DIAGNOSTIC GUIDE

ExtremeRouting MLX Series and XMR Series Diagnostic Guide, 06.2.00

Supporting NetIron OS 06.2.00

53-1005219-02
15 June 2018

© 2018, Extreme Networks, Inc. All Rights Reserved.

Extreme Networks and the Extreme Networks logo are trademarks or registered trademarks of Extreme Networks, Inc. in the United States and/or other countries. All other names are the property of their respective owners. For additional information on Extreme Networks Trademarks please see www.extremenetworks.com/company/legal/trademarks. Specifications and product availability are subject to change without notice.

Contents

Preface

Document conventions

The document conventions describe text formatting conventions, command syntax conventions, and important notice formats used in Extreme technical documentation.

Notes, cautions, and warnings

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

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

ATTENTION
An Attention statement indicates a stronger note, for example, to alert you when traffic might be interrupted or the device might reboot.

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

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

Text formatting conventions

Text formatting conventions such as boldface, italic, or Courier font may be used to highlight specific words or phrases.

Format	Description
bold text	Identifies command names.
	Identifies keywords and operands.
	Identifies the names of GUI elements.
	Identifies text to enter in the GUI.
italic text	Identifies emphasis.
	Identifies variables.
	Identifies document titles.
`Courier font`	Identifies CLI output.

Format	Description
	Identifies command syntax examples.

Command syntax conventions

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

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

Extreme resources

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

White papers, data sheets, and the most recent versions of Extreme software and hardware manuals are available at www.extremenetworks.com. Product documentation for all supported releases is available to registered users at www.extremenetworks.com/support/documentation.

Document feedback

Quality is our first concern at Extreme, 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.

You can provide feedback in two ways:

- Use our short online feedback form at http://www.extremenetworks.com/documentation-feedback-pdf/
- Email us at internalinfodev@extremenetworks.com

Provide the publication title, part number, and as much detail as possible, including the topic heading and page number if applicable, as well as your suggestions for improvement.

Contacting Extreme Technical Support

As an Extreme customer, you can contact Extreme Technical Support using one of the following methods: 24x7 online or by telephone. OEM customers should contact their OEM/solution provider.

If you require assistance, contact Extreme Networks using one of the following methods:

- GTAC (Global Technical Assistance Center) for immediate support
  - Phone: 1-800-998-2408 (toll-free in U.S. and Canada) or +1 408-579-2826. For the support phone number in your country, visit: www.extremenetworks.com/support/contact.
  - Email: support@extremenetworks.com. To expedite your message, enter the product name or model number in the subject line.
- GTAC Knowledge - Get on-demand and tested resolutions from the GTAC Knowledgebase, or create a help case if you need more guidance.
- The Hub - A forum for Extreme customers to connect with one another, get questions answered, share ideas and feedback, and get problems solved. This community is monitored by Extreme Networks employees, but is not intended to replace specific guidance from GTAC.
- Support Portal - Manage cases, downloads, service contracts, product licensing, and training and certifications.

Before contacting Extreme Networks for technical support, have the following information ready:

- Your Extreme Networks service contract number and/or serial numbers for all involved Extreme Networks products
- A description of the failure
- A description of any action(s) already taken to resolve the problem
- A description of your network environment (such as layout, cable type, other relevant environmental information)
- Network load at the time of trouble (if known)
- The device history (for example, if you have returned the device before, or if this is a recurring problem)
- Any related RMA (Return Material Authorization) numbers

About This Document

Supported hardware and software

The hardware platforms in the following table are currently supported for this publication.

TABLE 1 Supported devices

ExtremeRouting XMR Series	ExtremeRouting MLX Series
ExtremeRouting XMR 4000	ExtremeRouting MLXe-4
ExtremeRouting XMR 8000	ExtremeRouting MLXe-8
ExtremeRouting XMR 16000	ExtremeRouting MLXe-16
ExtremeRouting XMR 32000	ExtremeRouting MLXe-32

What's new in this document

Information has been added or updated to reflect new or modified debug commands for Extreme NetIron features or enhancements.

> **NOTE**
> On October 30, 2017, Extreme Networks, Inc. acquired the SLX-OS product line from Brocade Communications Systems, Inc. This transitional release includes references to both companies.

