.54.119,
       Ref time 3528037311.842989044 (18:21:51.842989044 GMT+00 Wed Oct 19 2011)
       Org 3528038269.241951685 (18:37:49.241951685 GMT+00 Wed Oct 19 2011)
       Rec 3528038269.260440083 (18:37:49.260440083 GMT+00 Wed Oct 19 2011)
       Xmt 3528038269.260707651 (18:37:49.260707651 GMT+00 Wed Oct 19 2011) pkt
len = 48 key 0
```

debug ip ntp server

Syntax: [no] debug ip ntp server

This command displays information about the NTP server. Command output resembles the following example.

```
Brocade# debug ip ntp server
Oct 19 18:42:09 NTP: poll_update: for peer 10.167.160.102 hpoll 7 burst 0 retry 0
throttle 62 next poll 135
Oct 19 18:42:09 NTP: Received NTP server packet from 10.167.160.102 on port 123
via port id mgmt1 at 18:42:09.503013486 GMT+00 Wed Oct 19 2011
Oct 19 18:42:09         Leap 0, Version 4, Mode server, Startum 2, Poll 7,
       Precision 2**-23, Root delay 533, Root disp 2638, Ref Id 10.9.54.119,
       Ref time 3528037311.842989044 (18:21:51.842989044 GMT+00 Wed Oct 19 2011)
       Org 3528038529.245415329 (18:42:09.245415329 GMT+00 Wed Oct 19 2011)
       Rec 3528038529.259664500 (18:42:09.259664500 GMT+00 Wed Oct 19 2011)
       Xmt 3528038529.260145073 (18:42:09.260145073 GMT+00 Wed Oct 19 2011) pkt
len = 48 key 0
```

IP security

The debug commands in this section apply to Internet Protocol security (IPsec) operation for OSPFv3. You must use them only if you suspect that something is not working correctly.

IPsec debug commands

This section describes the IPsec-related debug commands.

debug ipsec esp

Syntax: [no] debug ipsec esp

This command turns on debugging of Encapsulating Security Payload (ESP). After key roll-over finishes (if it is in process), the ESP debugging facility can show error data (if it exists). Note that the first row of information shows the SPI in hexadecimal format (0x1d97c), and this value is the equivalent of decimal 121212.

```
Brocade# debug ipsec esp
          IPSec:  esp debugging is on
Brocade# show ipsec esp
Dec 12 11:37:39 IPSEC,ESP: decrypt ok, seq=0  (SA: ESP  in spi=0x1d97c
dst=FE80::)
psec sa
             IPSEC Security Association Database(Entries:2)
SPDID Dir Encap SPI        Destination          AuthAlg  EncryptAlg
8     out ESP   121212     ::                   sha1     Null
8     in  ESP   121212     FE80::               sha1     Null
Dec 12 11:37:49 IPSEC,ESP: decrypt ok, seq=0  (SA: ESP  in spi=0x1d97c
dst=FE80::)
Dec 12 11:37:59 IPSEC,ESP: decrypt ok, seq=0  (SA: ESP  in spi=0x1d97c
dst=FE80::)
Dec 12 11:38:08 IPSEC,ESP: decrypt ok, seq=0  (SA: ESP  in spi=0x1d97c
dst=FE80::)
Dec 12 11:38:18 IPSEC,ESP: decrypt ok, seq=0  (SA: ESP  in spi=0x1d97c
dst=FE80::)
Dec 12 11:38:30 IPSEC,ESP: decrypt ok, seq=0  (SA: ESP  in spi=0x1d97c
dst=FE80::)
Dec 12 11:38:40 IPSEC,ESP: decrypt ok, seq=0  (SA: ESP  in spi=0x1d97c
dst=FE80::)
```

debug ipsec sa

Syntax: [no] debug ipsec sa

This command enables the display of debugging information related to the security associations used by IPsec for OSPFv3 packets.

```
Brocade# debug ipsec sa
          IPSec:  sa debugging is on
Brocade(config)# interface ethernet 1/8
Brocade(config-if-e1000-1/8) # ipv6 ospf authentication ipsec spi 121212 esp sha1
no-encrypt 1234567890123456789012345678901234567890
Brocade(config-if-e1000-1/8) #Dec 12 11:45:37 IPSEC,SA: ipipsec_pfkeyv2_input()
:: receiving 'ADD' command
Dec 12 11:45:37 IPSEC,SA: Adding SA: ESP  in spi=0x1d97c dst=FE80::, replay=0
Dec 12 11:45:37 IPSEC,SA: ipipsec_pfkeyv2_input() :: succeeded
Dec 12 11:45:37 IPSEC,SA: ipipsec_pfkeyv2_input() :: receiving 'X_ADDFLOW'
command
Dec 12 11:45:37 IPSEC,SA: ipipsec_pfkeyv2_input() :: succeeded
Dec 12 11:45:47 IPSEC,SA: ipipsec_pfkeyv2_input() :: receiving 'ADD' command
Dec 12 11:45:47 IPSEC,SA: Adding SA: ESP  out spi=0x1d97c dst=::, replay=0
Dec 12 11:45:47 IPSEC,SA: ipipsec_pfkeyv2_input() :: succeeded
Dec 12 11:45:47 IPSEC,SA: ipipsec_pfkeyv2_input() :: receiving 'X_ADDFLOW'
command
Dec 12 11:45:47 IPSEC,SA: ipipsec_pfkeyv2_input() :: succeeded
```

debug ipsec policy

Syntax: [no] debug ipsec policy

This command enables the display of debugging information for IPsec policy.

```
Brocade# debug ipsec policy
          IPSec:  policy debugging is on
Brocade(config)# interface ethernet 1/8
Brocade(config-if-e1000-1/8)# ipv6 ospf authentication ipsec spi 121212 esp sha1
no-encrypt 1234567890123456789012345678901234567890
Brocade(config-if-e1000-1/8)# Dec 12 11:47:45 IPSEC,Policy: Creating flow [input
use 'prot=OSPF src=FE80::/10:0 dst=::/0:0' -> SA: ESP  in spi=0x1d97c dst=FE80::]
: ok
Brocade(config-if-e1000-1/8)#Dec 12 11:47:55 IPSEC,Policy: Creating flow [output
use 'prot=OSPF src=FE80::/10:0 dst=::/0:0' -> SA: ESP  out spi=0x1d97c dst=::] :
ok
```

debug ipsec in

Syntax: [no] debug ipsec in

This command enables the display of debugging information related to inbound OSPFv3 packets with IPsec.

```
Brocade# debug ipsec in
          IPSec:  in debugging is on
Dec 12 11:39:49 IPSEC,IN:  ESP  spi=121212 (pkt 'ESP  FE80:: -> FE80::')
payloadlength =64
Dec 12 11:39:49 IPSEC,IN: Incoming packet matches Policy :  input use 'prot=OSPF
src=FE80::/10:0 dst=::/0:0' -> SA: ESP  in spi=0x1d97c dst=FE80::
12 11:39:59 IPSEC,IN:  ESP  spi=121212 (pkt 'ESP  FE80:: -> FE80::') payloadlength
=64
Dec 12 11:39:59 IPSEC,IN: Incoming packet matches Policy :  input use 'prot=OSPF
src=FE80::/10:0 dst=::/0:0' -> SA: ESP  in spi=0x1d97c dst=FE80::
sec po
             IPSEC Security Policy Database(Entries:2)
PType  Dir Proto Source(Prefix:TCP/UDP Port)    Destination(Prefix:TCP/UDPPort)
                    SA: SPDID Dir Encap SPI       Destination
use    in  OSPF  FE80::/10:any                  ::/0:any
                    SA: 8     in  ESP   121212     FE80::
use    out OSPF  FE80::/10:any                  ::/0:any
                    SA: 8     out ESP   121212     ::
```

debug ipsec out

Syntax: [no] debug ipsec out

This command enables the display of debugging information related to outbound OSPFv3 packets with IPsec.

```
Brocade# debug ipsec out
IPSEC,OUT:    Matching Flow:  output use 'prot=OSPF src=FE80::/1
0:0 dst=::/0:0' -> SA: ESP  out spi=0x258 dst=::
IPSEC,OUT: SA ESP  out spi=0x258 dst=:: payloadlength =60
IPSEC,OUT:    Matching Flow:  output use 'prot=OSPF src=FE80::/1
0:0 dst=::/0:0' -> SA: ESP  out spi=0x14a dst=::
IPSEC,OUT: SA ESP  out spi=0x14a dst=:: payloadlength =60
    IPSEC,OUT: OSPF FE80::1 -> FE80::1, payloadlength =40
```

debug ipv6 ospf ipsec

Syntax: [no] debug ipv6 ospf ipsec

This command displays information about IPsec events. This command can show if IPsec is actually providing its services to IPv6 OSPFv3. The example output shows success in the attempts to provide various IPsec services to OSPFv3.

```
Brocade# debug ipv6 ospf ipsec
Brocade(config-if-e1000-1/8)# ipv6 ospf authentication ipsec spi 121212 esp sha1
no-encrypt 1234567890123456789012345678901234567890
Brocade(config-if-e1000-1/8)# Dec 12 11:53:24 OSPFv3: ITC_AUTHENTICATION_CONFIG
message received
Dec 12 11:53:24 OSPF6: Sending request to IPSEC to ADD Inbound SA for SA with
SPI=121212 SPDID=8
Dec 12 11:53:24 OSPFv3: IPSEC ADD Inbound SA SUCCESS for SA with SPI=121212,
SPDID=8!
Dec 12 11:53:24 OSPF6: Sending request to IPSEC to ADD Inbound Policy with
SPI=121212
Dec 12 11:53:24 OSPFv3: IPSEC ADD Inbound Policy SUCCESS for SA with SPI=121212,
SPDID=8!
Dec 12 11:53:24 OSPF6: Auth timer started
Dec 12 11:53:24 OSPFv3: Key Rollover, for 1/8, state change NOT_ACTIVE->STARTED
Dec 12 11:53:34 OSPFv3: Key Rollover, for 1/8, state change STARTED->IN-PROGRESS
Dec 12 11:53:34 OSPF6: Sending request to IPSEC to ADD Outbound SA for SA with
SPI=121212 SPDID=8
Dec 12 11:53:34 OSPFv3: IPSEC ADD Outbound SA SUCCESS for SA with SPI=121212,
SPDID=8!
Dec 12 11:53:34 OSPF6: Sending request to IPSEC to ADD Outbound Policy with
SPI=121212
Dec 12 11:53:34 OSPFv3: IPSEC ADD Outbound Policy SUCCESS for SA with SPI=121212,
SPDID=8!
Dec 12 11:53:44 OSPFv3: Key Rollover, for 1/8, state change
IN-PROGRESS->NOT_ACTIVE
Dec 12 11:53:44 OSPF6: Auth timer stopped
```

Chapter

Forwarding Diagnostics 10

In this chapter

This chapter describes diagnostics for forwarding protocols and environments on Brocade NetIron XMR series and Brocade MLX series routers.

ARP

Address Resolution Protocol (ARP) is a standard protocol that enables a router to obtain the MAC address of an interface on another device when the router knows the IP address of the interface. ARP is enabled by default and cannot be disabled.

ARP show commands

This section describes the show commands that display ARP information.

show ip arp-inspection

Syntax: **show ip arp-inspection** [**vlan** *vlan_id*]

This command displays the ARP inspection status and the trusted and untrusted ports in a VLAN. Command output resembles the following example.

```
Brocade# show ip arp-inspection
ARP inspected VLANs:
1000

ARP inspection trusted ports:
ethe 2/1
```

show ip static-arp

Syntax: **show ip static-arp**

This command displays the ARP inspection table. Command output resembles the following example.

```
Brocade# show ip static-arp
Total no. of entries: 2
  Index   IP Address          MAC Address         Port/            VLAN  ESI
                                                  Vpls-Vlan:Port/
                                                  Vpls-Peer
  1       10.10.10.10         0000.0033.4444                       100
  2       10.11.11.11         0000.0066.7777      4/1
  3       10.12.12.12         0000.0023.4343      *:10.2.3.105
```

ARP debug commands

For ARP debug commands, refer to "debug ip arp" on page 82.

Configuration notes

To delete the static MAC entry, you must delete the static ARP entry first.

ECMP

Equal-Cost MultiPath (ECMP) supports traffic that is forwarded in software. ECMP applies only to traffic forwarded by software, not to traffic forwarded by hardware. Normally, traffic is forwarded in software when you configure a CPU-based feature such as ACLs, rate limiting, or NetFlow Switching. Traffic also is forwarded by software if the CAM (used for hardware-forwarding) becomes full.

ECMP show commands

This section describes the show commands that display ECMP information.

show ipv6

Syntax: **show ipv6**

This command displays the status of ECMP load sharing for IPv6, as shown in the following example.

```
Brocade# show ipv6
Global Settings

  unicast-routing enabled, hop-limit 64
  No Inbound Access List Set
  No Outbound Access List Set
  Prefix-based IPv6 Load-sharing is Enabled, Number of load share paths: 4
```

show ipv6 cache

Syntax: **show ipv6 cache** [*index-number* | *ipv6-prefix/prefix-length* | *ipv6-address* | **ethernet** *slot/port* | **pos** *slot/port* | **tunnel** *number* | **ve** *number*]

This command displays all the entries in the IPv6 forwarding cache, or those specified using the syntax variables.

- *index-number* - Displays the IPv6 cache for the specific entry index.
- *ipv6-prefix/prefix-length* - Displays the IPv6 cache for the specific IPv6 prefix.
- *ipv6-address* - Displays the IPv6 cache for the specific IPv6 address.
- **ethernet** *slot/port* - Displays the IPv6 cache for the specific Ethernet interface.
- **pos** *slot/port* - Displays the IPv6 cache for the specific POS interface.
- **tunnel** *number* -Displays the IPv6 cache for the specific tunnel interface.
- **ve** *number* - Displays the IPv6 cache for the specific virtual Ethernet interface.

```
Brocade# show ipv6 cache
Total number of cache entries: 10
    IPv6 Address                       Next Hop                     Port
1   2001:DB8:2::2                      LOCAL                        tunnel 2
2   2000:4::106                        LOCAL                        ethe 2
3   2001:DB8:4::110                    DIRECT                       ethe 2
4   2001:DB8:c0a8:46a::1               LOCAL                        ethe 2
5   fe80::2e0:52ff:fe99:9737           LOCAL                        ethe 2
6   fe80::ffff:ffff:feff:ffff          LOCAL                        loopback 2
7   fe80::c0a8:46a                     LOCAL                        tunnel 2
8   fe80::c0a8:46a                     LOCAL                        tunnel 6
9   2001:DB8::1                        LOCAL                        loopback 2
10  fe80::2e0:52ff:fe99:9700           LOCAL                        ethe 1
```

ECMP debug commands

There are no debug commands specific to ECMP.

Multicast VRF

The provider edge (PE) router maintains a Virtual Routing and Forwarding table (VRF) for each customer that is attached to it through a customer edge (CE) router. The VRF contains routes between the PE and the CE and Label Switched Paths (LSPs) across the MPLS domain for each PE that is a member of the customer's VPN. VRFs are defined on interfaces of the PE.

Multicast VRF show commands

This section describes the show commands that display multicast VRF information.

show ip bgp vpnv4 neighbor flap-statistics

Syntax: **show ip bgp vpnv4** *vrf-name* **neighbor** *ip-addr* **flap-statistics**

- *vrf-name* - Specifies the VPNv4 neighbor for which you want to display flap statistics.
- *ip-addr* - Specifies a particular route.

This command displays flap statistics for routes learned from the specified VRF neighbor, as shown in the following example.

```
Brocade# show ip bgp vpnv4 neighbor 10.2.2.2 flap-statistics
Total number of flapping routes: 0
```

This output shows the total number of routes in the Layer 3 switch BGP4 route table that have changed state and have been marked as flapping routes.

show ip pim prune

Syntax: **show ip pim** [**vrf** *vrf-name*] **prune**

- **vrf** *vrf-name* - Specifies the VRF name for which you want to display multicast cache entries.

This command displays all multicast cache entries that are currently in a pruned state and have not yet aged out, as shown in the following example.

```
Brocade# show ip pim vrf med prune
Index Port    PhyPort SourceNet       Group           Nbr             Age
                                                                      sec
    1 v12     1/1     10.47.2.10     228.172.0.77    0.0.0.0         40
    2 v12     1/1     10.47.2.10     228.172.0.73    0.0.0.0         40
    3 v12     1/1     10.47.2.10     228.172.0.69    0.0.0.0         40
    4 v12     1/1     10.47.2.10     228.172.0.65    0.0.0.0         40
    5 v12     1/1     10.47.2.10     228.172.0.61    0.0.0.0         40
    6 v12     1/1     10.47.2.10     228.172.0.57    0.0.0.0         40
    7 v12     1/1     10.47.2.10     228.172.0.53    0.0.0.0         40
    8 v12     1/1     10.47.2.10     228.172.0.49    0.0.0.0         40
    9 v12     1/1     10.47.2.10     228.172.0.45    0.0.0.0         40
   --------------------------------------
Total Prune entries: 9
```

show ip pim group

Syntax: **show ip pim vrf** *vrf-name*] **group**

This command displays PIM group information for an entire PIM group, or for the VRF instance identified by the **vrf** *vrf-name* option. Command output resembles the following example.

```
Brocade# show ip pim group

Total number of Groups: 2
Index 1         Group 239.255.162.1       Ports e3/11
```

Multicast VRF debug commands

This section describes the multicast VRF-related debug commands.

debug ip vrf

Syntax: [**no**] **debug ip vrf**

This command generates information about synchronization of VRF routing information to line cards, as shown in the following example.

```
Brocade# debug ip vrf
RTM (vrf): Processing tree download for vrf 1
```

This is a download request from the line card to start the tree download for VRF 1.

debug ip bgp all-vrfs

Syntax: [**no**] **debug ip bgp all-vrfs** [*A.B.C.D* | **dampening** | **events** | **graceful-restart** | **keepalives** | **updates**]

- *A.B.C.D* - Displays information about a BGP neighbor address.
- **dampening** - Displays information about BGP dampening.
- **events** - Displays information about BGP events.
- **graceful-restart** - Displays information about graceful restart events.
- **keepalives** - Displays information about BGP keepalives.
- **updates** - Displays information about BGP updates.

This command displays information about BGP activity, with output that is limited to VRF events.

debug ip ospf all-vrfs

Syntax: [**no**] **debug ip ospf all-vrfs** [*A.B.C.D* | **adj** | **bfd** | **error** | **events** | **flood** | **graceful-restart** | **log-debug-message** | **log-empty-lsa** | **lsa-generation** | **max-metric** | **packet** | **retransmission** | **route** | **sham-link** | **shortcuts** | **spf**]

- *A.B.C.D* - Display OSPF information for a specific IP address.
- **adj** - Displays OSPF adjacency events.
- **bfd** - Displays OSPF BFD events.
- **error** - Displays possible OSPF error in run time.
- **events** - Displays OSPF events.
- **flood** - Displays OSPF flooding information.
- **graceful-restart** - Displays OSPF graceful restart events.
- **log-debug-message** - Enables OSPF debug message logging.
- **log-empty-lsa** - Enables OSPF empty LSA logging.
- **lsa-generation** - Enables OSPF LSA generation.
- **max-metric** - Displays information about OSPF Stub Router Advertisement.
- **packet** - Displays information about OSPF packets.
- **retransmission** - Displays OSPF retransmission events.
- **route** - Displays information about OSPF routes.
- **sham-link** - Displays OSPF sham-link traces.
- **shortcuts** - Displays OSPF shortcuts.
- **spf** - Displays OSPF SPF traces.

This command displays OSPF information for all VRF activity. Command output resembles the following example.

```
Brocade# debug ip ospf all-vrfs adj
         OSPF:  adjacency events debugging is on
Dec 10 20:41:09 OSPF: rcvd hello from 10.101.1.2 area 0 on interface 10.101.1.1,
state BackupDR, DR 10.101.1.2, BDR 10.101.1.1
Dec 10 20:41:09 OSPF: Neighbor 10.101.1.2, int v501,  state FULL processing event
HELLO_RECEIVED
Dec 10 20:41:14 OSPF: send hello on area 0 interface 10.13.13.13
```

```
Dec 10 20:41:15 OSPF: rcvd hello from 10.1.1.2 area 0 on interface 10.1.1.1, state
BackupDR, DR 10.1.1.2, BDR 10.1.1.1
Dec 10 20:41:15 OSPF: Neighbor 10.1.1.2, int 1/1,  state FULL processing event
HELLO_RECEIVED
Dec 10 20:41:15 OSPF: send hello on area 0 interface 10.1.1.1
Dec 10 20:41:17 OSPF: send hello on area 0 interface 10.101.1.1
Dec 10 20:41:20 OSPF: rcvd hello from 10.101.1.2 area 0 on interface 10.101.1.1,
state BackupDR, DR 10.101.1.2, BDR 10.101.1.1
Dec 10 20:41:20 OSPF: Neighbor 10.101.1.2, int v501,  state FULL processing event
HELLO_RECEIVED
```

Configuration notes

You must configure a VRF on an interface before configuring a Virtual Router (VRRP-E) on it. If you enable the Virtual Router before you enable the VRF, the Virtual Router configuration will be deleted.

Trunking

Trunk groups are manually-configured aggregate links containing multiple ports. Trunk groups enable load sharing of traffic, and they also provide redundant, alternate paths for traffic if any of the segments fail.

Trunking show commands

Trunk group configuration information can be displayed using the same command for either a server or switch trunk. The information is displayed in two sections: configured trunks and operational trunks.

show lag error counters id

Syntax: show lag error counters id *LAG-id*

This command displays error counters for the configured Link Aggregation Group (LAG). Command output resembles the following example.

```
Brocade# show lag error counters id 2
LAG Id: 2
LAG ITC Error Count = 1
LAG IPC Error Count = 2
LAG FID Error Count = 0
LAG All ports down Error Count = 0
```

show lag error debug id

Syntax: show lag error debug id *LAG-id*

This command displays the reason code for why a LAG or some of the ports within a LAG are in down states. Command output resembles the following example.

```
Brocade# show lag error debug id 1
LAG Id:   1
Type: Dynamic
LAG Ports Down and Reason:
e2/1:Physically down
```

```
e2/2:Administratively Down
e2/3:Threshold reached
e2/11:Timeout
e2/12:Peer is out of sync
```

show lag ethernet

Syntax: show lag ethernet *slotnum/portnum* **to** *slotnum/portnum*

This command displays information about trunk groups, and is divided into sections for configured trunks and operational trunks. A configured trunk group is one that has not yet been activated.

To display server trunk group information for a *range of ports*, enter a command similar to the following example.

```
Brocade# show lag ethernet 12/1 to 12/3
Max number of trunks: 128
available: 127
Configured number of server trunks: 1
Configured trunks:
Trunk ID: 1
Type: Server
Ports_Configured: 3
Base FID: 0x0400
FID count: 16
Ports         12/1    12/2      12/3
Port Names    none    none      none
Port_Status   enable  enable    enable
Operational trunks:
Trunk ID: 1
Type: Server
Duplex: Full
Speed: 1G
Tag: No
Priority: level0
Active Ports: 3
Ports                12/1    12/2    12/3
Link_Status          active  active  active
```

To display switch trunk group information for *specific* ports, enter a command similar to the following example.

```
Brocade# show lag ethernet 9/1 to 9/2
Max number 206 (128 server trunks, 78 switch trunks)
Number of hash buckets per server trunk: 256
Configured number of server trunks: 0
Configured trunks:
Trunk ID: 66
Type: Switch
Ports_Configured: 2
Ports         9/1     9/2
Port Names    none    none
Port_Status   enable  enable

Operational trunks:
Trunk ID: 66
Type: Switch
Duplex: Full
Speed: 10G
Tag: No
Priority: level0
```

```
Active Ports: 1
Ports                9/1    9/2
Link_Status          active down
Load Sharing
 Mac Address         0      0
 IP                  0      0
 Multicast           0      0
 PBR                 0      0
```

show trunk

Syntax: show trunk

This command displays trunk or LAG information on LP such as the type of trunk, trunk FID, number of maximum ports per trunk, configured ports and its status, and so on.

Command output resembles the following example.

```
Brocade# show trunk
Trunk ID: 1
Type: Server(base_fid=0780, base_oar_fid=0780, 100G LAG shadow_fid = 07C0)
Number of FIDs = 4
3/1       3/2
active active
Active primary: 3/1
Active Port Count: Sw 2, Hw(ppcr, val): (3:1, 2) (3:2, 2)
```

Trunking debug commands

There are no debug commands specific to trunking.

Configuration notes

There are several trunk group rules. For a full description of these trunk rules, refer to the *Brocade MLX Series and NetIron Family Configuration Guide.*

The following items must be considered when configuring trunk groups:

- You can use both static trunk groups and 802.3ad trunking on the same device. However, you can use only one type of trunking for a given port. For example, you can configure port 1/1 as a member of a static trunk group or you can enable 802.3ad link aggregation on the port, but you cannot do both.
- The ports in a trunk group make a single logical link. Therefore, all the ports in a trunk group must be connected to the same device at the other end.
- A trunk threshold must be configured on only one end of the trunk. If a threshold is set on both sides, link failures will result in race-conditions and the trunk will not function properly.
- If you connect physical cables before configuring the trunk groups and then reboot, traffic on the ports can create a spanning tree loop.

Common diagnostic scenarios

Trunk transaction failed; ports overlap with other trunks.

With a static trunk, you must first remove the existing trunk and reconfigure a new one. If you are using dynamic trunk configuration, you would be able to add a port dynamically in the trunk.

MCT

Multi-Chassis Trunking (MCT) provides switch-level redundancy. If one of the switches goes down, then the LAG is still up and the traffic flows without any disruption.

MCT show commands

This section describes the show command that displays MCT information.

show cluster

Syntax: show cluster

This command displays the complete cluster information on the ICL peer and clients, as shown in the following example.

```
Brocade# show cluster
Cluster abc 1
=============
Rbridge Id: 100, Session Vlan: 4090
Cluster State: Deploy
Clients State: All client ports are administratively disabled [Optional]
Client Isolation Mode: Strict [Optional]
Configured Member Vlan Range: 20 to 30
Active Member Vlan Range: 20
ICL Info:
---------
Name Port Trunk
icl1 1/1 -
Peer Info:
----------
Peer IP: 10.10.10.2, Peer Rbridge Id: 200, ICL: icl1
KeepAlive Interval: 20 , Hold Time: 60, Fast Failover
Active Vlan Range: 20
Peer State: CCP Up (Up Time: 0 days: 0 hr:15 min:44 sec)
Client Info:
------------
Name Rbridge-id Config Port Trunk FSM-State
c1 300 Deployed 1/13 - Up
```

The following command output displays the complete cluster information on the ICL peer and clients for Layer 2 Virtual Private Network (L2VPN) MCT.

```
Brocade# show cluster
Cluster clu 1
=============
Rbridge Id: 4, Session Vlan: 0
Cluster State: Deploy
Client Isolation Mode: Loose
Configured Member Vlan Range:
Active Member Vlan Range:

L2VPN Peer Info:
---------------
Peer IP: 10.5.5.5, Peer Rbridge Id: 5
KeepAlive Interval: 300 , Hold Time: 900
Node KeepAlive Interval: 2000 , Hold Time: 6000
l2vpn-revertible-timer 300
Peer State: CCP Up (Up Time:   0 days: 0 hr:21 min:52 sec)
```

```
Client Info:
------------
Name           Rbridge-id Config     Port  Trunk FSM-State
c1             101        Deployed   1/4     2   Admin Up
c2             102        Deployed   1/2     1   Up
```

MCT debug commands

This section describes the MCT-related debug commands.

debug cluster forwarding

Syntax: [no] debug cluster forwarding

This command displays all the MCT forwarding-related events or messages in the MP that can affect traffic forwarding. Some examples include remote CCEP status changes, MCT FID creation, FID updates, and so on.

Command output resembles the following example.

```
Brocade# debug cluster forwarding
CLUSTER FORWARDING: MCT CCEP control FID 0xa006 for vlan 101 - REMOVE CCEP port
1/1
CLUSTER FORWARDING: Processing remote CCEP UP event for port eth 1/1
CLUSTER FORWARDING: Processed remote CCEP event for port 1/1 for 100 vlans
CLUSTER FORWARDING: MCT CCEP control FID 0xa006 for vlan 101 - ADD CCEP port 1/1
CLUSTER FORWARDING: Processing remote CCEP DOWN event for port eth 1/1
```

debug cluster actions

Syntax: [no] debug cluster actions

This command displays the debug messages for MCT-related events, such as MCT FID port changes, and so on.

Command output resembles the following example.

```
Brocade# debug cluster actions
CLUSTER ACTIONS: Mac learning disabled for MAC 0000.0022.0001 on ICL  4/2, vlan
101
CLUSTER ACTIONS: Received remote CCEP UP IPC
```

debug cluster active-passive

Syntax: [no] debug cluster active-passive

This command displays the MCT active-passive messages for the cluster.

Command output resembles the following example.

```
Brocade# debug cluster active-passive
CLUSTER MODE - Update Client Role, Client:client-1 My Client role: 1, Peer Client
Role: 0

CLUSTER MODE - Client: client-1, Client old role: Passive, New elected role:
Active"

Cluster Mode: For Client client-1 Revert Mode timer expired, move to Active
```

```
Cluster Mode: For Client client-1 CCEP is UP, Role Revert Mode is ON
```

debug cluster cam

Syntax: [no] debug cluster cam

This command displays all the CAM- or PRAM-related activities for MCT-related events in the LP.

Command output resembles the following example.

```
Brocade# debug cluster cam
CLUSTER CAM: Added peer interface cam:mac 0000.003b.a200, ppcr 0
CLUSTER CAM: Deleted peer interface cam:mac 0000.003b.a200, ppcr 0
```

debug cluster ipc

Syntax: [no] debug cluster ipc

This command is helpful to debug IPC to LP to sync forwarding information in the LP.

Command output resembles the following example.

```
Brocade# debug cluster ipc
CLUSTER IPC: MCT FID 0xa005 received for vlan 101
CLUSTER IPC: cluster 1, vlan mask change
```

debug cluster config

Syntax: [no] debug cluster config

This command displays information about the Layer 2 Virtual Private Network (L2VPN) cluster configurations.

Command output resembles the following example.

```
Brocade# debug cluster config
Brocade(config)#cluster c1
Nov 23 08:39:03 CLUSTER CONFIG: CLUSTER_CONFIG_ADD_CLUSTER ITC message received
Brocade(config-cluster-c)# rbridge-id 4
Nov 23 08:39:03 CLUSTER CONFIG: CLUSTER_CONFIG_CLUSTER_RBRIDGE_ID ITC message
received
Brocade(config-cluster-c)# session-vlan 31
Nov 23 08:39:03 CLUSTER CONFIG: CLUSTER_CONFIG_CLUSTER_MGMT_VLAN ITC message
received
Brocade(config-cluster-c)# icl icl1 ethernet 1/3
Nov 23 08:39:03 CLUSTER CONFIG: CLUSTER_CONFIG_ADD_CLUSTER_ICL ITC message
received
Brocade(config-cluster-c)# peer 10.31.31.5 rbridge-id 5 icl icl1
Nov 23 08:39:03 CLUSTER CONFIG: CLUSTER_CONFIG_ADD_CLUSTER_PEER ITC message
received
Brocade(config-cluster-c)# l2vpn-peer 10.5.5.5 rbridge-id 5
Nov 23 08:39:03 CLUSTER CONFIG: CLUSTER_CONFIG_L2VPN_PEER_TIMERS ITC message
received
Brocade(config-cluster-c)# deploy
Nov 23 08:39:03 CLUSTER CONFIG: CLUSTER_CONFIG_CLUSTER_DEPLOY ITC message
received
Nov 23 08:39:03 CLUSTER CONFIG: sending EVENT_ID_CLUSTER_DEPLOY event cluster 1
Brocade(config-cluster-c)# client c1
Nov 23 08:39:03 CLUSTER CONFIG: CLUSTER_CONFIG_ADD_CLUSTER_CLIENT ITC message
received
Brocade(config-cluster-c-client-c1)#  rbridge-id 101
```

```
Nov 23 08:39:03 CLUSTER CONFIG: CLUSTER_CONFIG_CLIENT_RBRIDGE_ID ITC message
received
Brocade(config-cluster-c-client-c1)#  client-interface ethernet 1/4
Nov 23 08:39:03 CLUSTER CONFIG: CLUSTER_CONFIG_ADD_CLIENT_PORTS ITC message
received
Nov 23 08:39:03  CLUSTER CONFIG: LACP port id for ccep 1/4 is 2051
Brocade(config-cluster-c-client-c1)#  deploy
Nov 23 08:39:03 CLUSTER CONFIG:
CLUSTER_CONFIG_CLIENT_DEPLOYCLUSTER_CONFIG_ADD_CLUSTER_L2VPN_PEER ITC message
received
Nov 23 08:39:03 CLUSTER CONFIG: sending EVENT_ID_CLUSTER_CLIENT_DEPLOY event
cluster 1, client port 3
Brocade(config-cluster-c-client-c1)# client c2
Nov 23 08:39:03 CLUSTER CONFIG: CLUSTER_CONFIG_ADD_CLUSTER_CLIENT ITC message
received
Brocade(config-cluster-c-client-c2)#  rbridge-id 102
Nov 23 08:39:03 CLUSTER CONFIG: CLUSTER_CONFIG_CLIENT_RBRIDGE_ID ITC message
received
Brocade(config-cluster-c-client-c2)#  client-interface ethernet 1/2
Nov 23 08:39:03 CLUSTER CONFIG: CLUSTER_CONFIG_ADD_CLIENT_PORTS ITC message
received
Nov 23 08:39:03  CLUSTER CONFIG: LACP port id for ccep 1/2 is 2049
```

debug cluster ccp fsm

Syntax: [no] debug cluster ccp fsm

This command displays information related to Finite State Machine (FSM) transactions.

Command output resembles the following example.

```
Brocade# debug cluster ccp fsm
Oct 19 14:36:27 CCP: Fsm2 entry 10.5.5.5
Oct 19 14:36:27 CLUSTER CCP: sent an init msg to = 10.5.5.5
Oct 19 14:36:27 CLUSTER CCP: Fsm4 sending keepalive to 10.5.5.5 in state recv

Oct 19 14:36:27 CLUSTER CCP: Fsm6 10.5.5.5 going operational
```

debug cluster ccp packet

Syntax: [no] debug cluster ccp packet

This command displays information specific to L2VPN timer packet exchange operation.

Command output resembles the following example.

```
Brocade# debug cluster ccp packet
Nov 23 08:42:42 CLUSTER CCP PACKET: Rx Keepalive - cluster id 00000001
cluster_rbridge_id 00000005 magic_no 000014ef
Nov 23 08:42:42 CLUSTER CCP PACKET: Rx Keepalive - matching cluster rbridge id
found
Nov 23 08:42:42 CLUSTER CCP PACKET: Rx Keepalive -Received CCP Keepalive
packet to be sent on VE interface - using source mac address 001bed3b 0000a200
Nov 23 08:42:42 CLUSTER CCP PACKET: Tx L2VPN  CCP Keepalive IP 10.5.5.5, vlan 31,
port 1/3
Nov 23 08:42:42 Couldn't get the Area ID for patherr from 10.13.13.5
Nov 23 08:42:42 Couldn't get the Area ID for patherr from 10.13.13.5
Nov 23 08:42:42 CLUSTER CCP PACKET: Rx Keepalive - cluster id 00000001
cluster_rbridge_id 00000005 magic_no 000014ef
Nov 23 08:42:42 CLUSTER CCP PACKET: Rx Keepalive - matching cluster rbridge id
found
```

```
Nov 23 08:42:42 CLUSTER CCP PACKET: Rx Keepalive -Received CCP Keepalive
packet to be sent on VE interface - using source mac address 001bed3b 0000a200
Nov 23 08:42:42 CLUSTER CCP PACKET: Tx L2VPN  CCP Keepalive IP 10.5.5.5, vlan 31,
port 1/3
Nov 23 08:42:42 CLUSTER CCP PACKET: Rx Keepalive - cluster id 00000001
cluster_rbridge_id 00000005 magic_no 000014ef
Nov 23 08:42:42 CLUSTER CCP PACKET: Rx Keepalive - matching cluster rbridge id
found
Nov 23 08:42:42 CLUSTER CCP PACKET: Rx Keepalive -Received CCP Keepalive
packet to be sent on VE interface - using source mac address 001bed3b 0000a200
Nov 23 08:42:42 CLUSTER CCP PACKET: Tx L2VPN  CCP Keepalive IP 10.5.5.5, vlan 31,
port 1/3
Nov 23 08:42:42 CLUSTER CCP PACKET: Rx Keepalive - cluster id 00000001
cluster_rbridge_id 00000005 magic_no 000014ef
Nov 23 08:42:42 CLUSTER CCP PACKET: Rx Keepalive - matching cluster rbridge id
found
Nov 23 08:42:42 CLUSTER CCP PACKET: Rx Keepalive -Received CCP Keepalive
```

debug cluster fsm

Syntax: [no] debug cluster fsm

This command displays FSM information for the cluster.

Command output resembles the following example.

```
Brocade# debug cluster fsm
Nov 23 08:40:27 CLUSTER CONFIG: CLUSTER_CONFIG_CLUSTER_DEPLOY ITC message
received
Nov 23 08:40:27 CLUSTER CONFIG: sending EVENT_ID_CLUSTER_DEPLOY event cluster 1
Nov 23 08:40:27 CLUSTER CONFIG: sending EVENT_ID_CLUSTER_CLIENT_DEPLOY event
cluster 1, client port 3
Nov 23 08:40:27 CLUSTER CONFIG: sending EVENT_ID_CLUSTER_CLIENT_DEPLOY event
cluster 1, client port 1
Nov 23 08:40:29 CLUSTER FSM: cluster id 1, peer rbridge id 5, old state: Init,
event: CCP Up
Nov 23 08:40:29 CLUSTER FSM: CCRR message sent for client for multiple clients
Nov 23 08:40:29 CLUSTER FSM: new state: Loading
Nov 23 08:40:29 CLUSTER FSM: Received CCRR message from peer when CCP is up
Nov 23 08:40:29 CLUSTER FSM: cluster id 1, client id 101, old state: Init, event:
CCP Up
Nov 23 08:40:29 CLUSTER FSM: sending EVENT_ID_MCT_REMOTE_DEPLOY event
Nov 23 08:40:29 CLUSTER FSM: new state: Admin Up, master: TRUE
Nov 23 08:40:29 CLUSTER FSM: cluster id 1, client id 102, old state: Init, event:
CCP Up
Nov 23 08:40:29 CLUSTER FSM: sending EVENT_ID_MCT_REMOTE_DEPLOY event
Nov 23 08:40:29 CLUSTER FSM: new state: Admin Up, master: TRUE
Nov 23 08:40:29 CLUSTER FSM: cluster id 1, client id 102, old state: Admin Up,
event: Remote Up
Nov 23 08:40:29 CLUSTER FSM: sending EVENT_ID_MCT_REMOTE_CCEP_UP event
Nov 23 08:40:29 CLUSTER FSM: CCRR message sent for client 102 (return code : 0,
value = 00660018)
Nov 23 08:40:29 CLUSTER FSM: new state: Remote Up, master: FALSE
Nov 23 08:40:29 CLUSTER FSM: Received Loading-Done message from peer
Nov 23 08:40:29 CLUSTER FSM: cluster id 1, peer rbridge id 5, old state: Loading,
event: Loading Done
Nov 23 08:40:29 CLUSTER FSM: sending EVENT_ID_MCT_CCP_UP event
Nov 23 08:40:29 CLUSTER FSM: new state: CCP Up
Nov 23 08:40:29 CLUSTER FSM: cluster id 1, client id 102, old state: Remote Up,
event: Spoke Down
```

```
Nov 23 08:40:29 CLUSTER FSM: Sending client fsm state to LP, state 1
Nov 23 08:40:29 CLUSTER FSM: CCRR message sent for client 102 (return code : 0,
value = 00660018)
Nov 23 08:40:29 CLUSTER FSM: new state: Remote Up, master: FALSE
Nov 23 08:40:29 CLUSTER L2VPN:Spoke PW event -  Master/Slave selection for client
rbridge 102 Successful (event = 2)
Nov 23 08:40:29 CLUSTER FSM: cluster id 1, client id 102, old state: Remote Up,
event: Spoke Up
Nov 23 08:40:29 CLUSTER FSM: new state: Remote Up, master: FALSE
Nov 23 08:40:29 CLUSTER L2VPN:Spoke PW event -  Master/Slave selection for client
rbridge 102 Successful (event = 1)
Nov 23 08:40:29 CLUSTER FSM: Received CCRR message from peer when CCP is up
Nov 23 08:40:29 CLUSTER FSM: Received CCRR message from peer when CCP is up
```

debug cluster fsm client

Syntax: [no] debug cluster fsm client

This command displays cluster FSM information for the client.

Command output resembles the following example.

```
Brocade# debug cluster fsm client 400
CLUSTER fsm debugging is now ON for client rbridge 400
Brocade(config)#interface ethernet 3/13
Brocade(config-if-e1000-3/13)#disable

SYSLOG: <46>Dec  5 15:55:50 Brocade System: Interface ethernet 3/13, state down -
disabled

SYSLOG: <46>Dec  5 15:55:50 Brocade PORT: 3/13 disabled by operator from console
session.
Brocade(config-if-e1000-3/13)#
SYSLOG: <44>Dec  5 15:55:50 Brocade LACP: ethernet 3/13 state changes from FORWARD
to LACP_BLOCKED

SYSLOG: <46>Dec  5 15:55:50 Brocade LACP: Port 3/13 mux state transition:
aggregate -> not aggregate (reason: peer is out of sync)

SYSLOG: <44>Dec  5 15:55:50 Brocade LACP: ethernet 3/13 state changes from
LACP_BLOCKED to DOWN
Dec  5 15:55:50 CLUSTER FSM: sending EVENT_ID_MCT_LOCAL_CCEP_DOWN event

SYSLOG: <46>Dec  5 15:55:50 Brocade CLUSTER FSM: Cluster MCT-VPLS (Id: 1), client
R4 (RBridge Id: 400) - Local client CCEP down
Dec  5 15:55:50 CLUSTER FSM: cluster id 1, client id 400, old state: Up, event:
Local Down
Dec  5 15:55:50 CLUSTER FSM: Sending client fsm state to LP, state 1
Dec  5 15:55:50 CLUSTER FSM: new state: Remote Up, master: FALSE
Dec  5 15:55:51 CLUSTER FSM: Received CCRR message from peer when CCP is up (value
= 01900016)

SYSLOG: <46>Dec  5 15:55:53 Brocade LACP: Port 3/13 rx state transition: current
-> expired (reason: timeout)
Brocade(config-if-e1000-3/13)#enable

SYSLOG: <46>Dec  5 15:55:55 Brocade PORT: 3/13 enabled by operator from console
session.
Brocade(config-if-e1000-3/13)#
```

```
SYSLOG: <44>Dec  5 15:55:58 Brocade LACP: ethernet 3/13 state changes from DOWN to
LACP_BLOCKED

SYSLOG: <46>Dec  5 15:55:58 Brocade LACP: Port 3/13 rx state transition: defaulted
-> current

SYSLOG: <46>Dec  5 15:55:58 Brocade LACP: Port 3/13 partner port state transition:
not aggregate -> aggregate

SYSLOG: <46>Dec  5 15:56:00 Brocade LACP: Port 3/13 mux state transition: not
aggregate -> aggregate

SYSLOG: <44>Dec  5 15:56:00 Brocade LACP: ethernet 3/13 state changes from
LACP_BLOCKED to FORWARD

SYSLOG: <46>Dec  5 15:56:00 Brocade System: Interface ethernet 3/13, state up
Dec  5 15:56:00 CLUSTER FSM: sending EVENT_ID_MCT_LOCAL_CCEP_UP event

SYSLOG: <46>Dec  5 15:56:00 Brocade CLUSTER FSM: Cluster MCT-VPLS (Id: 1), client
R4 (RBridge Id: 400) - Local client CCEP up
Dec  5 15:56:00 CLUSTER FSM: cluster id 1, client id 400, old state: Remote Up,
event: Local Up
Dec  5 15:56:00 CLUSTER FSM: sending EVENT_ID_MCT_LOCAL_CCEP_UP event
Dec  5 15:56:00 CLUSTER FSM: new state: Preforwarding Remote Up, master: FALSE
Dec  5 15:56:00 CLUSTER FSM: Received CCRR message from peer when CCP is up (value
= 0190001e)
Dec  5 15:56:00 CLUSTER FSM: cluster id 1, client id 400, old state: Preforwarding
Remote Up, event: CCRR Ack Rcvd
Dec  5 15:56:00 CLUSTER FSM: Sending client fsm state to LP, state 2
Dec  5 15:56:00 CLUSTER FSM: new state: Up, master: FALSE
```

debug cluster l2vpn

Syntax: [no] debug cluster l2vpn

This command displays the events specific to L2VPN operation.

Command output resembles the following example.

```
Brocade# debug cluster l2vpn
Oct 19 14:32:40 CLUSTER_L2VPN: clustermgr_cluster_deploy Done for L2VPN cluster
Oct 19 14:32:40 CLUSTER L2VPN: L2VPN task received Cluster Deploy event
Oct 19 14:32:40 CLUSTER L2VPN: L2VPN task received Client Deploy event
Oct 19 14:32:40 CLUSTER L2VPN: L2VPN ading client information for port 1/2
Oct 19 14:32:40 CLUSTER L2VPN: L2VPN task received Client Deploy event
Oct 19 14:32:40 CLUSTER L2VPN: L2VPN ading client information for port 1/4
Oct 19 14:32:40 CLUSTER L2VPN: L2VPN task received Remote CCEP 1/2 is up
Oct 19 14:32:40 CLUSTER L2VPN: L2VPN task received CCP is up
```

debug cluster mdup

Syntax: [no] debug cluster mdup

This command displays information related to MAC Database Update Protocol (MDUP) operation.

Command output resembles the following example.

```
Brocade# debug cluster mdup
CLUSTER mdup debugging is now ON
```

```
Brocade# Dec  5 16:04:35 CLUSTER MDUP: Received MAC FLUSH message, option: Flush
Client Rbridge, cluster_id: 1, Peer Rbridge: 1, Flush Rbridge: 400, vlan: ,
port_id: 65535
Dec  5 16:04:35 CLUSTER MDUP: Received MAC FLUSH message, option: Flush Rbridge,
cluster_id: 1, Peer Rbridge: 1, Flush Rbridge: 1, vlan: , port_id: 12
Dec  5 16:04:35 CLUSTER MDUP: Received MAC INFO message, Type: Rbridge, cluster:
1, vlan: , Rbridge id: 400
Dec  5 16:04:35 CLUSTER FSM: Received CCRR message from peer when CCP is up (value
= 0190001a)
Dec  5 16:04:35 CLUSTER FSM: cluster id 1, client id 400, old state: Up, event:
Remote Down
Dec  5 16:04:35 CLUSTER FSM: sending EVENT_ID_MCT_REMOTE_CCEP_DOWN event

SYSLOG: <46>Dec  5 16:04:35 Brocade CLUSTER FSM: Cluster MCT-VPLS (Id: 1), client
R4 (RBridge Id: 400) - Remote client CCEP down
Dec  5 16:04:35 CLUSTER FSM: Sending client fsm state to LP, state 3
Dec  5 16:04:35 CLUSTER FSM: new state: Local Up, master: FALSE
Dec  5 16:04:35 CLUSTER FSM: Received CCRR message from peer when CCP is up (value
= 0190001a)
Dec  5 16:04:35 CLUSTER MDUP: Convert CCR to CCL for cluster_id: 1, client rbridge
id = 400

SYSLOG: <41>Dec  5 16:04:37 Brocade DOT1AG: Remote MEP 4000 in Domain VPLS_SP, MA
vpls_420 aged out

SYSLOG: <41>Dec  5 16:04:39 Brocade DOT1AG: Remote MEP 4000 in Domain VPLS_SP, MA
vpls_420 become UP state
Dec  5 16:04:39 CLUSTER FSM: Received CCRR message from peer when CCP is up (value
= 0190001e)
Dec  5 16:04:39 CLUSTER FSM: cluster id 1, client id 400, old state: Local Up,
event: Remote Up
Dec  5 16:04:39 CLUSTER FSM: sending EVENT_ID_MCT_REMOTE_CCEP_UP event

SYSLOG: <46>Dec  5 16:04:39 Brocade CLUSTER FSM: Cluster MCT-VPLS (Id: 1), client
R4 (RBridge Id: 400) - Remote client CCEP up
Dec  5 16:04:39 CLUSTER FSM: Sending client fsm state to LP, state 2
Dec  5 16:04:39 CLUSTER FSM: new state: Up, master: FALSE
Dec  5 16:04:39 CLUSTER FSM: Received CCRR message f
```

Configuration notes

- ICL ports must not be untagged members of any VLAN. An ICL is preferably a LAG that provides port level redundancy and higher bandwidth for cluster communication.
- ICL ports can be part of MCT VLANs as well as regular VLANs.
- The ICL VLAN mask must be a superset of the client VLAN mask.
- MAC learning is disabled on ICL ports for the MCT VLANs.
- MAC Database Update Protocol (MDUP) will synchronize all MAC entries for VLANs served by an ICL link.
- On CCEP ports, MCT does not support MPLS, VLL, VPLS, 802.1ah, 802.1ad, and routing protocols.

VPLS unicast forwarding

VPLS enhances the point-to-point connectivity defined in the Draft-Martini IETF documents by specifying a method for Virtual Circuits (VCs) to provide point-to-multipoint connectivity across the MPLS domain, allowing traffic to flow between remotely connected sites as if the sites were connected by a Layer 2 switch.

VPLS can be used to transport Ethernet frames to and from multiple, geographically dispersed sites belonging to a customer VPN. The provider edge (PE) devices connecting the customer sites provide functions similar to a Layer 2 switch. The PE devices learn the MAC addresses of locally connected customer devices, flood broadcast and unknown-unicast frames to other PE devices in the VPN, and create associations between remote MAC addresses and the VC LSPs used to reach them.

For more information about MPLS VPLS diagnostics, refer to Chapter 5, “MPLS Diagnostics”.

VPLS unicast forwarding show commands

This section describes the show commands that display VPLS unicast forwarding information.

show mpls debug vpls

Syntax: **show mpls debug vpls** *vpls id*

This command displays generic VPLS debug information, as shown in the following example.

```
Brocade# show mpls debug vpls 1
ID:        1      Name:     test 1
CPU-Prot: OFF    MVID:     INVD     FID:    0x00002002
MAC Info:
     Total MACs: 2  Local: 2  Remote: 0
     Max Exceed: 0 Table Full: 0
```

show mpls debug vpls local

Syntax: **show mpls debug vpls local** *num*

This command displays the state of a VPLS. Specify the VPLS ID to see the local entries for a specific VPLS. Command output resembles the following example (specified for VPLS 2).

```
Brocade# show mpls debug vpls local 2
VPLS 2:
    VLAN  Port   Valid  Pending
    ====  ====   =====  =======
    4     2/19   1      0
    Local Broadcast Fids:
    =====================
    VLAN 4        --  Fid: 00008fa6, Ports: 1
      Port 2/19   --  Fid: 0000003a
```

show mpls debug vpls remote

Syntax: **show mpls debug vpls remote** *num*

This command displays the state of all VPLS peers configured in the system. To display information for a specific VPLS, enter a VPLS ID number. Command output resembles the following example (specified for remote VPLS 1).

```
Brocade# show mpls debug vpls remote 1
VPLS 1:
    Peer:  10.5.5.5             Valid:  Yes     Pending Delete: 0
    Label: 983040              Tagged: No      Load Balance:   No
                                               Num LSP Tnnls:  1
          VC        Tunnel    NHT            Use
    Port  Label     Index     COS  COS
    ====  =====     ======    =====     ===  ===
    2/9   983040    3         0         0    0
    2/9   983040    3         0         0    0
    2/9   983040    3         0         0    0
    2/9   983040    3         0         0    0
    Active Trunk Index: 0
```

Internally, a maximum of four LSP tunnels are maintained to reach the peer. If load balancing is disabled, information for only one tunnel is displayed in the output.

Common diagnostic scenarios

- VPLS does not accept a receiver report where VPLS VLAN 1002 is configured with multicast active.

  This issue is resolved by upgrading the software version to reflect the latest patches and versions.
- VPLS drops multicast traffic with multicast addresses that have not been reported to the Brocade MLX series routers (through IGMP Report).

  This issue is resolved by upgrading the software version to reflect the latest patches and versions.

GRE and IPv6 tunnels

The following commands provide information about Generic Routing Encapsulation (GRE) and IPv6 tunnels. Because both tunnel types are used by various applications (Multicast, PBR, and so on), the following commands display specific debugging messages with detailed information. These commands help to isolate events for a specific tunnel, a range of tunnels, or all tunnels.

debug iptunnel

Syntax: [**no**] **debug iptunnel** [**errors** | **events** | **ipc** | **keepalives** | **packets** | **statistics** | **tunnel-type** [**gre** | **ipv6**]] [**range** *tunnel-id low* - *tunnel-id high*]

- **errors** - Displays IP tunnel errors.
- **events** - Displays IP tunnel events.
- **ipc** - Displays IP tunnel IPC messages.
- **keepalives** - Displays keepalive information (for GRE tunnels only).
- **packets** - Displays IP tunnel packets information.
- **statistics** - Displays tunnel statistics.
- **tunnel-type** - Displays information for a specific tunnel type.
- **gre** - Displays information for a specific GRE tunnel.
- **ipv6** - Displays information for a specific IPv6 tunnel.

- **range** *tunnel-id low* - *tunnel-id high* - Displays information for a range of tunnels (specify lower and higher tunnel ID values).

debug iptunnel errors

Syntax: [**no**] **debug iptunnel errors** [**range** *tunnel-id low* - *tunnel-id high*]

This command displays any error messages for the specified range of tunnels, including unexpected events as part of the packet flow, IPC flow, or configuration of IP tunnels.

Command output resembles the following example.

```
Brocade# debug iptunnel errors
Apr 28 11:54:24 TNNL_GRE:ERRORS: tnnl 2- IP tunnel is invalid
Apr 28 11:54:24 TNNL_GRE:ERRORS: tnnl 2- L3 encapsulate and send - Dropping as
Tnnl is Down
```

debug iptunnel events

Syntax: [**no**] **debug iptunnel events** [**range** *tunnel-id low* - *tunnel-id high*]

This command displays any tunnel events for the specified range of tunnels, including.route changes, IPv4 or IPv6 port state notifications, tunnel status changes because of destination route changes or source interface changes, and any events that cause the tunnel status and operational information to be changed that can affect the routes.

Command output resembles the following example.

```
Brocade# debug iptunnel events
//Disable the tunnel on local side
TNNL_GRE:EVENTS:tnnl 1-IP notify -  Removing Tunnel IP - tnnl, state DOWN
TNNL_GRE:EVENTS:tnnl 1-IP notify-Tnnl state DOWN: Notifying all routing protocols
TNNL_GRE:EVENTS:tnnl 1-Deleting NHT for tnnl, nht 0
TNNL_GRE:EVENTS:tnnl 1-Active MP Sending NHT for tnl,nht_idx 0, action Delete
TNNL_GRE:EVENTS:tnnl 1-Update NHT Entry as Tnnl Dest route change for Tunnel

//Enable the tunnel on local side
TNNL_GRE:EVENTS:tnnl 1-IP notify -  Adding Tunnel IP - tnnl, state UP
TNNL_GRE:EVENTS:tnnl 1-IP notify-Tnnl state UP: Notifying all routing protocols
TNNL_GRE:EVENTS:tnnl 1-NHT entry Created for tnnl, nht 0, IP 10.11.11.21
TNNL_GRE:EVENTS:tnnl 1-Creating NHT and sending info to LP for tnnl
TNNL_GRE:EVENTS:tnnl 1-Active MP Sending NHT for tnl,nht_idx 0, action Add
TNNL_GRE:EVENTS:tnnl 1-Adding NHT index for tnnl, nht 0, outport 2/12
```

debug iptunnel ipc

Syntax: [**no**] **debug iptunnel ipc** [**range** *tunnel-id low* - *tunnel-id high*]

This command displays IPC messages related to debugging information for the specified range of tunnels. Command output resembles the following example.

```
Brocade# debug iptunnel ipc
TNNL_GRE:IPC:tnnl 1-Sending GRE Tunnel Update for tnnl
TNNL_GRE:IPC:tnnl 1-One Tunnel update - src-ip 10.11.11.1, dst-ip 10.11.11.21

With tunnel enabled manually
TNNL_GRE:IPC:tnnl 1-Sending GRE Tunnel Update for tnnl
TNNL_GRE:IPC:tnnl 1-Tnnl is UP
TNNL_GRE:IPC:tnnl 1-One Tunnel update - src-ip 10.11.11.1, dst-ip 10.11.11.21
```

debug iptunnel keepalives

Syntax: [**no**] **debug iptunnel keepalives** [**range** *tunnel-id low* - *tunnel-id high*]

This command displays keepalive events for the specified range of tunnels.The output displays messages about the receiving and transmitting of keepalive packets, keepalive timer actions, and bringing the tunnel up or down based on keepalive actions.

NOTE
This command only applies to GRE tunnels and does not work for IPv6 tunnels.

Normal keepalive messages:

```
Brocade# debug iptunnel keepalives
TNNL_GRE:KALIVE:tnnl 1-Keepalive Timer- For Tunnel, remaining time 10
TNNL_GRE:KALIVE:tnnl 1-TX Keepalive packet on tnnl, src 10.11.11.1, dst
10.11.11.21
TNNL_GRE:KALIVE:tnnl 1-Keepalive packet received at MP for tunnel
TNNL_GRE:KALIVE:tnnl 1-Rx Keepalive packet, src 10.11.11.21, dst 10.11.11.1
TNNL_GRE:KALIVE:tnnl 1-Reset the keepalives in the Keepalive List for tunnel
```

With tunnel disabled on the remote side so that keepalives bring down the tunnel:

```
TNNL_GRE:KALIVE:tnnl 1-Bring DOWN GRE Tunnel due to keepalive timeout
TNNL_GRE:EVENTS:tnnl 1-IP notify -  Removing Tunnel IP - tnnl, state DOWN
TNNL_GRE:EVENTS:tnnl 1-IP notify-Tnnl state DOWN: Notifying all routing protocols
TNNL_GRE:KALIVE:tnnl 1-TX Keepalive packet on tnnl, src 10.11.11.1, dst
10.11.11.21
```

With tunnel enabled on the remote side so that the tunnel comes back up on the local side:

```
TNNL_GRE:KALIVE:tnnl 1-TX Keepalive packet on tnnl, src 10.11.11.1, dst
10.11.11.21
TNNL_GRE:KALIVE:tnnl 1-Keepalive packet received at MP for tunnel
TNNL_GRE:KALIVE:tnnl 1-Rx Keepalive packet, src 10.11.11.21, dst 10.11.11.1
TNNL_GRE:KALIVE:tnnl 1-Reset the keepalives in the Keepalive List for tunnel
TNNL_GRE:KALIVE:tnnl 1-Bring UP GRE Tunnel as keepalive response is received
TNNL_GRE:EVENTS:tnnl 1-IP notify -  Adding Tunnel IP - tnnl, state UP
TNNL_GRE:EVENTS:tnnl 1-IP notify-Tnnl state UP: Notifying all routing protocols
```

debug iptunnel packets

Syntax: [**no**] **debug iptunnel packets** [**range** *tunnel-id low* - *tunnel-id high*]

This command displays packet processing details for the specified range of tunnels. Output shows the packets received and transmitted on any particular tunnel including control packets, keepalive packets, and so on.

Command output resembles the following example.

```
Brocade# debug iptunnel packets
TNNL_GRE:PKTS:tnnl 1-Sending packets to tunnel, dest 10.111.111.21
TNNL_GRE:PKTS:tnnl 1-Sending packets to tunnel, dest 10.111.111.21
```

debug iptunnel statistics

Syntax: [**no**] **debug iptunnel statistics** [**range** *tunnel-id low* - *tunnel-id high*]

This command displays tunnel statistics for the specified range of tunnels, including messages about statistics collection, statistics IPC actions, background polling of statistics information, and so on.

Command output resembles the following example.

```
Brocade# debug iptunnel statistics
TNNL_GRE:STATS: tnnl 1-Statistics sync processed for IP tunnel
TNNL_GRE:STATS: tnnl 1-Statistics sync processed for IP tunnel
TNNL_GRE:STATS: tnnl 1-Statistics sync processed for IP tunnel
TNNL_GRE:STATS: tnnl 1-LP to MP IPC:PPCR0: recv_ucast 2, recv_mcast 0, xmit_ucast
0, xmit_mcast 0
```

debug iptunnel tunnel-type

Syntax: [**no**] **debug iptunnel tunnel-type** [**gre** | **ipv6**] [**range** *tunnel-id low* - *tunnel-id high*]

You can specify the type of tunnel to debug. The valid options are **gre** for GRE tunnels and **ipv6** for IPv6 tunnels. Within a specified range of tunnels, information is filtered for the tunnel type if it is configured. If a tunnel type is not configured, debug messages are printed based on range only.

The **no debug iptunnel tunnel-type** command resets the previously configured tunnel type (if any), and prints messages for tunnels within the specified range.

LP CPU packet statistics

Protocol control packets and data packets are forwarded to the Line Processor Central Processing Unit (LP CPU) for packet processing. The LP CPU forwards the packets to a destination port or the Forwarding ID (FID), and the packet is either sent to the Management Processor (MP) module for processing, or the packet is dropped based on the forwarding decision. At various stages of the LP CPU packet processing, packet counters are maintained to assist with debugging.

NOTE
LP CPU packet statistics are maintained after MP switchover.

NOTE
LP CPU packet statistics are reset to zero after hitless upgrade and LP reboot.

LP CPU packet statistics show command

This section describes the show commands that display LP CPU packet information.

show lp-cpu packet statistics

Syntax: **show lp-cpu packet statistics**

This command displays the total number of packets that are received, dropped, forwarded, or flooded by the LP CPU during packet processing. For packets that are dropped, packet drop causes are indicated in the output display. This command also displays the number of packets processed by the LP CPU received from Layer 2, Layer 3, MPLS uplinks, VPLS, and other data packets. The LP counters are text-wrapped to zero on the CLI. The LP CPU counters are not cleared after you display packet statistics using the **show lp-cpu packet statistics** command. The LP CPU counters are displayed on both the MP and the LP.

The following counters are not displayed on the CLI using the **show lp-cpu packet statistics** command:

- PBIF RX or TX packet counters

- XPP and TM packet counters
- MP or LP IPC packet counters
- Packets originated by the MP or LP.

NOTE
The **show lp-cpu packet statistics** command displays statistics for packets that are received by the LP CPU from hardware only.

This command can be used on both the MP and LP modules. On the LP module, the **show lp-cpu packet statistics** command displays LP CPU counters only for a specific LP module. The following example output displays the total number of packets received by the LP CPU from MPLS, VPLS, VLL, ARP, and other data packets on module 4.

```
Brocade# show lp-cpu packet statistics
MODULE 4
----------
Total Packets Received: 1000
MPLS uplink packets received: 0
    VPLS packets received:0
    VLL packets received: 0
    L3 VPN packets received:0
    Other MPLS packets received:0
ARP packets received:0
    ARP response packets received:0
    ARP request packets received:0
IPV4 packets received:0
    IPv4 unicast packets routed:0
    IPv4 protocol packets received:0
    GRE tunnel packets received:0
    6to4 tunnel packets received:0
IPV6 packets received:0
    IPv6 unicast packets routed:0
    Ipv6 protocol packets received:0
IPv4 multicast packets routed:0
IPv6 multicast packets routed:0
L2VPN endpoint packets received:0
    VPLS endpoint packets received:0
    VLL endpoint packets received:0
    Local-VLL endpoint packets received:0
L2 packets received:1000
    L2 known unicast packets forwarded:600
    L2 unknown unicast packets flooded:100
    L2 broadcast Packets flooded:100
    L2 multicast Packets flooded:0
    Packets received for SA learning:0
Other packets received:0
Total Packets dropped:200
Packet drop causes:
[PP – Post Processing Drops; CV – Configuration Violation Drops;
 PE - Packet Error Drops; PFE - Potential Forwarding Error Drops;
 OD - Other Drops]

3-L2 packet forwarded in hardware(PP):100
20-L2 port state is blocking(CV):100

ARP packets captured for DAI:0
ARP packets failed DAI:0
```

Table 8 displays the descriptions of the fields from the **show lp-cpu packet statistics** command.

TABLE 8 Output from the **show lp-cpu packet statistics** command

Field	Description
Module	The interface module number that is currently connected to the chassis.
Total packets Received	The total number of packets received by the LP CPU. For example, when a packet is received by the LP CPU from a local port, the Total packets Received counter is incremented. The LP CPU performs various checks on the received packet. For example, the Total packet Received counter is incremented when a packet is received on a trunk that has no active local ports, or when a packet is received with an invalid VLAN ID.
MPLS uplink packets received	The number of MPLS labeled packets (VPLS, VLL, L3 VPN, and other MPLS packets) received on MPLS uplinks by the LP CPU.
VPLS packets received	
VLL packets received	
L3 VPN packets received	
Other MPLS packets received	
ARP packets received	The total number of ARP packets received by the LP CPU.
ARP response packets received	The number of ARP response packets received by the LP CPU.
ARP request packets received	The number of ARP request packets received by the LP CPU.
IPv4 packets received	The total number of IPv4 routed packets received by the LP CPU.
IPv4 unicast packets routed	The number of IPv4 unicast packets routed by the LP CPU.
IPv4 protocol packets received	The number of IPv4 protocol packets received by the router or the switch.
GRE tunnel packets received	The number of IPv4 or GRE packets received by the LP CPU.
6to4 tunnel packets received	The number of IPv4 6to4 tunnel packets received by the LP CPU.
IPv6 packets received	The total number of IPv6 packets received by the LP CPU.
IPv6 unicast packets routed	The number of IPv6 unicast packets routes by the LP CPU.
IPv6 protocol packets received	The number of IPv6 protocol packets destined to the router or the switch.
IPv4 multicast packets routed	The number of IPv4 multicast packets (including multicast snooping packets), received by the LP CPU.
IPv6 multicast packets routed	The number of IPv6 multicast packets routed by the LP CPU.
L2VPN endpoint packets received	The total number of L2VPN endpoint packets received by the LP CPU.
VPLS endpoint packets received	The number of VPLS packets received on the VPLS endpoints.
VLL endpoint packets received	The number of VLL packets received on the VLL endpoints.
Local-VLL Endpoint packets received	The number of local-VLL endpoint packets received by the LP CPU.
L2 packets received	The number of packets received for Layer 2 processing by the LP CPU. The Layer 2 packets received counter is incremented when Layer 2 forwarding receives a packet.
L2 known unicast packets forwarded	The number of known Layer 2 unicast packets forwarded by the LP CPU. The L2 known unicast packets forwarded counter is incremented when a packet received with a known unicast MAC address is forwarded to the destination port.
L2 unknown unicast packets flooded	The number of Layer 2 unknown-unicast packets flooded by the LP CPU.

TABLE 8 Output from the **show lp-cpu packet statistics** command (Continued)

Field	Description
L2 broadcast packets flooded	The number of Layer 2 broadcast packets flooded by the LP CPU.
L2 multicast packets flooded	The number of Layer 2 multicast packet (including packets forwarded by IGMP snooping and multi-port MAC packets) received by the LP CPU.
Packets received for SA learning	The number of packets received for source MAC address learning.
Other packets received	The number of miscellaneous packets received by the LP CPU. The Other packets received counter is incremented when a packet is not processed by any of the forwarding counters displayed previously (for example, MPLS uplinks packets received counter, ARP packets received counter, and so on).
Total packets dropped	The total number of packets dropped by the LP CPU.
Packet drop causes	The reasons for dropping a packet, and the number of packets that are dropped for a given reason. The LP CPU forwards the packet to a destination port or destination FID, drops the packet based on the forwarding decision, or drops the packet due to errors in packet processing by the LP CPU. Packets can be dropped based on following reasons: • Post Processing (PP) Drops - The packets are dropped because the packets were forwarded in hardware by the LP CPU and copied to the CPU to learn SA or DA MAC addresses, or for sFlow logging. The PP drops are to be expected and are not harmful to the LP CPU packet processing. For a complete list of all PP drop causes, refer to Table 9 on page 431. • Configuration Violation (CV) Drops- The packets are dropped because of configuration violations, such as Layer 2 packets received on a route-only switch. A route-only switch implies that Layer 2 packets are only routed, not switched. Layer 2 packets are usually forwarded within the same VLAN and are not routed to a different VLAN. If packets are configured to route only, then Layer 2 packets are dropped. A CV drop can also occur when the number of broadcast packets has exceeded the configured limit set by the user. A port in a blocking state constitutes a CV drop. For a complete list of all CV drop causes, refer to Table 9 on page 431.
Packet drop causes (continued)	Packets can be dropped based on following reasons (continued): • Packet Error (PE) Drop - A packet is dropped because it is received with errors because the packet is corrupted, or the packet contains incorrect information such as an invalid SA or DA MAC address. For a complete list of all PE drop causes, refer to Table 9 on page 431. • Potential Forwarding Error (PFE) Drops - A PFE drop occurs when the information required to forward a packet to its destination is not available in the LP software. For example, the destination FID is invalid, or ARP is not resolved. For a complete list of all PFE drop causes, refer to Table 9 on page 431. • Other Drops (OD) - OD drops are drops that cannot be classified into any of the preceding drop causes. For a complete list of all OD drop causes, refer to Table 9 on page 431. For example, the following is displayed in the output: **3-L2 packet forwarded in hardware(PP):9619** • 3 refers to the drop code. • L2 packet forwarded in hardware refers to the type of packet drop cause. For a list of all packet drop causes, refer to Table 2. • 9619 refers to the number of packets that are dropped.
ARP packets captured for DAI	The number of ARP packets captured for Dynamic ARP Inspection (DAI).
ARP packets failed DAI	The number of DAI-failed ARP Packets.

TABLE 9 Packet drop causes for LP CPU counters

Drop Category	Drop code	Drop message	Drop description
Post Processing (PP) Drops	1	Layer 2 packet forwarded in hardware (PP)	A Layer 2 packet is forwarded in hardware, and a copy of the packet is sent to the CPU for source MAC learning.
	2	Layer 3 packet forwarded in hardware (PP)	A Layer 3 packet is forwarded in hardware, and a copy of the packet is sent to the CPU for source MAC learning.
	3	IPv4 RPF logging packet (PP)	An IPv4 packet is dropped because of the RPF check failure, and the packet is sent to the CPU for logging.
	4	IPv6 RPF logging packet (PP)	A n IPv6 packet is dropped because of the RPF check failure, and the packet is sent to the CPU for logging.
	5	IPv4 multicast filtering drop (PP)	An IPv4 multicast packet is received for a group with no receivers. IPv4 multicast filtering is enabled.
	6	IPv6 multicast filtering drop (PP)	An IPv6 multicast packet is received for a group with no receivers. IPv6 multicast filtering is enabled.
	7	ACL logging packet (PP)	A packet is dropped because of ACL filtering, and the packet is sent to the CPU for logging.
	8	Packet received for sFlow sampling (PP)	A packet is forwarded in hardware, and a copy of the packet is sent to the CPU for sFlow sampling.
	9	IP packet forwarded as fragments and dropped the original (PP)	The IP packet is sent as fragments, and the original packet is dropped.
	10	L3 multicast drop	A packet is dropped in multicast processing.
	11	Reserved UDLD packet (PP)	A Brocade UDLD packet is processed and dropped.
	12	PBB Packet for source MAC learning (PP)	A PBB packet is sent to software for software MAC learning.
	13	PBB Packet for software s-flooding (PP)	The original packet is dropped in software, and copies of the packet are sent to the provider edge ports.
Configuration Violation (CV) Drops	14	Route-only enabled (CV)	A switch is configured in route-only mode, and the Layer 2 packet received is dropped.
	15	Invalid VLAN ID (CV)	A packet is received with a VLAN ID that is not configured in the switch.
	16	Layer 2 Trace - L2 Switching not enabled (CV)	Layer 2 switching is not enabled on the switch, and the Layer 2 trace route packet received is dropped.

TABLE 9 Packet drop causes for LP CPU counters (Continued)

Drop Category	Drop code	Drop message	Drop description
	17	Layer 2 Trace - unknown VLAN (CV)	The Layer 2 trace route packet is received with a VLAN ID that is not configured in the switch.
	18	Port security violation (CV)	A packet with an unauthorized MAC address is received on a port.
	19	Logical port state is down (CV)	The link level protocol state is not enabled.
	20	Layer 2 port state is blocking (CV)	The port on which the packet is received is not in the forwarding state.
	21	Layer 2 Trace - STP port state is blocking (CV)	The port on which the Layer 2 trace packet is received is not in the forwarding state.
	22	STP/PVST protected port drop (CV)	The port is configured to drop BPDUs.
	23	STP packet dropped on STP boundary port (CV)	A BPDU is received on a configured SuperSpan boundary port.
	24	STP packet dropped on layer 2 free port (CV)	A Layer 2 protocol packet is received on a port with all Layer 2 protocols disabled.
	25	STP packet dropped while cluster VLAN processing (CV)	An STP BPDU is received on a port when the **cluster-l2protocol-forward** command is not enabled.
	26	Broadcast packet limit reached (CV)	The number of received Layer 2 broadcast packets that have exceeded the configured limit in a given time interval.
	27	Multicast packet limit reached (CV)	The number of received Layer 2 multicast packets that have exceeded the configured limit in a given time interval.
	28	Unknown unicast packet limit reached (CV)	The number of received Layer 2 unknown-unicast packets that have exceeded the configured limit in a given time interval.
	29	IP port is down (CV)	The port on which the packet is received is down.
	30	ARP inspection failed	A packet is dropped because of an ARP inspection failure.
	31	Port disabled for IPv4 multicast (CV)	An IPv4 multicast packet is received on a disabled multicast port.
	32	Port disabled for IPv6 multicast (CV)	An IPv6 multicast packet is received on a disabled multicast port.
	33	IP receive ACL deny (CV)	Packets are dropped in software by the IP receive ACL configuration.

TABLE 9 Packet drop causes for LP CPU counters (Continued)

Drop Category	Drop code	Drop message	Drop description
	34	DOS attack drop exceeded configuration limit (CV)	Packets are dropped in software by the DOS attack configuration.
	35	MPLS unknown VC label (CV)	An MPLS packet is received with an unknown VC label.
	36	VPLS ISID Configuration Mismatch (CV)	The ISID value received in a packet is not the same as the configured ISID value in the VPLS instance.
	37	VPLS Local Switching disabled drop (CV)	VPLS local switching is disabled in the VPLS instance.
	38	VPLS CPU Broadcast packet limit reached (CV)	The number of received VPLS broadcast packets that have exceeded the configured limit in a given time interval.
	39	VPLS CPU Multicast packet limit reached (CV)	The number of received VPLS multicast packets that have exceeded the configured limit in a given time interval.
	40	VPLS CPU Unknown unicast packet limit reached (CV)	The number of received VPLS unknown-unicast packets that have exceeded the configured limit in a given time interval.
	41	ARP not resolved (CV)	Packets are dropped because ARP is not resolved for a destination IP address. Packets that are dropped when ARP is not resolved are known as IP drop pending packets. IP drop pending packets are configured.
Packet Error (PE) Drops	42	Invalid destination MAC (PE)	Packets received with flow control destination MAC addresses, and other invalid destination MAC addresses.
	43	Layer 2 Trace - BAD packet (PE)	A Layer 2 trace route packet is received with an incorrect packet length or type information.
	44	IP TTL too small (PE)	An IP packet is received with an expired TTL.
	45	IP header length error (PE)	A malformed IPv4 packet with an incorrect header length.
	46	IP Header error (PE)	An IPv4 packet is dropped because of an invalid IPv4 header.
	47	Multicast packet received with unicast address (PE)	A Layer 3 IP multicast packet is received with a unicast destination MAC address.
	48	Packet received with error bit set (PE)	A packet is sent to the CPU as an error packet.

TABLE 9 Packet drop causes for LP CPU counters (Continued)

Drop Category	Drop code	Drop message	Drop description
	49	Invalid source MAC (PE)	Packets received with all zero source MAC addresses.
Potential Forwarding Error (PFE) drops	50	Layer 2 Trace - next Hop information failed (PFE)	A packet is dropped because it was unable to send a Layer 2 trace packet to the next hop.
	51	MAC table uninitialized (PFE)	The MAC address database is not initialized in the software.
	52	IPv4 protocol drop (PFE)	A packet is dropped during Layer 3 processing.
	53	IP no route (PFE)	A packet is dropped because there is no route to its destination.
	54	Layer 3 invalid CAM entry type (PFE)	A Layer 3 packet is dropped because of an invalid CAM entry type.
	55	Layer 3 invalid FID (PFE)	The forwarding information for a Layer 3 packet is invalid in software.
	56	IPv6 protocol drop (PFE)	A packet is dropped during IPv6 processing.
	57	MPLS LSP XC drop (PFE)	A packet is dropped because of MPLS transit processing.
	58	VLL Local processing drop (PFE)	A packet is dropped by local VLL processing.
	59	VLL processing drop (PFE)	A packet is dropped by VLL processing.
	60	VPLS processing drop (PFE)	A packet is dropped by VPLS processing.
	61	Layer 2 invalid FID (PFE)	The forwarding information for a Layer 2 packet is invalid in software.
	62	Invalid Flooding domain (PFE)	The flooding information required to flood the packet is invalid.
	63	Invalid trunk drops (PFE)	Trunk information is incorrect in software.
Other Drop (OD) Drops	64	No Active Ports in Trunk (OD)	A packet is received on a trunk port with no active trunk ports in the local line card.
	65	Invalid source port (OD)	A packet is received with an invalid port ID, or the port information is missing in software.
	66	DA learned on source port (OD)	The destination MAC address is learned on the port on which the packet is received.
	67	PPP control packet drop (OD)	A packet is dropped because of an unsupported Point-to-Point Protocol (PPP) control packet.

TABLE 9 Packet drop causes for LP CPU counters (Continued)

Drop Category	Drop code	Drop message	Drop description
	68	NON ISIS OSI packet (OD)	A packet is dropped because a non-IS-IS OSI packet is received.
	69	Loop detection BPDU drop (OD)	A packet is dropped because of an invalid loop detect BPDU.

show lp-cpu packet statistics brief

Syntax: **show lp-cpu packet statistics brief**

This command displays the total number of packets processed and dropped on all LPs by the LP CPU. This command can be used on both the MP and the LP modules. On the LP module, this command displays brief counters only for a specific LP module. The following example output displays the total number of packet statistics that are processed and dropped on module 4 and module 7 on the MP.

```
Brocade# show lp-cpu packet statistics brief

MODULE 4
--------
Packets processed: 1000
Packets dropped: 200

MODULE 7
--------
Packets processed: 236076
Packets dropped: 194163
```

show lp-cpu packet statistics protocol

Syntax: **show lp-cpu packet statistics protocol** [**l2** | **l3**] [*slotnum/portnum* | *slotnum*]

- **l2** - Displays Layer 2 protocol packet counters received by the LP CPU.
- **l3** - Displays Layer 3 protocol packet counters received by the LP CPU.
- *slotnum/portnum* - Displays protocol packet statistics for an individual port.
- *slotnum* - Displays protocol packet statistics for all ports on the LP.

This command displays Layer 2 or Layer 3 protocol packets received on a specific slot port, or for all ports on the LP. This command can be used on both the MP and the LP, and the output display is the same for both types of modules when Layer 2 or Layer 3 protocol counters are displayed.

The following example output displays Layer 2 protocol packet counters received by the LP CPU on port 4/1 on the MP.

```
Brocade# show lp-cpu packet statistics protocol l2 4/1
LACP packets received:421
STP packets received:200
PVST packets received:0
VSRP packets received:0
ERP packets received:0
MRP packets received:0
L2 Trace Packets received:0
Loop detect packets received:0
FDP packets received:0
CDP packets received:0
Dot1x packet received:0
UDLD packets received:0
```

Table 10 displays the output from the **show lp-cpu packet statistics protocol l2** *slotnum/portnum* command.

TABLE 10 Output from the **show lp-cpu packet statistics protocol l2** command

Field	Description
LACP packets received	The number of LACP packets received by the LP CPU.
STP packets received	The number of STP packets received by the LP CPU.
PVST packets received	The number of PVST packets received by the LP CPU.
VSRP packets received	The number of VSRP packets received by the LP CPU.
ERP packets received	The number of ERP packets received by the LP CPU.
MRP packets received	The number of MRP packets received by the LP CPU.
L2 Trace Packets received	The number of Layer 2 trace packets received by the LP CPU.
Loop detect packets received	The number of loop detect packets received by the LP CPU. Loop detect packets refers to a protocol used to detect loops in the network.
FDP packets received	The number of FDP packets received by the LP CPU.
CDP packets received	The number of CDP packets received by the LP CPU.
Dot1x packet received	The number of Dot1x packets received by the LP CPU. Dot1x packets refer to the 802.1 authentication protocol.
UDLD packets received	The number of UDLD packets received by the LP CPU.

The following example output displays Layer 3 protocol packet counters received by the LP CPU on port 4/1 on the MP.

```
Brocade# show lp-cpu packet statistics protocol l3 4/1
IPv4 Protocol packets:
IPv4 IGMP packets received:0
IPv4 ICMP packets received:0
IPv4 UDP packets received:0
IPv4 TCP packets received:0
IPv4 RSVP packets received:0
IPv4 IGP packets received:0
IPv4 OSPF packets received:0
IPv4 VRRP packets received:0
IPv4 PIM packets received:0
IPv4 GRE packets received:0
IPv4 BFD packets received:0
IPv6 over IPv4 packets received:0

IPv6 Protocol packets:
IPv6 BFD packets received:0
IPv6 PIM packets received:0
IPv6 ICMP packets received:0
IPv6 UDP packets received:0
IPv6 TCP packets received:0
IPv6 OSPF packets received:0
IPv6 VRRP packets received:0
```

Table 11 displays the output from the **show lp-cpu packet statistics protocol l3** *slotnum/portnum* command.

TABLE 11 Output from the **show lp-cpu packet statistics protocol l3** command

Field	Description
IPv4 Protocol packets received	The number of IPv4 protocol packets received by the LP CPU.
IPv4 IGMP packets received	The number of IPv4 IGMP packets received by the LP CPU.
IPv4 ICMP packets received	The number of IPv4 ICMP packets received by the LP CPU.
IPv4 UDP packets received	The number of IPv4 UDP packets received by the LP CPU.
IPv4 TCP packets received	The number of IPv4 TCP packets received by the LP CPU.
IPv4 RSVP packets received	The number of IPv4 RSVP packets received by the LP CPU.
IPv4 IGP packets received	The number of IPv4 IGP packets received by the LP CPU.
IPv4 OSPF packets received	The number of IPv4 OSPF packets received by the LP CPU.
IPv4 VRRP packets received	The number of IPv4 VRRP packets received by the LP CPU.
IPv4 PIM packets received	The number of IPv4 PIM packets received by the LP CPU.
IPv4 GRE packets received	The number of IPv4 GRE packets received by the LP CPU.
IPv4 BFD packets received	The number of IPv4 BFD packets received by the LP CPU.
IPv6 over IPv4 packets received	The number of IPv6 over IPv4 packets received by the LP CPU. IPv6 over IPv4 packets refers to IPv6 packets encapsulated in the IPv4 header.
IPv6 Protocol packets	The number of IPv6 protocol packets received by the LP CPU.
IPv6 BFD packets received	The number of IPv6 BFD packets received by the LP CPU.
IPv6 PIM packets received	The number of IPv6 PIM packets received by the LP CPU.

TABLE 11 Output from the **show lp-cpu packet statistics protocol l3** command (Continued)

Field	Description
IPv6 ICMP packets received	The number of IPv6 ICMP packets received by the LP CPU.
IPv6 UDP packets received	The number of IPv6 UDP packets received by the LP CPU.
IPv6 TCP packets received	The number of IPv6 TCP packets received by the LP CPU.
IPv6 OSPF packets received	The number of IPv6 OSPF packets received by the LP CPU.
IPv6 VRRP packets received	The number of IPv6 VRRP packets received by the LP CPU.

show lp-cpu packet statistics slot

Syntax: **show lp-cpu packet statistics slot** *slot-number*

The *slot-number* variable specifies the slot number for which you want to display packet statistics.

This command displays a detailed view of packet statistics processed on a specified LP module, and the total number of packets received on each slot number. Since the packet statistics for each of the LP modules are polled from the MP module, the **show lp-cpu packet statistics slot** command is supported only on the MP module. The following example output displays the total number of packets received by the LP CPU from MPLS, ARP, IPv4, IPv6, and other data packets on module 4.

```
Brocade# show lp-cpu packet statistics slot 4
MODULE 4
----------
Total Packets Received: 1000
MPLS uplink packets received: 0
    VPLS packets received:0
    VLL packets received: 0
    L3 VPN packets received:0
    Other MPLS packets received:0
ARP packets received:0
    ARP response packets received:0
    ARP request packets received:0
IPV4 packets received:0
    IPv4 unicats packets routed:0
    IPv4 protocol packets received:0
    GRE tunnel packets received:0
    6to4 tunnel packets received:0
IPV6 packets received:0
    IPv6 unicast packets routed:0
    Ipv6 protocol packets received:0
IPv4 multicast packets routed:0
IPv6 multicast packets routed:0
L2VPN endpoint packets received:0
    VPLS endpoint packets received:0
    VLL endpoint packets received:0
    Local-VLL endpoint packets received:0

L2 packets received:1000
    L2 known unicast packets forwarded:600
    L2 unknown unicast packets flooded:100
    L2 broadcast Packets flooded:100
    L2 multicast Packets flooded:0
    Packets received for SA learning:0
Other packets received:0
Total Packets dropped:200
```

```
Packet drop causes:
[PP – Post Processing Drops; CV – Configuration Violation Drops;
 PE - Packet Error Drops; PFE - Potential Forwarding Error Drops;
 OD - Other Drops]

3-L2 packet forwarded in hardware(PP):100
20-L2 port state is blocking(CV):100

ARP packets captured for DAI:0
ARP packets failed DAI:0

packets processed by port:
        port 4/1:1000
        port 4/2:0
        port 4/3:0
        port 4/4:0
```

LP CPU packet statistics clear command

NOTE
LP CPU packet statistics can only be cleared using the **clear lp-cpu packet statistics** command.

clear lp-cpu packet statistics

Syntax: clear lp-cpu packet statistics *slot-number*

This command clears all packet statistics processed on a specific LP, or for all LPs. This command is used on both the MP and the LP. On the MP module, the **clear lp-cpu packet statistics** command clears all packet statistics for a specific slot, or clears packet statistics on all LPs. On the LP module, the **clear lp-cpu packet statistics** command clears packet statistics on all ports of the LP.

The *slot-number* variable specifies the slot number for which you want to clear packet statistics.

To clear all LP CPU packet statistics processed on all LPs, enter the following command.

```
Brocade# clear lp-cpu packet statistics
```

To clear all LP CPU packet statistics processed on slot 1 on the MP, enter the following command.

```
Brocade# clear lp-cpu packet statistics 1
```

CPU aggregate counter statistics

This section describes the commands that display the aggregate statistics for all the CPU queues. To display CPU aggregate statistics per priority from the Multicast queue on a module, enter a command such as the following.

```
Brocade(config)# show tm-voq-stat src_port eth 2/1 cpu-queue-all 2
EnQue Pkt Count 100
EnQue Bytes Count 22400
DeQue Pkt Count 100
DeQue Bytes Count 22400
Total Discard Pkt Count 0
Total Discard Bytes Count 0
Oldest Discard Pkt Count 0
Oldest Discard Bytes Count 0
WRED Dropped Pkt Count 0
```

```
WRED Dropped Bytes Count 0
```

Syntax: show tm-voq-stat src_port *source-port* **cpu-queue-all** [*priority* | **all**]

You must specify a source-port.

You can optionally specify a priority to limit the display to a single priority or use the **all** parameter to display all priorities.

You can still configure statistics for each of the CPU queue like **cpu-copy-queue**, **cpu-mgmt-queue**, **cpu-proto-queue** or **cpu-queue**.

```
Brocade(config)# show tm-voq-stat src_port eth x/y cpu-copy-queue 1
Brocade(config)# show tm-voq-stat src_port eth x/y cpu-mgmt-queue 1
Brocade(config)# show tm-voq-stat src_port eth x/y cpu-proto-queue 1
Brocade(config)# show tm-voq-stat src_port eth x/y cpu-queue 1
```

Syntax: show tm-voq-stat src_port *source-port* **cpu-queue** | **cpu-copy-queue** | **cpu-mgmt-queue** | **cpu-proto-queue** [*priority* | **all**]

The range for the priority is from 0 through 7 and **all** is for all priority.

The following example is the sample output from the **cpu-copy-queue** command.

```
Brocade(config)# show tm-voq-stat src_port eth 2/2 cpu-copy-queue 1
   EnQue Pkt Count 0
   EnQue Bytes Count 0
   DeQue Pkt Count 0
   DeQue Bytes Count 0
   Total Discard Pkt Count 0
   Total Discard Bytes Count 0
   Oldest Discard Pkt Count 0
   Oldest Discard Bytes Count 0
   WRED Dropped Pkt Count 0
   WRED Dropped Bytes Count 0
   Current Queue Depth 0
   Maximum Queue Depth since Last read 0
```

The following example is the sample output from the **cpu-proto-queue** command.

```
Brocade(config)# show tm-voq-stat src_port eth 2/2 cpu-proto-queue 1
   EnQue Pkt Count 0
   EnQue Bytes Count 0
   DeQue Pkt Count 0
   DeQue Bytes Count 0
   Total Discard Pkt Count 0
   Total Discard Bytes Count 0
   Oldest Discard Pkt Count 0
   Oldest Discard Bytes Count 0
   WRED Dropped Pkt Count 0
   WRED Dropped Bytes Count 0
   Current Queue Depth 0
   Maximum Queue Depth since Last read 0
```

The **clear tm-voq-stat src_port** *source-port* **cpu-queue-all** command clears all the statistics collected per priority.

```
Brocade(config)# clear tm-voq-stat src_port eth 2/1 cpu-queue-all 2
```

Chapter

Software Licensing Diagnostics 11

In this chapter

This chapter contains diagnostic information for software licensing.

Software licensing

Software licensing enables premium features in the software. Software licensing uses the same command parsing control that is used in the EEPROM version of packaging, so it is built on an already proven infrastructure. Software licenses can be pre-installed in the factory or ordered and installed on demand by the customer.

Software licensing show command

This section describes the show command that displays software licensing information.

show license

Syntax: show license

This command displays the licenses in the system, as shown in the following example.

```
Brocade# show license
Total no. of entries: 6
Index      Package Name      Lid         Valid  Type   Period
1       NI-CES-2024-L3U   egut-cdOJ     yes    trial  2 hours
2       NI-CES-2024-L3U   egut-cdOJ     yes    trial  48 hours
3       NI-CER-2048-ADV   ucHJFOFGOH    no     trial  1 days
4       NI-CER-2048-ADV   ucHJFOFGOH    no     normal unlimited
5       NI-CES-2024-L3U   egut-cdOJ     yes    normal unlimited
```

For Brocade MLX series devices, the command output resembles the following example.

```
Brocade# show license
Index  Package Name         Lid           Slot   License    Status     License
                                                 Type                  Period
1      BR-MLX-10Gx4-MLUpg   doaFIGFhFGG   S4     normal     active     unlimited
2      BR-MLX-10Gx4-MLUpg   doaFIGFhFGG   S4     trial      not used   3 hours
3      BR-MLX-1Gx24-MLUpg   ONMLKJIHG     S3     normal     active     unlimited
4      BR-MLX-1Gx24-MLUpg   ONMLKJIHG     S3     trial      not used   3 hours
```

Software licensing debug command

This section describes the debug command that generates software licensing information.

debug license

Syntax: [no] debug license

This command is used to display the package information on which the license has been loaded. It is encoded as a Hex value. This information can be displayed only when the show command is used with the license index; for example, **show license 1**.

Before enabling debugging:

```
Brocade# show license
Index      Package Name         Lid          License Type    Status     License
Period
1        NI-CES-2048-L3U     ucGNFOGHGO   normal         active    unlimited
Brocade#show license 1
License information for license <1>:
        +package name:        NI-CES-2048-L3U
        +lid:                 ucGNFOGHGO
        +license type:        normal
        +status:              active
        +license period:      unlimited
```

After enabling debugging:

```
Brocade# debug license
License all debugging ON
Brocade# show license 1
License information for license <1>:
        +package name:        NI-CES-2048-L3U
        +lid:                 ucGNFOGHGO
        +license type:        normal
        +status:              active
        +license period:      unlimited
Brocade license information:
        +pkg info:     0X00000003
```

Chapter

NETCONF Diagnostics

12

In this chapter

This chapter contains diagnostic information for the Netconf protocol environments.

NETCONF

Network Configuration (NETCONF) is a XML-based protocol that deals with automated configuration management. The NETCONF protocol runs on top of a secure transport, such as Simple Object Access Protocol (SOAP) or Secure Shell (SSH).

NETCONF debug commands

This section describes the NETCONF-related debug commands.

debug ip netconf content

Syntax: [no] **debug ip netconf content**

This command enables NETCONF application data debugging and displays debug messages related to the configuration and state data manipulated by the NETCONF protocol, NETCONF remote procedure calls, and NETCONF notifications. Command output resembles the following example.

```
Brocade# debug ip netconf content
NETCONF:  content debugging is on
```

debug ip netconf engine

Syntax: [no] **debug ip netconf engine**

This command enables NETCONF engine debugging, as shown in the following example.

```
Brocade# debug ip netconf engine
         NETCONF:  engine debugging is on
Dec 22 19:02:45 NETCONF [0]: FSM state: 1
Dec 22 19:02:45 NETCONF [0]: FSM state: 2
Dec 22 19:03:11 NETCONF [0]: FSM state: 2
Dec 22 19:03:11 NETCONF [0]: FSM state: 2
Dec 22 19:03:11 NETCONF [0]: FSM state: 2
Dec 22 19:03:11 NETCONF [0]: FSM state: 3
Dec 22 19:03:31 NETCONF [0]: FSM state: 3
Dec 22 19:03:31 NETCONF [0]: FSM state: 3
Dec 22 19:03:31 NETCONF [0]: FSM state: 3
Dec 22 19:03:31 NETCONF [0]: FSM state: 3
Dec 22 19:03:31 NETCONF [0]: FSM state: 3
```

```
Dec 22 19:03:31 NETCONF [0]: FSM state: 3
Dec 22 19:03:31 NETCONF [0]: FSM state: 3
Dec 22 19:03:31 NETCONF [0]: FSM state: 3
Dec 22 19:03:31 NETCONF [0]: FSM state: 3
Dec 22 19:03:31 NETCONF [0]: FSM state: 3
Dec 22 19:03:31 NETCONF [0]: FSM state: 3
Dec 22 19:03:31 NETCONF [0]: FSM state: 3
Dec 22 19:03:31 NETCONF [0]: FSM state: 3
Dec 22 19:03:31 NETCONF [0]: FSM state: 3
Dec 22 19:03:31 NETCONF [0]: FSM state: 3
Dec 22 19:03:31 NETCONF [0]: FSM state: 3
Dec 22 19:03:31 NETCONF [0]: FSM state: 3
Dec 22 19:03:31 NETCONF [0]: FSM state: 3
Dec 22 19:03:31 NETCONF [0]: FSM state: 3
```

debug ip netconf framer

Syntax: [no] debug ip netconf framer

This command displays the NETCONF XML framer level debugging information, as shown in the following example.

```
Brocade# debug ip netconf framer
         NETCONF:  framer debugging is on
Dec 22 19:12:25 NETCONF[0]: Framing Rpc Reply
Dec 22 19:12:25 NETCONF[0]: RxFrame(24271e96, 238a05d4, 33536)
Dec 22 19:12:25 NETCONF[0]: Framing Netiron Config
Dec 22 19:12:25 NETCONF[0]: RxFrame(24274bd0, 238a0675, 33375)
Dec 22 19:12:25 NETCONF[0]: Framing Vlan Config
Dec 22 19:12:25 NETCONF[0]: RxFrame(2427790a, 238a068c, 33352)
Dec 22 19:12:25 NETCONF[0]: Fetching VLAN. Config. VLAN ID 0
Dec 22 19:12:25 NETCONF[0]: Fetching Tagged/Untagged/Uplink Switch. Config. PORT
ID 65535
Dec 22 19:12:25 NETCONF[0]: Fetching Tagged/Untagged/Uplink Switch. Config. PORT
ID 65535
Dec 22 19:12:25 NETCONF[0]: Fetching Tagged/Untagged/Uplink Switch. Config. PORT
ID 47
Dec 22 19:12:25 NETCONF[0]: Fetching Tagged/Untagged/Uplink Switch. Config. PORT
ID 55
Dec 22 19:12:25 NETCONF[0]: Fetching Tagged/Untagged/Uplink Switch. Config. PORT
ID 65535
Dec 22 19:12:25 NETCONF[0]: Fetching VLAN. Config. VLAN ID 1
Dec 22 19:12:25 NETCONF[0]: Fetching Tagged/Untagged/Uplink Switch. Config. PORT
ID 65535
Dec 22 19:12:25 NETCONF[0]: Fetching Tagged/Untagged/Uplink Switch. Config. PORT
ID 47
Dec 22 19:12:25 NETCONF[0]: Fetching Tagged/Untagged/Uplink Switch. Config. PORT
ID 55
Dec 22 19:12:25 NETCONF[0]: Fetching Tagged/Untagged/Uplink Switch. Config. PORT
ID 65535
Dec 22 19:12:25 NETCONF[0]: Fetching Tagged/Untagged/Uplink Switch. Config. PORT
ID 65535
Dec 22 19:12:25 NETCONF[0]: Fetching VLAN. Config. VLAN ID 4095
Dec 22 19:12:25 NETCONF[0]: RxFrame() returns 0, used 515
Dec 22 19:12:25 NETCONF[0]: RxFrame() returns 0, used 562
Dec 22 19:12:25 NETCONF[0]: RxFrame() returns 0, used 753
```

debug ip netconf operation

Syntax: [no] debug ip netconf operation

This command displays information about NETCONF operations, such as retrieve, configure, copy, delete, and so on, as shown in the following example.

```
Brocade# debug ip netconf operation
         NETCONF:  operation debugging is on
Dec 22 19:27:55 NETCONF[0]: Received NETCONF <get-config> operation
Dec 22 19:28:32 NETCONF[0]: Received NETCONF <close-session> operation
```

debug ip netconf parser

Syntax: [no] debug ip netconf parser

This command enables NETCONF XML parser level debugging, as shown in the following example.

```
Brocade# debug ip netconf parser
         NETCONF:  parser debugging is on
Dec 22 19:33:58 NETCONF: netconf_error_reset() cdb session 11
Dec 22 19:34:41 Object = rpc, Context = 0x2426f15c and Cookie = 0x24271e96
Dec 22 19:34:41 Object = get-config, Context = 0x24271e96 and Cookie = 0x24271e96
Dec 22 19:34:41 Object = rpc, Context = 0x2426f15c and Cookie = 0x24271e96
Dec 22 19:34:41 Object = get-config, Context = 0x24271e96 and Cookie = 0x24274bd0
Dec 22 19:34:41 Object = filter, Context = 0x24274bd0 and Cookie = 0x24274bd0
Dec 22 19:34:41 NETCONF[0]: FILTER(filter): 24271e96
Dec 22 19:34:41 Object = rpc, Context = 0x2426f15c and Cookie = 0x24271e96
Dec 22 19:34:41 Object = get-config, Context = 0x24271e96 and Cookie = 0x24274bd0
Dec 22 19:34:41 Object = filter, Context = 0x24274bd0 and Cookie = 0x2427790a
Dec 22 19:34:41 Object = netiron-config, Context = 0x2427790a and Cookie =
0x2427790a
Dec 22 19:34:41 Object = rpc, Context = 0x2426f15c and Cookie = 0x24271e96
Dec 22 19:34:41 Object = get-config, Context = 0x24271e96 and Cookie = 0x24274bd0
Dec 22 19:34:41 Object = filter, Context = 0x24274bd0 and Cookie = 0x2427790a
Dec 22 19:34:41 Object = netiron-config, Context = 0x2427790a and Cookie =
0x2427a644
Dec 22 19:34:41 Object = vlan-config, Context = 0x2427a644 and Cookie = 0x2427a644
Dec 22 19:34:41 14 contexts available after releasing context for vlan-config
Dec 22 19:34:41 15 contexts available after releasing context for netiron-config
Dec 22 19:34:41 (filterDec 22 19:34:41 (netiron-configDec 22 19:34:41
(vlan-configDec 22 19:34:41 )Dec 22 19:34:41 )Dec 22 19:34:41 )Dec 22 19:34:41
Dec 22 19:34:41 16 contexts available after releasing context for filter
Dec 22 19:34:41 17 contexts available after releasing context for get-config
Dec 22 19:34:41 NETCONF: netconf_error_reset() cdb session 11
```

debug ip netconf rpc

Syntax: [no] debug ip netconf rpc

This command displays debug messages related to the NETCONF Remote Procedure Call (RPC) layer. NETCONF uses RPC-based mechanisms to ease the communication between the NETCONF client and NETCONF server.

Command output resembles the following example.

```
Brocade# debug ip netconf rpc
         NETCONF:  rpc debugging is on
Dec 22 19:57:26 NETCONF [0]: Rpc request length (16845 ) is too-big : 16768
Dec 22 19:57:26 NETCONF [0]: Rpc request length (16926 ) is too-big : 16768
Dec 22 19:57:26 NETCONF [0]: Rpc request length (16944 ) is too-big : 16768
Dec 22 19:57:26 NETCONF [0]: Rpc request length (16959 ) is too-big : 16768
Dec 22 19:57:26 NETCONF [0]: Rpc request length (16979 ) is too-big : 16768
Dec 22 19:57:26 NETCONF [0]: Rpc request length (16995 ) is too-big : 16768
Dec 22 19:57:26 NETCONF [0]: Rpc request length (17012 ) is too-big : 16768
```

```
Dec 22 19:57:26 NETCONF [0]: Rpc request length (17022 ) is too-big : 16768
Dec 22 19:57:26 NETCONF [0]: Rpc request length (17029 ) is too-big : 16768
```

debug ip netconf transport

Syntax: [**no**] **debug ip netconf transport**

This command displays debug messages related to the NETCONF transport layer.

The following example output is generated when the NETCONF server is enabled or disabled, and NETCONF transport layer debugging is enabled.

```
Brocade# debug ip netconf transport
         NETCONF:  transport debugging is on
Brocade# config t
Brocade(config)# netconf server
Oct 26 13:11:16 NETCONF: Enabling the Netconf Server port
Oct 26 13:11:16 NETCONF: Received TCP listen callback
Brocade(config)# no netconf server
Oct 26 13:11:23 NETCONF: Disabling the Netconf server port
Brocade# no debug ip netconf transport
         NETCONF:  transport debugging is off
```

Chapter

OpenFlow Diagnostics

13

In this chapter

This chapter contains diagnostic information for the OpenFlow protocol environments.

OpenFlow

OpenFlow is a programmable network protocol that manages and directs traffic among Ethernet switches, routers, and wireless access points over the network in support of Software-Defined Networking (SDN) applications. OpenFlow can be used for traffic flow management in metro, WAN, and data center networks, and also security management in enterprise and campus data center applications, and other applications with the appropriate use of OpenFlow controllers.

OpenFlow debug commands

This section describes the OpenFlow-related debug commands.

debug openflow

Syntax: [**no**] **debug openflow** [**all-stack** | **config** | **error-stack** | **flow-all** | **forwarding** | **hardware** | **info-stack** | **ipc** | **show** | **socket** | **warn-stack**]

- **all-stack** - Displays all the OpenFlow stack debug messages.
- **config** - Displays OpenFlow configuration debug messages.
- **error-stack** - Displays OpenFlow stack error messages.
- **flow-all** - Displays all the OpenFlow flow debug messages.
- **forwarding** - Displays OpenFlow forwarding debug messages.
- **hardware** - Displays OpenFlow hardware debug messages.
- **info-stack** - Displays OpenFlow stack information messages.
- **ipc** - Displays OpenFlow Interprocess Communication (IPC) debug messages.
- **show** - Displays OpenFlow debug flags.
- **socket** - Displays OpenFlow socket interface messages.
- **warn-stack** - Displays OpenFlow stack warning messages.

The **debug openflow** command enables logging of OpenFlow-related messages.

debug openflow all-stack

Syntax: [no] debug openflow all-stack

This command displays all the OpenFlow stack debug messages. Command output resembles the following example.

```
Brocade# debug openflow all-stack
SO_SOCKIF: task so.36.372 - sent 8 bytes
Mar 27 00:09:52.974 SO_SOCKIF: task so.36.372, read 8 bytes on sockfd 36
Mar 27 00:09:53.224 SO_SOCKIF: task so.36.372, read 8 bytes on sockfd 36
Mar 27 00:09:53.224 SO_SOCKIF: task so.36.372, read 72 bytes on sockfd 36
Mar 27 00:09:53.224 01583|vlogshim|INFO|rule_alloc: called
Mar 27 00:09:53.224 01584|vlogshim|INFO|insert rule: rule validation succeeded
Mar 27 00:09:53.224 01585|vlogshim|INFO|add rule: rule_id=4, table_id=4,
flow_type=11
Mar 27 00:09:53.224 01586|vlogshim|INFO|add rule: returns 0
Mar 27 00:09:53.224 01587|vlogshim|INFO|rule_update_state: rule 4, success 1
Mar 27 00:09:53.224 SO_SOCKIF: task so.36.372, read 8 bytes on sockfd 36
Mar 27 00:09:53.225 01589|vlogshim|ERR|ofconn->blocked = 2e8d31c0, ofconn->retry
= 0, count = 0,remote_ip:0,remote_port:0
Mar 27 00:09:53.225 01590|vlogshim|ERR|ofconn->blocked = 2e8d31c0, ofconn->retry
= 0, count = 0,remote_ip:0,remote_port:0
Mar 27 00:09:53.308 01591|vlogshim|INFO|rule_update_state: rule 4, success 1
Mar 27 00:09:53.308 01592|vlogshim|ERR|rule state of 4 shouldn't receive an state
update
Mar 27 00:09:53.473 01593|vlogshim|INFO|reap_completed_operations: rule 4, state
4
Mar 27 00:09:53.474
SO_SOCKIF: task so.36.372 - sent 8 bytes
Mar 27 00:09:53.476 SO_SOCKIF: !!!! Connect closed by remote for task:(so.36.372)
Mar 27 00:09:53.476 SO_SOCKIF: ending task so.36.372 with sockfd:36
Mar 27 00:09:53.724 SO_SOCKIF: session closed for sockfd:36
```

debug openflow config

Syntax: [no] debug openflow config

This command displays OpenFlow configuration debug messages. Command output resembles the following example.

```
Brocade# debug openflow config
Mar 27 00:30:28.135 OPENFLOW CONFIG: Flow type for port 1/12 is set to L2
Mar 27 00:30:28.203 OPENFLOW CONFIG: Flow type for port 1/13 is set to L2
Mar 27 00:30:28.351 OPENFLOW CONFIG: Flow type for port 1/14 is set to L2
Mar 27 00:30:28.479 OPENFLOW CONFIG: Flow type for port 1/15 is set to L2
Mar 27 00:35:23.382 OPENFLOW CONFIG: All L2 rules matched, Flow Validation
Successful
Mar 27 00:35:23.472 Updating the programmed status of the flow.
Mar 27 00:35:23.472 Update the ADD status for flow id = 1
```

debug openflow error-stack

Syntax: [no] debug openflow error-stack

This command displays OpenFlow stack error messages. Command output resembles the following example.

```
Brocade# debug openflow error-stack
OpenFlow Stack Error debugging is now ON
```

```
4P-06-31-14-1#Mar 26 23:40:13.715 01076|vlogshim|ERR|ofconn->blocked = 2e8d31c0,
ofconn->retry = 0, count = 0,remote_ip:0,remote_port:0
Mar 26 23:40:13.715 01077|vlogshim|ERR|ofconn->blocked = 2e8d31c0, ofconn->retry
= 0, count = 0,remote_ip:0,remote_port:0
Mar 26 23:40:13.805 01079|vlogshim|ERR|rule state of 4 shouldn't receive an state
update
```

debug openflow flow-all

Syntax: [no] debug openflow flow-all

This command displays all the OpenFlow flow debug messages. Command output resembles the following example.

```
Brocade# debug openflow flow-all
Mar 27 00:41:54.134 Received DELETE flow message flow id = 4
Mar 27 00:41:54.134 OPENFLOW CONFIG: All L2 rules matched, Flow Validation
Successful
Mar 27 00:41:54.134 OPENFLOW FORWARD: L2 RULE: Flow Id: 4, In Port: 1/6, In Vlan:
0, Tagged 0
  Vlan Upbit: 0, Priority:32768,Ether Type: 0
  Source Mac: 0000.0000.0000, Dest Mac:  0000.0000.0000 , Src Mac Mask:
0000.00ff.ffff, Dest Mac Mask:  0000.00ff.ffff
Mar 27 00:41:54.134 OPENFLOW FORWARDING:openflow copy flow - member count 1
Mar 27 00:41:54.134 OPENFLOW FORWARDING: Actions - out port 1/8,vlan 4096,tagged
0,prio 15,tos 3
  src-mac 0000.0000.0000, dst-mac 0000.0000.0000
Mar 27 00:41:54.134 OPENFLOW - Programming flow with 1 output member only
Mar 27 00:41:54.134 SO_SOCKIF: task so.30.77, read 8 bytes on sockfd 30
Mar 27 00:41:54.168 Openflow: Received an IPC from from LP
Mar 27 00:41:54.168 Updating the programmed status of the flow.
Mar 27 00:41:54.168 Update the ADD status for flow id = 4
```

debug openflow forwarding

Syntax: [no] debug openflow forwarding

This command displays OpenFlow forwarding debug messages. Command output resembles the following example.

```
Brocade# debug openflow forwarding
OpenFlow Forwarding debugging is now ON
4P-06-31-14-1#Mar 27 00:23:56.229 OPENFLOW:FWD: Adding a flow in the Openflow
Generic Mode.
Mar 27 00:23:56.229 OPENFLOW FORWARD: L2 RULE: Flow Id: 6, In Port: 1/3, In Vlan:
0, Tagged 0
  Vlan Upbit: 0, Priority:32768,Ether Type: 0
  Source Mac: 0000.0000.0000, Dest Mac:  0000.0000.0000 , Src Mac Mask:
0000.00ff.ffff, Dest Mac Mask:  0000.00ff.ffff
Mar 27 00:23:56.229 OPENFLOW FORWARDING:openflow copy flow - member count 1
Mar 27 00:23:56.229 OPENFLOW FORWARDING: Actions - out port 1/4,vlan 4096,tagged
0,prio 15,tos 3
  src-mac 0000.0000.0000, dst-mac 0000.0000.0000
Mar 27 00:23:56.229 OPENFLOW - Programming flow with 1 output member only
```

debug openflow hardware

Syntax: [no] debug openflow hardware

This command displays OpenFlow hardware debug messages. Command output resembles the following example.

```
Brocade# debug openflow hardware
Openflow Hardware debugging is now ON
LP-1#Mar 27  00:38:34 OPENFLOW HARDWARE: generic flow add - FID 0x0000ffff, mvid
2048
Mar 27  00:38:34 OPENFLOW HARDWARE: Openflow L2 flow - member count 0
Mar 27  00:38:34 OPENFLOW HARDWARE: pram fid 0x00000005
Mar 27  00:38:34 OPENFLOW HARDWARE: Created CAM index 0x0002801b, PRAM index
0x000000b5 for flow 3 on port 1/5
```

Command output such as the following will be displayed, when protected VLANs are configured.

```
Brocade# debug openflow hardware
Jan 25 03:13:01.875 OPENFLOW HARDWARE: Adding Hybrid Vlans
Jan 25 03:13:01.875 L3 Vlan id 10
Jan 25 03:13:01.875 OPENFLOW HARDWARE: Created CAM index 0x0001473f, PRAM index
0x000000bd on port 1/1 for Hybrid Vlan 10
```

Command output such as the following will be displayed, when protected VLANs are deleted.

```
Brocade# debug openflow hardware
Jan 25 03:17:59.925 OPENFLOW HARDWARE: Deleting Hybrid Vlans
Jan 25 03:17:59.925 L3 Cam entry found for vlan 10
Jan 25 03:17:59.925 OPENFLOW HARDWARE: Deleted CAM index 0x0001474a on port 1/1
for Hybrid Vlan 10
```

debug openflow info-stack

Syntax: [no] debug openflow info-stack

This command displays OpenFlow stack information messages. Command output resembles the following example.

```
Brocade# debug openflow info-stack
OpenFlow Stack Info debugging is now ON
4P-06-31-14-1#Mar 26 23:25:11.210 00783|vlogshim|INFO|rule_alloc: called
Mar 26 23:25:11.210 00784|vlogshim|INFO|insert rule: rule validation failed
Mar 26 23:25:11.210 00785|vlogshim|INFO|rule_dealloc: rule 0
Mar 26 23:25:40.459 00802|vlogshim|INFO|rule_alloc: called
Mar 26 23:25:40.459 00803|vlogshim|INFO|insert rule: rule validation succeeded
Mar 26 23:25:40.460 00804|vlogshim|INFO|add rule: rule_id=1, table_id=4,
flow_type=11
Mar 26 23:25:40.460 00805|vlogshim|INFO|add rule: returns 0
Mar 26 23:25:40.460 00806|vlogshim|INFO|rule_update_state: rule 1, success 1
Mar 26 23:25:40.550 00810|vlogshim|INFO|rule_update_state: rule 1, success 1
Mar 26 23:25:40.710 00812|vlogshim|INFO|reap_completed_operations: rule 1, state
4
```

debug openflow ipc

Syntax: [no] debug openflow ipc

This command displays OpenFlow IPC debug messages. Command output resembles the following example.

```
Brocade# debug openflow ipc
```

```
Mar 27 00:39:48.460 Openflow: Received an IPC from from LP
Mar 27 00:39:48.460 Updating the programmed status of the flow.
Mar 27 00:39:48.460 Update the ADD status for flow id = 4
Mar 27 00:39:50.085 Received flow stats sync ipc form lp
Mar 27 00:39:50.085 Number flows in the ipc are = 4
Mar 27 00:39:50.085 Synced flow stats with flow_id= 1 and flow_stats = 0
Mar 27 00:39:50.085 Synced flow stats with flow_id= 2 and flow_stats = 0
Mar 27 00:39:50.085 Synced flow stats with flow_id= 3 and flow_stats = 0
Mar 27 00:39:50.085 Synced flow stats with flow_id= 4 and flow_stats = 0
Mar 27 00:39:52.085 Received flow stats sync ipc form lp
```

debug openflow show

Syntax: [no] debug openflow show

This command displays OpenFlow debug flags. Command output resembles the following example.

```
Brocade# debug openflow show
Openflow debugging is                 :ENABLED
Openflow ALL debugging is             :OFF
Openflow Stack ALL debugging is         :OFF
Openflow Stack ERR debugging is         :ON
Openflow Stack WARN debugging is        :ON
Openflow Stack INFO debugging is        :OFF
Openflow IPC debugging is               :OFF
Openflow Hardware debugging is          :OFF
Openflow Config debugging is            :OFF
Openflow Forwarding debugging is        :OFF
```

debug openflow socket

Syntax: [no] debug openflow socket

This command displays OpenFlow socket interface messages. Command output resembles the following example.

```
Brocade# debug openflow socket
Mar 27 00:13:26.976 ces_socket_session_in_active_task: tcp_connect ITC error: 0
Mar 27 00:13:26.977 SO_SOCKIF: !!!! Connect closed by remote for task:(so.14.399)
Mar 27 00:13:26.977 SO_SOCKIF: socket task so.14.399 with sockfd:14 is closing
Mar 27 00:13:27.225 SO_SOCKIF: socket already closed for sockfd:14
Mar 27 00:13:30.795 SO_SOCKIF: alloc new sockfd:16
Mar 27 00:13:30.795 SO_SOCKIF: duplicate sock parentfd:15, childfd:16
Mar 27 00:13:30.796 SO_SOCKIF: starting child
task:(so.16.400),sockfd:16:local:0.0.0.0:6633
Mar 27 00:13:30.974 SO_SOCKIF: accept remote connection  10.25.128.243:35891 ->
10.25.128.246:6633:   sockfd:16:parentfd:15:task:so.16.400
Mar 27 00:13:31.225
SO_SOCKIF: task so.16.400 - sent 8 bytes
Mar 27 00:13:31.225 SO_SOCKIF: task so.16.400, read 8 bytes on sockfd 16
Mar 27 00:13:31.474 SO_SOCKIF: task so.16.400, read 8 bytes on sockfd 16
Mar 27 00:13:31.474 SO_SOCKIF: task so.16.400, read 64 bytes on sockfd 16
Mar 27 00:13:31.475 SO_SOCKIF: task so.16.400, read 8 bytes on sockfd 16
Mar 27 00:13:31.725
SO_SOCKIF: task so.16.400 - sent 8 bytes
```

debug openflow warn-stack

Syntax: [no] debug openflow warn-stack

This command displays OpenFlow stack warning messages. Command output resembles the following example.

```
Brocade# debug openflow warn-stack
Mar 26 23:41:18.819 01077|vlogshim|WARN|received vendor message for unknown
vendor 1234
Mar 26 23:43:15.516 01077|vlogshim|WARN|received OFPST_FLOW request of unknown
type 2
```

debug ip ssl

Syntax: [no] debug ip ssl

This command enables debugging of a Secure Socket Layer (SSL) connection. Command output resembles the following example.

```
Brocade# debug ip ssl
Dec  6 22:11:28 SSL:    ssl_open request to open SSL connection:
ssl:10.25.105.201:444
Dec  6 22:11:28 SSL[0]: set_ssl_client_session_free: Marking the session free
Dec  6 22:11:28 SSL[0]: ssl_open to ssl:10.25.105.201:444: successful
Dec  6 22:11:28 SSL:    TCP connection has been established (sock fd: 2)
Dec  6 22:11:28 SSL[0]: Initiating SSL SSL Handshake for sockfd 2
Dec  6 22:11:28 SSL[0]: ssl_complete_handler: for command 6, Status 2
Dec  6 22:11:28 SSL[0]: Connection is open. Starting SSL Handshake
Dec  6 22:11:29 SSL[0]: ssl_complete_handler: for command 3, Status 2
Dec  6 22:11:29 SSL[0]: SSL Handshake successful, Encryption has started
Dec  6 22:11:29 SSL[0]: ssl_connect for stream ssl:10.25.105.201:444 successful
Dec  6 22:11:29 SSL[0]: send request initiated for 8 bytes
Dec  6 22:11:29 SSL[0]: receive request initiated for 8 bytes
Dec  6 22:11:29 SSL[0]: ssl_complete_handler: for command 1, Status 2
Dec  6 22:11:29 SSL[0]: Send Success 8/8
Dec  6 22:11:29 SSL[0]: ssl_complete_handler: for command 2, Status 2
Dec  6 22:11:29 SSL[0]: receive success 8 bytes
Dec  6 22:14:52 SSL[0]: ssl_close request for ssl:10.25.105.201:444
Dec  6 22:14:52 SSL[0]: Closing the SSL connection
Dec  6 22:14:52 SSL[0]: TP Close request initiated
Dec  6 22:14:52 SSL[0]: Remote close connection called for socket 2
Dec  6 22:14:52 SSL[0]: Closing the SSL connection
Dec  6 22:14:53 SSL[0]: ssltcpCloseConnection: Closing
Dec  6 22:14:53 SSL[0]: ssltcpCloseStatus: Pending
Dec  6 22:15:52 SSL[0]: ssl_complete_handler: for command 4, Status 2
Dec  6 22:15:52 SSL[0]: Command:eTpAbort/eTpClose called!
Dec  6 22:15:52 SSL[0]: set_ssl_client_session_free: Marking the session free
```

Chapter

Technical Support Diagnostics

14

In this chapter

This chapter describes the **show tech-support** commands used for troubleshooting purposes.

show tech-support

The **show tech-support** command is useful when collecting a large amount of information about the Brocade NetIron XMR series and Brocade MLX series routers for troubleshooting purposes. The output of this command is used by technical support representatives when reporting a problem.

show tech-support

Syntax: **show tech-support** [**bfd** | **bgp**| **cluster** | **fib** | **ip** | **ipv6** | **isis** | **l2** | **mpls** | **oam** | **openflow** | **rtm** | **rtm6** | **sfm** | **system** | **tm**]

- **bfd** - Generates system and debugging information specific to Bidirectional Forwarding Detection (BFD) configurations.
- **bgp** - Generates system and debugging information specific to Border Gateway Protocol (BGP) configurations.
- **cluster** - Generates system and debugging information specific to cluster configurations.
- **fib** - Generates system and debugging information specific to Forwarding Information Base (FIB) configurations.
- **ip** - Generates system and debugging information specific to IPv4 configurations.
- **ipv6** - Generates system and debugging information specific to IPv6 configurations.
- **isis** - Generates system and debugging information specific to Intermediate System to Intermediate System (IS-IS) configurations.
- **l2** - Generates system and debugging information specific to Layer 2 protocols: RSTP,MSTP, STP, LACP, ERP, MRP, VLAN and VSRP.
- **mpls** - Generates system and debugging information specific to Multiprotocol Label Switching (MPLS) configurations.
- **oam** - Generates system and debugging information specific to Operations, Administration, and Maintenance (OAM) configurations.
- **openflow** - Generates system and debugging information specific to OpenFlow configurations.
- **rtm** - Generates system and debugging information specific to IPv4 routing table manager (RTM) configurations.

- **rtm6** - Generates system and debugging information specific to IPv6 RTM configurations.
- **sfm** - Generates system and debugging information specific to Switch Fabric Module (SFM) configurations.
- **system** - Generates system related debugging information.
- **tm** - Generates system and debugging information specific to Traffic Manager (TM) configurations.

The **show tech-support** command displays the output of several show commands at once. The output from this command varies depending on the router configuration. The format of the **show tech-support** * command output is modified to include a header for each of the sub-commands which gets called from the CLI to help in automated parsing and lookup of the output.

The header contains the following keywords:

- **BEGIN** - Used to indicate a sub-command which will begin execution next. If the command is an internal command, a textual description of the command is displayed.
- **CONTEXT** - Used to indicate where the sub-commands are executed such as INTERNAL, MP, MP-OS, LP, and LP-OS.
- **TIME-STAMP** - A time stamp, with millisecond granularity, helps in determining time difference between separate runs of the same command.

In addition to the header, the **show clock** command is run at the beginning and the end of the **show tech-support** * command for elapsed time calculation.

The default output of the **show tech-support** command includes the output of the following commands:

- **show version**
- **show running-config**
- **dm save**
- **show interfaces**
- **show statistics**
- **show logging**
- **show save**
- **show clock**
- **show bm-overflow**
- **show memory**
- **show memory pool**
- **show cpu**
- **show bm**
- **show emac**
- **dx 0 mib**
- **dx 1 mib**
- **show mq**
- **show cpu lp**
- **ipc show stat**
- **show flash**

- dir
- show media
- show redundancy
- show module
- show chassis
- show power
- show temperature
- show fan
- show ip ssh
- show ip ssh config
- show snmp
- show mac statistics
- show ip route summary
- show ipv6 route summary
- show mpls summary

The following example shows the truncated output of **show tech-support** default command.

```
Brocade# show tech-support
============================================================
BEGIN: show clock
CONTEXT: MP
TIME-STAMP: 3753400
============================================================
07:23:12.928 GMT+00 Thu Nov 29 2012

============================================================
BEGIN: show version
CONTEXT: MP
TIME-STAMP: 3754300
============================================================
System Mode: MLX
Chassis: NetIron 4-slot (Serial #: A32752FB0G, Part #: 35550-000C)
NI-X-SF Switch Fabric Module 1 (Serial #: S62643F0CW, Part #: 35548-102F)
FE 1: Type fe200, Version 2
Switch Fabric Module 1 Up Time is 1 hours 2 minutes 34 seconds
NI-X-SF Switch Fabric Module 2 (Serial #: S62646F0DY, Part #: 35548-102F)
FE 1: Type fe200, Version 2
Switch Fabric Module 2 Up Time is 1 hours 2 minutes 34 seconds
NI-X-SF Switch Fabric Module 3 (Serial #: S62641F0BZ, Part #: 35548-102E)
FE 1: Type fe200, Version 2
Switch Fabric Module 3 Up Time is 1 hours 2 minutes 34 seconds
==========================================================================
SL M1: NI-MLX-MR Management Module Active (Serial #: N02642F1FH, Part #:
35524-103K):
Boot : Version 5.4.0T165 Copyright (c) 1996-2012 Brocade Communications Systems,
Inc.
Compiled on Jun 15 2012 at 11:09:28 labeled as xmprm05400
 (521694 bytes) from boot flash
Monitor : Version 5.4.0T165 Copyright (c) 1996-2012 Brocade Communications
Systems, Inc.
Compiled on Jun 15 2012 at 11:08:50 labeled as xmb05400
 (528223 bytes) from code flash
```

```
IronWare : Version 5.5.0T163 Copyright (c) 1996-2012 Brocade Communications
Systems, Inc.
Compiled on Nov 28 2012 at 18:07:18 labeled as xmr05500b280
 (9087307 bytes) from Primary
Board ID : 00 MBRIDGE Revision : 36
Warning: Invalid MBRIDGE FPGA, expected revision 37
916 MHz Power PC processor 7447A (version 8003/0101) 166 MHz bus
512 KB Boot Flash (MX29LV040C), 32 MB Code Flash (MT28F128J3)
1024 MB DRAM INSTALLED
1024 MB DRAM ADDRESSABLE
Active Management uptime is 1 hours 2 minutes 34 seconds
==========================================================================
SL M2: NI-MLX-MR Management Module Standby (Serial #: N00414G00B, Part #:
35524-104A):
Boot : Version 5.4.0T165 Copyright (c) 1996-2012 Brocade Communications Systems,
Inc.
Compiled on Jun 15 2012 at 11:09:28 labeled as xmprm05400
 (521694 bytes) from boot flash
Monitor : Version 5.4.0T165 Copyright (c) 1996-2012 Brocade Communications
Systems, Inc.
Compiled on Jun 15 2012 at 11:08:50 labeled as xmb05400
 (528223 bytes) from code flash
IronWare : Version 5.5.0T163 Copyright (c) 1996-2012 Brocade Communications
Systems, Inc.
Compiled on Nov 28 2012 at 18:07:18 labeled as xmr05500b280
 (9087307 bytes) from Primary
Board ID : 00 MBRIDGE Revision : 36
916 MHz Power PC processor 7447A (version 8003/0101) 166 MHz bus
512 KB Boot Flash (MX29LV040C), 32 MB Code Flash (MT28F128J3)
1024 MB DRAM INSTALLED
1024 MB DRAM ADDRESSABLE
Standby Management uptime is 1 hours 1 minutes 59 seconds
==========================================================================
SL 2: BR-MLX-1GFx24-X 24-port 1GbE SFP Module (Serial #: BND0417G038, Part #:
60-1001892-08)
License: MLX-1Gx24-X-Upgrade (LID: dpfFJGMiFIN)
Boot : Version 5.1.0T175 Copyright (c) 1996-2012 Brocade Communications Systems,
Inc.
Compiled on Aug 11 2010 at 14:07:20 labeled as xmlprm05100
 (492544 bytes) from boot flash
Monitor : Version 5.4.0T175 Copyright (c) 1996-2012 Brocade Communications
Systems, Inc.
Compiled on Jun 15 2012 at 11:10:18 labeled as xmlb05400
 (526710 bytes) from code flash
IronWare : Version 5.5.0T177 Copyright (c) 1996-2012 Brocade Communications
Systems, Inc.
Compiled on Nov 28 2012 at 18:13:56 labeled as xmlp05500b280
 (7293242 bytes) from Primary
FPGA versions:
WARN: Invalid PBIF Version = 3.30, Build Time = 6/1/2012 11:09:00

Valid XPP Version = 7.23, Build Time = 6/7/2012 10:37:00

Valid STATS Version = 0.09, Build Time = 11/21/2010 14:52:00

BCM56512GMAC 0
BCM56512GMAC 1
666 MHz MPC 8541 (version 8020/0020) 333 MHz bus
512 KB Boot Flash (MX29LV040C), 16 MB Code Flash (MT28F128J3)
1024 MB DRAM, 8 KB SRAM, 286331153 Bytes BRAM
```

```
PPCR0: 1024K entries CAM, 16384K PRAM, 2048K AGE RAM
LP Slot 2 uptime is 1 hours 22 seconds
==========================================================================
All show version done

============================================================
BEGIN: show running-config
CONTEXT: MP
TIME-STAMP: 3754800
============================================================
Current configuration:
!
ver V5.5.0T163
module 2 br-mlx-24-port-1gf-x
!
!
!
!

!
no spanning-tree
.
.
.
```

show tech-support bfd

Syntax: show tech-support bfd [**session** *session-id*]

The **session** *session-id* parameter specifies the BFD session ID.

This command generates system and debugging information specific to BFD configurations.

The default output of the **show tech-support bfd** command includes the output of the following commands:

- **show bfd**
- **show bfd neighbor detail**
- **show bfd application**
- **show bfd mpls detail**
- **show bfd debug**

For the specified session ID, the output of the **show tech-support bfd** command includes the output of the **show bfd debug session** *session-id* command.

show tech-support bgp

Syntax: show tech-support bgp [[**vrf** *vrf-name* | **l2vpn vpls** | **vpnv4** | **ipv4 unicast** | **ipv4 multicast**] [**route** *ip-prefix*] | [[**ipv6 unicast** | **ipv6 multicast**] [**route** *ipv6-prefix*]]

- **vrf** *vrf-name* - Specifies the name of the Virtual Routing and Forwarding (VRF) instance for which you want to display the configuration information.
- **l2vpn vpls** - Displays information related to all the BGP Layer 2 Virtual Private Network (L2VPN) Virtual Private LAN Services (VPLS) routes.
- **vpnv4** - Displays information related to all the BGP Virtual Private Network Version 4 (VPNv4) routes.

- **ipv4 unicast** - Displays the IPv4 unicast route information for BGP routes.
- **ipv4 multicast** - Displays the IPv4 multicast route information for BGP routes.
- **route** *ip-prefix* - Specifies the IPv4 destination prefix and mask length for the BGP route.
- **ipv6 unicast** - Displays the IPv6 unicast route information for BGP routes.
- **ipv6 multicast** - Displays the IPv6 multicast route information for BGP routes.
- **route** *ipv6-prefix* - Specifies the IPv6 destination prefix and mask length for the BGP route.

This command generates system and debugging information specific to BGP configurations.

The default output of the **show tech-support bgp** command includes the output of the following commands:

- **show ip bgp config**
- **show ip bgp peer-group**
- **show ip bgp summary**
- **show ip bgp neighbors**
- **show ip bgp nexthop**
- **show ip bgp debug**
- **show ip bgp debug memory**
- **show ip bgp debug out-policy**
- **show ip bgp debug out-policy peer-list**
- **show ip tcp connections all-vrfs**
- **show ipv6 tcp connections**
- **show ip tcp debug**

For a particular route, the output of the **show tech-support bgp** command includes the output of the following commands:

- **show ip bgp route detail** *ip-prefix*
- **show ip bgp nexthop** *nexthop-addr*
- **show ip route** *nexthop-addr*

show tech-support fib

Syntax: show tech-support fib ipv4 *ip-address/prefix*

The *ip-address/prefix* variable specifies the destination IPv4 address and IP subnet mask length.

This command generates system and debugging information specific to FIB configurations. The command displays route details of an ipv4 address or prefix including information such as Management Processor (MP) route information, MP data structure information, Line Processor (LP) cache information, Programmable Random Access Memory (PRAM) information, and LP data structure information.

For a static or dynamically learned route, the output of the **show tech-support fib** command includes the output of the following commands:

- **show ip route** *ip-address/prefix* **debug**
- **show arp** *nexthop ip*
- **dm pram** *port index* **ip**

For a Generic Routing Encapsulation (GRE) tunnel route, the output of the **show tech-support fib** command includes the output of the following commands:

- **show ip route** *ip-address/prefix* **debug**
- **show ip-tunnels** *tunnel-id*
- **show nht-table**
- **dm pram** *port ingress-pram-index* **ip**

For a IP over MPLS (IPoMPLS) tunnel route, the output of the **show tech-support fib** command includes the output of the following commands:

- **show ip route** *ip-address/prefix* **debug**
- **show mpls lsp name** *name*
- **show nht-table**
- **dm pram** *port ingress-pram-index* **ip**

show tech-support ip ospf

Syntax: **show tech-support ip ospf** [**route** *ip-address* | **vrf** *vrf-name* | **vrf-all**]

- **route** *ip-address* - Specifies the destination IP address for the OSPF route.
- **vrf** *vrf-name* - Specifies the name of the VRF instance for which you want to display the configuration information.
- **vrf-all** - Displays the configuration information for all the VRF instances.

This command generates system and debugging information specific to IPv4 OSPF configurations.

The default output of the **show tech-support ip ospf** command includes the output of the following commands:

- **show ip ospf summary**
- **show ip ospf**
- **show ip ospf interface**
- **show ip ospf neighbor extensive**
- **show ip ospf border-routers**
- **show ip ospf database database-summary**
- **show ip ospf traffic**
- **show ip ospf debug**
- **show ip ospf debug memory**
- **show ip ospf debug appendix-e**
- **show ip ospf debug summary-routes**
- **show ip ospf debug misc**
- **show ip ospf config**
- **show ip ospf area**
- **show ip ospf sham-links**
- **show ip ospf virtual link**
- **show ip ospf virtual neighbor**
- **show ip ospf debug interface**

- **show ip ospf debug graceful-restart**
- **show ip ospf debug hello-timer**
- **show ip ospf debug opaque-link**
- **show ip ospf debug dspt**
- **show ip ospf debug lsp-shortcuts**

For a particular route, the output of the **show tech-support ip ospf** command includes the output of the following commands:

- **show ip ospf route** i*p-address*
- **show ip ospf database link-state link-state-id** *ip-address*

show tech-support ip vrrp

Syntax: show tech-support ip vrrp

This command generates system and debugging information specific to IPv4 VRRP configurations.

The default output of the **show tech-support ip vrrp** command includes the output of the following commands:

- **show ip vrrp brief**
- **show ip vrrp statistics**

show tech-support ip vrrp-extended

Syntax: show tech-support ip vrrp-extended

This command generates system and debugging information specific to IPv4 VRRP-extended (VRRP-E) configurations.

The default output of the **show tech-support ip vrrp-extended** command includes the output of the following commands:

- **show ip vrrp-extended brief**
- **show ip vrrp-extended statistics**

show tech-support ipv6 ospf

Syntax: show tech-support ipv6 ospf [**route** *ipv6-address* | **vrf** *vrf-name* | **vrf-all**]

- **route** *ipv6-address* - Specifies the destination IPv6 address for an OSPF route.
- **vrf** *vrf-name* - Specifies the name of the VRF instance for which you want to display the configuration information.
- **vrf-all** - Displays the configuration information for all VRF instances.

This command generates system and debugging information specific to IPv6 OSPF configurations.

The default output of the **show tech-support ipv6 ospf** command includes the output of the following commands:

- **show ipv6 ospf summary**
- **show ipv6 ospf**
- **show ipv6 ospf area**
- **show ipv6 ospf spf tree**

- **show ipv6 ospf interface**
- **show ipv6 ospf neighbor**
- **show ipv6 ospf mem**
- **show ipv6 ospf virtual-links**
- **show ipv6 ospf virtual-neighbor**

For a particular route, the output of the **show tech-support ipv6 ospf** command includes the output of the following commands:

- **show ipv6 ospf route** *ipv6-prefix*
- **show ipv6 ospf database prefix** *ipv6-prefix*

show tech-support ipv6 vrrp

Syntax: **show tech-support ipv6 vrrp**

This command generates system and debugging information specific to IPv6 VRRP configurations.

The default output of the **show tech-support ipv6 vrrp** command includes the output of the following commands:

- **show ipv6 vrrp brief**
- **show ipv6 vrrp statistics**

show tech-support ipv6 vrrp-extended

Syntax: **show tech-support ipv6 vrrp-extended**

This command generates system and debugging information specific to IPv6 VRRP-E configurations.

The default output of the **show tech-support ipv6 vrrp-extended** command includes the output of the following commands:

- **show ipv6 vrrp-extended brief**
- **show ipv6 vrrp-extended statistics**

show tech-support isis

Syntax: **show tech-support isis** [**route** [*ip-prefix* | *ipv6-prefix*]

- **route** *ip-prefix* - Specifies the IPv4 destination prefix and mask length for the IS-IS route.
- **route** *ipv6-prefix* - Specifies the IPv6 destination prefix and mask length for the IS-IS route.

This command generates system and debugging information specific to IS-IS configurations.

The default output of the **show tech-support isis** command includes the output of the following commands:

- **show isis configuration**
- **show isis interface**
- **show isis neighbor detail**
- **show isis hostname**
- **show isis spf-log detail**
- **show isis counts**

- show isis traffic
- show isis shortcut detail
- show isis debug
- show isis debug counters
- show isis debug memory
- show isis debug memory pool
- show isis debug pent
- show isis debug ip-nexthop-set
- show isis debug ipv6-pent
- show isis debug ipv6-nexthop-set
- show isis debug link-info
- show isis debug redis
- show isis debug adj-timer
- show isis debug lsp-timer
- show isis debug adj-options-order

For a particular route, the output of the **show tech-support isis** command includes the output of the following commands:

- **show isis debug route-info** *ip-prefix*
- **show isis debug v6route-info** *ipv6-prefix*

show tech-support mpls

Syntax: **show tech-support mpls** [**brief** | **detail** | **debug** |**ldp** | **rsvp** | **static-lsp**]

- **brief** - Collects brief technical support information for the MPLS configurations.
- **detail** - Collects detailed technical support information for the MPLS configurations.
- **debug** - Collects debugging information for the MPLS configurations.
- **ldp** - Collects Label Distribution Protocol (LDP) related debug information.
- **rsvp** - Collects Resource ReSerVation Protocol (RSVP) related debug information.
- **static-lsp** - Collects static Label Switched Path (LSP) related debug information.

The **show tech-support mpls** command generates system and debugging information specific to MPLS configurations. Instead of collecting output of all **show mpls *** and **show mpls debug *** using the **show tech-support mpls** command, you can collect information based on protocol and the level of detail you require. You can further use the **brief** | **detail** | **debug** options with **ldp** | **rsvp** | **static-lsp** options to specify the level of debugging information to be displayed for the specified protocol. For example, execute the **show tech-support mpls ldp detail** command to collect detailed technical support information for the MPLS LDP configurations.

The default output of the **show tech-support mpls** command includes the output of the following commands:

- show mpls autobw-template
- show mpls bfd
- show mpls bypass-lsp extensive
- show mpls config

- show mpls dynamic-bypass
- show mpls dynamic-bypass interface detail
- show mpls forwarding
- show mpls interfaces
- show mpls ldp
- show mpls ldp database
- show mpls ldp fec prefix
- show mpls ldp fec vc
- show mpls ldp interface
- show mpls ldp neighbor detail
- show mpls ldp path
- show mpls ldp peer detail
- show mpls ldp session detail
- show mpls ldp statistics
- show mpls ldp targeted-peer
- show mpls ldp tunnel detail
- show mpls lsp extensive
- show mpls memory
- show mpls path detail
- show mpls policy
- show mpls route
- show mpls rsvp
- show mpls rsvp interface detail
- show mpls rsvp neighbor
- show mpls rsvp session debug
- show mpls rsvp statistics
- show mpls statistics 6pe
- show mpls statistics label
- show mpls statistics ldp transit
- show mpls statistics lsp
- show mpls statistics oam
- show mpls statistics tunnel
- show mpls summary
- show mpls ted database detail
- show mpls debug counters
- show mpls debug cspf map
- show mpls debug cspf tables
- show mpls debug dynamic-bypass
- show mpls debug flooding-thresholds

- show mpls debug ldp lsp
- show mpls debug ldp peer
- show mpls debug ldp session
- show mpls debug lib
- show mpls debug lmgr
- show mpls debug lsp
- show mpls debug next-hop
- show mpls debug oam
- show mpls debug oam session all
- show mpls debug perf
- show mpls debug pw
- show mpls debug pw pw-status-tree
- show mpls debug rcp fec-history
- show mpls debug rcp lsp-cb
- show mpls debug rcp session
- show mpls debug rcp summary
- show mpls debug rcse
- show mpls debug rldf
- show mpls debug rldf-hist
- show mpls debug rri addr
- show mpls debug rri interface
- show mpls debug rri route
- show mpls debug rsir
- show mpls debug rsir del-pending-fec-tree
- show mpls debug rsir egress-fec-tree
- show mpls debug rsir query-fec-tree
- show mpls debug rsir unresolved-fec-tree
- show mpls debug rsvp
- show mpls debug secondary-lsp
- show mpls debug settings
- show mpls debug tunnel-interface

The output of the show tech-support mpls brief command includes the output of the following commands:

- show mpls config brief
- show mpls forwarding
- show mpls interfaces brief
- show mpls memory
- show mpls route
- show mpls statistics oam

- show mpls statistics tunnel
- show mpls summary
- show mpls autobw-template wide
- show mpls bfd
- show mpls bypass-lsp wide
- show mpls dynamic-bypass
- show mpls dynamic-bypass interface brief
- show mpls lsp wide
- show mpls path wide
- show mpls policy
- show mpls rsvp
- show mpls rsvp interface brief
- show mpls rsvp neighbor
- show mpls rsvp session wide
- show mpls rsvp session p2mp s2l
- show mpls forwarding p2mp
- show mpls rsvp statistics
- show mpls statistics lsp
- show mpls rsvp igp-sync
- show mpls rsvp igp-sync link
- show mpls rsvp igp-sync lsp
- show mpls ted database
- show mpls ldp
- show mpls ldp database
- show mpls ldp fec prefix
- show mpls ldp fec vc
- show mpls ldp interface
- show mpls ldp neighbor
- show mpls ldp path
- show mpls ldp peer brief
- show mpls ldp session brief
- show mpls statistics ldp tunnel
- show mpls ldp statistics
- show mpls ldp targeted-peer
- show mpls ldp tunnel brief
- show mpls static-lsp

The output of the **show tech-support mpls ldp brief** command includes the output of the following commands:

- show mpls ldp

- show mpls ldp database
- show mpls ldp fec prefix
- show mpls ldp fec vc
- show mpls ldp interface
- show mpls ldp neighbor
- show mpls ldp path
- show mpls ldp peer brief
- show mpls ldp session brief
- show mpls statistics ldp tunnel
- show mpls ldp statistics
- show mpls ldp targeted-peer
- show mpls ldp tunnel brief

The output of the **show tech-support mpls rsvp detail** command includes the output of the following commands:

- show mpls autobw-template detail
- show mpls bfd
- show mpls bypass-lsp detail
- show mpls dynamic-bypass
- show mpls dynamic-bypass interface detail
- show mpls lsp detail
- show mpls path detail
- show mpls policy
- show mpls rsvp
- show mpls rsvp interface detail
- show mpls rsvp neighbor
- show mpls rsvp session detail
- show mpls rsvp session p2mp detail
- show mpls forwarding p2mp detail
- show mpls rsvp statistics
- show mpls statistics lsp
- show mpls rsvp igp-sync
- show mpls rsvp igp-sync link detail
- show mpls rsvp igp-sync lsp detail
- show mpls ted database detail

The output of the **show tech-support mpls static-lsp debug** command includes the output of the following commands:

- show mpls label-range
- show mpls static-lsp debug

show tech-support openflow

Syntax: show tech-support openflow

This command generates system and debugging information specific to OpenFlow configurations.

The output of the **show tech-support openflow** command includes the output of the following commands:

- **show openflow datapath-id**
- **show openflow controller**
- **show openflow interface**
- **show openflow flows**
- **show versions**
- **show interfaces**
- **show statistics**
- **show running-config**
- **show logging**
- **show save**

show tech-support rtm

Syntax: show tech-support rtm [**vrf** *vrf-name* [**route** *ip-prefix*]]

- **vrf** *vrf-name* - Specifies the name of the VRF instance for which you want to display the configuration information.
- **route** *ip-prefix* - Specifies the IPv4 destination prefix and mask length for the RTM route.

This command generates system and debugging information specific to IPv4 RTM configurations.

The default output of the **show tech-support rtm** command includes the output of the following commands:

- **show ip rtm**
- **show ip route summary**
- **show ip route nexthop**
- **show ip rtm api**

For a particular route, the output of the **show tech-support rtm** command includes the output of the following commands:

- **show ip route** *ip-prefix* **debug**
- **show ip static route** *ip-prefix*
- **show ip ospf routes** *ip-address*
- **show isis routes** *ip-prefix*
- **show ip bgp routes** *ip-prefix*
- **show ip rip route** *ip-prefix*
- **show ip network** *ip-prefix*

show tech-support rtm6

Syntax: show tech-support rtm6 [**vrf** *vrf-name* [**route** *ipv6-prefix*]]

- **vrf** *vrf-name* - Specifies the name of the VRF instance for which you want to display the configuration information.
- **route** *ipv6-prefix* - Specifies the IPv6 destination prefix and mask length for the RTM route.

This command generates system and debugging information specific to IPv6 RTM configurations.

The default output of the **show tech-support rtm6** command includes the output of the following commands:

- **show ipv6 rtm**
- **show ipv6 route summary**
- **show ipv6 rtm api**

For a particular route, the output of the **show tech-support rtm6** command includes the output of the following commands:

- **show ipv6 route** *ipv6-prefix* **debug**
- **show ipv6 static route** *ipv6-prefix*
- **show ipv6 ospf routes** *ipv6-address*
- **show ipv6 isis routes** *ipv6-prefix*
- **show ipv6 bgp routes** *ipv6-prefix*
- **show ipv6 rip route** *ipv6-prefix*
- **show ipv6 network** *ipv6-prefix*

show tech-support sfm

Syntax: show tech-support sfm

This command generates system and debugging information specific to SFM and High-speed SFM (hSFM) configurations.

The output of the **show tech-support sfm** command includes the output of the following commands:

- **show sfm-mode**
- **show sfm-serdes-mode**
- **show sfm utilization**
- **show sfm logs**
- **show sfm table-sync statistics**
- **show sfm-links all detail**
- **show sfm all all queue-occupancy**
- **show sfm all all statistics drop**
- **show sfm critical registers**
- **show sfm all all link-status brief**
- **show internal fe queue occupancy**
- **show internal fe link status**

- show internal fe serdes status
- show internal fe critical registers

show tech-support system

Syntax: show tech-support system

This command is available on both the MP and the LP separately to fetch the system related information for debugging purpose.

The output of the **show tech-support system** command executed from the MP includes the output of the following commands:

- show clock
- show version
- show flash
- dir
- show redundancy
- show module
- show sfm-mode
- show sfm-utilization all
- show sfm-link all error
- show sfm-links all
- dm rw-snm all all get-link-stats brief
- show sfm logging
- show tm log
- show interface brief
- show lag brief
- show lag
- show stat brief
- show memory
- show task
- show cpu
- show cpu lp
- show np stat
- show tm stat all
- show tm non
- Ipc show stats
- ipc show dy-sync master
- dm pstat
- show temp
- show chassis
- show memory pool
- show memory

- show emac
- show bm

The output of the **show tech-support system** command executed from the LP includes the output of the following commands:

- show tm stats all
- show tm link serdes-error
- dm status rx me/x
- dm status tx me/x
- dm debug -stat me/x
- show packet pbif
- dm packet xpp
- dm tm nif-stats
- show memory pool
- show memory
- tsec 0 show
- show bm

show tech-support tm

Syntax: show tech-support tm [**slot** *slot_number*] [**critical-registers**]

- **slot** *slot_number* - Collects technical support information related to the TM for a specific slot.
- **critical-registers** - Collects critical registers information for the TM.

This command generates system and debugging information specific to TM configurations.

The output of the **show tech-support tm** command includes the output of the following commands:

- show tm statistics
- show tm non-empty queues
- show tm interrupts
- show tm link serdes errors
- show tm critical registers
- show tm logging

Tracing routing history

The IP Forwarding Information Base (FIB) trace feature records the history of routing events on line processor (LP) along with the allocated and de-allocated Content Addressable Memory (CAM) and Parameter Random Access Memory (PRAM) indexes. The trace information is used for debugging purpose.

NOTE
The debug tracing is supported only on LP.

trace-util ip

Syntax: trace-util ip [**change-size** | **clear-trace** | **dump** | **show-trace-ctrl** | **start-trace** | **stop-trace** | **stop-when-full** | **time-stamp**]

- **change-size** - Reinitializes trace and changes trace buffer size.
- **clear-trace** - Clears trace buffer entries.
- **dump** - Dumps trace buffer entries.
- **show-trace-ctrl** - Displays trace control information.
- **start-trace** - Starts tracing of routing events.
- **stop-trace** - Stops tracing of routing events.
- **stop-when-full** - Stops tracing of routing events if trace buffer is full.
- **time-stamp** - Allows to either skip or capture time stamp.

The **trace-util ip** command records the history of routing events.

trace-util ip dump

Syntax: trace-util ip dump [**all** | **grp-by-ip** | **ip-prefix**]

- **all** - Displays all trace buffer entries.
- **grp-by-ip** - Displays all trace buffer entries grouped by the IP address.
- **ip-prefix** - Displays all trace buffer entries for the specified IP prefix and mask.

The **trace-util ip dump all** command displays all trace buffer entries. Command output resembles the following example.

```
Brocade# trace-util ip dump all
Trace Contro Info:
   start_tag_id:0x0000
   stop_tag_id :0x0000
   num_entries :16384
   next_entry  :71
   p_log_buf   :0x25575000
   flag        :0x00000aa0
        Trace Control Block Initialized       (0x00000020)
        Trace Control Start Tracing is SET    (0x00000080)
70 Mar 27 16:27:14 (tag_id:0x0403) LP_IP_CAM_ADD_IP_ROUTE_CAM2PRAM_ENTRY:
   ipaddr.10.0.0.0/8, ppcr:7 cam_index:0x0004549e, pram_index:0x000000a4
69 Mar 27 16:27:14 (tag_id:0x0403) LP_IP_CAM_ADD_IP_ROUTE_CAM2PRAM_ENTRY:
   ipaddr.10.0.0.0/8, ppcr:6 cam_index:0x0004549e, pram_index:0x000000a4
68 Mar 27 16:27:14 (tag_id:0x0006) IP_ADD_NETWORK_ROUTE_TO_L3_CAM:
    ipaddr:10.0.0.0/8)
67 Mar 27 16:27:14 (tag_id:0x0403) LP_IP_CAM_ADD_IP_ROUTE_CAM2PRAM_ENTRY:
   ipaddr.10.1.1.1/32, ppcr:7 cam_index:0x0004549d, pram_index:0x000000a4
66 Mar 27 16:27:14 (tag_id:0x0403) LP_IP_CAM_ADD_IP_ROUTE_CAM2PRAM_ENTRY:
   ipaddr.10.1.1.1/32, ppcr:6 cam_index:0x0004549d, pram_index:0x000000a4
65 Mar 27 16:27:14 (tag_id:0x0006) IP_ADD_NETWORK_ROUTE_TO_L3_CAM:
    ipaddr:10.1.1.1/32)
64 Mar 27 16:27:08 (tag_id:0x0406) LP_IP_CAM_DEL_IP_ROUTE:
   ipaddr.10.1.1.1/32, ppcr:6 cam_index:0x0004549b
63 Mar 27 16:27:08 (tag_id:0x0406) LP_IP_CAM_DEL_IP_ROUTE:
   ipaddr.10.1.1.1/32, ppcr:7 cam_index:0x0004549b
62 Mar 27 16:27:08 (tag_id:0x0406) LP_IP_CAM_DEL_IP_ROUTE:
   ipaddr.10.0.0.0/8, ppcr:6 cam_index:0x0004549c
61 Mar 27 16:27:08 (tag_id:0x0406) LP_IP_CAM_DEL_IP_ROUTE:
```

```
   ipaddr.10.0.0.0/8, ppcr:7 cam_index:0x0004549c
```

The **trace-util ip dump grp-by-ip** command displays all trace buffer entries grouped by the IP address. Command output resembles the following example.

```
Brocade# trace-util ip dump grp-by-ip
Trace for IP address 0:0:0:0/0 count[2] start of list[20]
20 Jan 1 00:00:00 (tag_id:0x0403) LP_IP_CAM_ADD_IP_ROUTE_CAM2PRAM_ENTRY:
   ipaddr.0.0.0.0/0, ppcr:6 cam_index:0x00040800, pram_index:0x0000008a
21 Jan 1 00:00:00 (tag_id:0x0403) LP_IP_CAM_ADD_IP_ROUTE_CAM2PRAM_ENTRY:
   ipaddr.0.0.0.0/0, ppcr:7 cam_index:0x00040800, pram_index:0x0000008a
---------------------------------------------------------------------------------
---------
Trace for IP address 1:0:0:0/24 count[3] start of list[42]
42 Mar 27 16:18:03 (tag_id:0x0006) IP_ADD_NETWORK_ROUTE_TO_L3_CAM:
    ipaddr:10.0.0.0/24)
43 Mar 27 16:18:03 (tag_id:0x0403) LP_IP_CAM_ADD_IP_ROUTE_CAM2PRAM_ENTRY:
   ipaddr.10.0.0.0/24, ppcr:6 cam_index:0x00045498, pram_index:0x000000a0
44 Mar 27 16:18:03 (tag_id:0x0403) LP_IP_CAM_ADD_IP_ROUTE_CAM2PRAM_ENTRY:
   ipaddr.10.0.0.0/24, ppcr:7 cam_index:0x00045498, pram_index:0x000000a0
---------------------------------------------------------------------------------
---------
Trace for IP address 1:0:0:0/0 count[2] start of list[40]
40 Mar 27 16:18:03 (tag_id:0x0402) LP_IP_CAM_ADD_IP_ROUTE_CAM_ENTRY:
   ipaddr:10.0.0.0/0, ppcr:6, cam_level:0,cam_index:0x00075810,
pram_index:0x0000009f
```

The **trace-util ip dump ip-prefix** command displays all trace buffer entries for the specified IP prefix and mask. Command output resembles the following example.

```
Brocade# trace-util ip dump ip-prefix 10.0.0.0/24
display trace for IP address 1:0:0:0/24 [T=N]

Trace for IP address 1:0:0:0/24 count[3] start of list[42]

42 Mar 27 16:18:03 (tag_id:0x0006) IP_ADD_NETWORK_ROUTE_TO_L3_CAM:
    ipaddr:10.0.0.0/24)
43 Mar 27 16:18:03 (tag_id:0x0403) LP_IP_CAM_ADD_IP_ROUTE_CAM2PRAM_ENTRY:
   ipaddr.10.0.0.0/24, ppcr:6 cam_index:0x00045498, pram_index:0x000000a0
44 Mar 27 16:18:03 (tag_id:0x0403) LP_IP_CAM_ADD_IP_ROUTE_CAM2PRAM_ENTRY:
   ipaddr.10.0.0.0/24, ppcr:7 cam_index:0x00045498, pram_index:0x000000a0
```

trace-util ip change-size

Syntax: **trace-util ip change-size** *size*

The *size* variable specifies the size of trace buffer. The buffer size can take a value from 256 through 307200.

This LP command allows configuring trace buffer size. Command output resembles the following example.

```
Brocade# trace-util ip change-size 5000
successful reinit of trace
```

You can change the trace buffer size from Management Processor (MP) by entering a command such as the following.

```
Brocade# ip fib-trace-size 256 all
```

Syntax: **ip fib-trace-size** *size* [**all** | *slot*]

- *size* - Specifies the size of trace buffer. The buffer size can take a value from 256 through 307200.
- all - Configures the specified trace buffer size for all slots.
- *slot* - Configures the specified trace buffer size for a specific slot.

trace-util nexthop

Syntax: trace-util nexthop [clear-trace | dump-trace | remove-filter | set-filter | set-start-tag | set-stop-tag | show-trace-ctrl | start-trace | stop-trace | stop-when-full | time-stamp | translate-tag]

- clear-trace - Clears trace buffer entries.
- dump-trace - Dumps trace buffer entries.
- remove-filter - Removes filter key.
- set-filter - Sets filter key.
- set-start-tag - Allows to set up start trace tag ID.
- set-stop-tag - Allows to set up stop trace tag ID.
- show-trace-ctrl - Displays trace control information.
- start-trace - Starts tracing of routing events.
- stop-trace - Stops tracing of routing events.
- stop-when-full - Stops tracing of routing events if trace buffer is full.
- time-stamp - Allows to either skip or capture timestamp.
- translate-tag - Translates tag ID to string.

The trace-util nexthop dump-trace command displays the associated next-hop traces. Command output resembles the following example.

```
Brocade# trace-util nexthop dump-trace lifo
Trace Contro Info:
   start_tag_id:0x0000
   stop_tag_id :0x0000
   num_entries :4096
   next_entry  :6
   p_log_buf   :0x254f3000
   flag        :0x00000ca0
        Trace Control Block Initialized       (0x00000020)
        Trace Control Start Tracing is SET    (0x00000080)

0 Apr 23 11:04:25 (tag_id:0x0507) LP_IP_NEXTHOP_UPD_PRAM_VALID:update pram for
0x2ba8b810, PPCR 7 out intf ve 10,pram_index 0x0000009a
1 Apr 23 11:04:25 (tag_id:0x0507) LP_IP_NEXTHOP_UPD_PRAM_VALID:update pram for
0x2ba8b810, PPCR 6 out intf ve 10,pram_index 0x0000009a
2 Apr 23 11:04:25 (tag_id:0x0502) LP_IP_NEXTHOP_PRAM_ALLOC_REC:started 0x2ba8b810
3 Jan 1 00:00:00 (tag_id:0x0507) LP_IP_NEXTHOP_UPD_PRAM_VALID:update pram for
0x2ba8b210, nh 10.31.166.172 PPCR 7 out intf drop,out port 1/2 pram_index
0x0000008a
4 Jan 1 00:00:00 (tag_id:0x0507) LP_IP_NEXTHOP_UPD_PRAM_VALID:update pram for
0x2ba8b210, nh 10.31.166.172 PPCR 6 out intf drop,out port 1/2 pram_index
0x0000008a
5 Jan 1 00:00:00 (tag_id:0x0502) LP_IP_NEXTHOP_PRAM_ALLOC_REC:started 0x2ba8b210
Done!
```

The **trace-util nexthop time-stamp** command allows to either skip or capture timestamp in the next-hop traces. Command output resembles the following example when you specify **skip-capture** option.

```
Brocade# trace-util nexthop time-stamp skip-capture
Done!
Brocade# trace nexthop dump fifo
Trace Contro Info:
   start_tag_id:0x0000
   stop_tag_id :0x0000
   num_entries :16384
   next_entry  :5
   p_log_buf   :0x25836000
   flag        :0x00000cb0
        Trace Control Block Initialized       (0x00000020)
        Trace Control Skip Time Stamp capture (0x00000010)
        Trace Control Start Tracing is SET    (0x00000080)

0 (tag_id:0x0506) LP_IP_NEXTHOP_REL_REC:started 0x2e7ac410
1 (tag_id:0x0502) LP_IP_NEXTHOP_PRAM_ALLOC_REC:started 0x2e7ac410
2 (tag_id:0x0507) LP_IP_NEXTHOP_UPD_PRAM_VALID:update pram for 0x2e7ac410, PPCR 9
out intf eth 4/17,pram_index 0x0003001d
3 (tag_id:0x0507) LP_IP_NEXTHOP_UPD_PRAM_VALID:update pram for 0x2e7ac410, PPCR
10 out intf eth 4/17,pram_index 0x0003001d
4 (tag_id:0x0507) LP_IP_NEXTHOP_UPD_PRAM_VALID:update pram for 0x2e7ac410, PPCR
11 out intf eth 4/17,pram_index 0x0003001d
```

supportsave

The **supportsave** command is useful when collecting a large amount of debug information for troubleshooting purposes. The output of this command is used by technical support representatives when reporting a problem.

The **supportsave** command has the following advantages over the **show tech-support** command:

- Allows you to add additional commands to collect more data.
- Allows to transfer the collected data to an external server such as Trivial File Transfer Protocol (TFTP) server.

supportsave

Collects all relevant support information to a remote TFTP server location.

Syntax

supportsave *tftp_IPaddress* [**custom**]

supportsave show

supportsave [**add-cmd** *index cmd_string* | **del-cmd** *index* [**to** *index*] | **show**]

Parameters

tftp_IPaddress	Specifies the IPv4 address of a TFTP server.
custom	Executes all commands present in the default and custom commands list and sends the output to the specified TFTP server address. This is an optional parameter and is supported only in the privileged EXEC mode. If you do not specify the custom option, the **supportsave** command executes all commands present only in the default command list and sends the output to the specified TFTP server address.
show	Displays all commands present in the default and custom commands list. This option is supported in both the privileged EXEC mode and global configuration mode.
add-cmd	Allows to add a command to the custom commands list. You can add up to eight commands to the custom command list. This option is supported only in the global configuration mode.
index	Specifies the index in the custom commands list where an additional command need to be added. The valid range is from 1 through 8.
cmd_string	Specifies the privileged EXEC mode commands to be added to the custom commands list.
del-cmd	Allows to remove a command present at the given index or index range from the custom commands list. This option is supported only in the global configuration mode.
index	Specifies the index of a command in the custom commands list that needs to be removed. The valid range is from 1 through 8.
to *index*	To specify a range, you can use this option to specify the end index in the custom commands list. This is an optional parameter.

Modes

Privileged EXEC mode

Global configuration mode

Usage Guidelines

Parallel execution of the **supportsave** command from two different sessions is not allowed.

Parallel execution of the **supportsave** command and the **copy tftp/scp** commands is not allowed.

Only IPv4 is supported for the TFTP destination.

Commands added to the custom command list must be privileged EXEC mode MP commands.

Commands are not expanded while adding a command to the custom commands list.

Brocade recommends not to add any filters with the commands.

Modifying the custom commands list using the **supportsave add-cmd** or **supportsave del-cmd** commands is not allowed while the supportsave data collection is in progress.

The abort operation of the **supportsave** [**custom**] command using CTRL+c will only happen after execution of the current command in the list.

The **supportsave** command cannot be added to the custom commands list. Also, the commands which change the CLI mode (**exit**, **quit**, **end**) and commands which restart the router (**switchover**, **reload**, **reload-yes**) must not be added.

When the supportsave data collection is in progres, if a command produces more than 5 MB of output then the output will be chunked into 5 MB files and the chunk files are sent to the TFTP server.

During supportsave data collection, if the total output of all the commands exceeds 100 MB then the supportsave operation is terminated.

The filename format of the generated file is as follows:

"ss_<4randomchars>_<date_time_stamp>_<c or d>_<command_index_in_the_list>_
<current_chunk_number>.txt"

<c> stands for custom command and <d> stands for default command.

Examples The following is the sample output from the **supportsave show** command.

```
Brocade# supportsave show
DEFAULT COMMANDS
================
1: show tech-support

CUSTOM COMMANDS
===============
1: show run
2: show ip interface
3: <EMPTY>
4: show interface
5: show cam-partition usage
6: show run
7: show ip interface
8: show ip tcp connection
```

The following is the sample output from the **supportsave** command when the TFTP IP address is specified.

```
Brocade# supportsave 10.1.1.1
...............................................................................................................Supportsave completed
```

The content of the generated file includes the hostname, command name, and the corresponding output as shown in the following example. Also, an indication of whether the collected data is complete or partial is included at the end.

```
Brocade
===========show version===================
Started at 04:49:09.405 GMT+00 Tue Feb 14 2012
….
….
Ended at 04:49:09.405 GMT+00 Tue Feb 14 2012

===========show version [END]===================
```

The following syslog message is generated at the beginning of the **supportsave** data collection.

```
SUPPORTSAVE: Data collection has been initiated
```

Either one of the following syslog messages is generated at the end of the **supportsave** data collection.

```
SUPPORTSAVE: Data collection has been completed successfully
SUPPORTSAVE: Partial Data collection has been completed successfully
```

The following syslog message is generated when the TFTP connection fails.

```
SUPPORTSAVE: TFTP connection failed with server ip address 10.0.0.1
```

History

Release	Command History
Release 05.5.00	The command was introduced.

Related Commands

show tech-support

Diagnostic Command Index

C

D

H

I

L

P

S

T