In this release of the Extreme NetIron debug command reference, not all debug commands supported on the NetIron devices are represented. All new commands since the NetIron Release 06.1.00 are included.

Using diagnostic commands

Using debug commands

This chapter describes how to use debug commands to monitor and troubleshoot the device configurations.

The debug commands are accessible from the Privileged EXEC mode in the NetIron command line interface (CLI). Most debug commands can be configured to send output to a specified destination.

When enabled, the debug commands can noticeably affect system performance. Many debug commands are specifically designed to be used in conjunction with calls to Extreme 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, which is relevant primarily to the Extreme engineering staff. These commands are not described in this document.

Generic debug commands

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

- **debug ?** - Generates a list of debug options.
- **show debug** - Shows all enabled debug settings.
- **debug destination** - Allows you to select an output destination: Telnet, SSH, console, or logging (default).

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 indicates that there are additional variables available and you have entered an incomplete command. Add a space and a question mark to your original command to view the additional variables.

```
device# debug vlan?
vlan Enable/Disable VLAN debug
```

show debug

The **show debug** command displays all the enabled debug functions. The output resembles the following example, which shows that VLAN debugging is enabled, with the console as the output destination.

```
device# show debug
Debug message destination: Console
VLAN
VLAN: debugging is on
```

debug destination

The **debug destination** command displays all the enabled debug functions. The output resembles the following example, which shows that VLAN debugging is enabled, with the console as the output destination.

- 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).

The **debug destination** command allows you to specify a destination for debugging output. The default destination is the system console, but you can redirect output to a syslog buffer, Telnet session, or SSH session.

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

```
device# show who
Console connections (by unit number):
1 established
4 minutes 29 seconds in idle
Telnet connections (inbound):
1 established, client ip address 172.31.0.1
you are connecting to this session
2 seconds in idle
2 closed
3 closed
4 closed
5 closed
Telnet connection (outbound):
6 closed
SSH connections:
1 closed
2 closed
3 closed
4 closed
```

To redirect the debug output to an active Telnet session, enter the following command.

```
device# debug destination telnet 1
```

Brief and detail debug options

Many debug commands can significantly impact system performance. If a debug command offers a brief or default option, see if it fills your needs before running the detailed option.

The reasons are as follows:

- Generating detailed output places an additional burden on system performance.
- In many cases, detailed results are more difficult to interpret than brief results.

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.

Debug commands A-G

debug access-list

Displays information about Layer 2 and Layer 4 ACLs.

Syntax

debug access-list { **accounting** | **ipv4** | **ipv6** | **l2** | **mirror** | **policy-based-routing** | **rate-limit** | **receive generic** | **lsp-out-acl** [**cam** | **generic**] }

no debug access-list { **accounting** | **ipv4** | **ipv6** | **l2** | **mirror** | **policy-based-routing** | **rate-limit** | **receive generic** | **lsp-out-acl** [**cam** | **generic**] }

Parameters

accounting
Displays information about inbound or outbound ACL accounting activity.

ipv4
Displays information about IPv4 access list activity.

ipv6
Displays information about IPv6 access list activity.

l2
Displays information about Layer 2 ACL activity.

mirror
Displays information about ACL mirroring activity.

policy-based-routing
Displays information about access list policy-based routing.

rate-limit
Displays information about rate limiting for IPv4 access lists.

receive generic
Displays information about generic access list receive activity.

lsp-out-acl
Displays LSP information for IPv4 access lists.

cam
Displays information about LSP CAM programming for IPv4 access lists.

generic
Displays generic LSP accounting information for IPv4 access lists.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

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

Examples

The following example displays inbound rate limit accounting for port 4/2.

```
device# debug access-list accounting
device# 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
```

The following example displays information about inbound activity for access group 101.

```
device# debug access-list ipv4
device# 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
```

The following example displays Layer 2 inbound accounting information for interface 4/2.

```
device# debug access-list l2
device# 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
```

The following example displays information about access list policy-based routing.

```
device# 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
```

The following example displays information about rate limiting for IPv4 access lists.

```
device# debug access-list rate-limit
device# rate-limit in access-group 101 500000000 750000000
device# 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
```

Displays information about generic access list receive activity.

```
device# debug access-list receive generic
device# 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
```

The following example displays information about LSP CAM programming for IPv4 access lists.

```
device# 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 following example displays generic LSP accounting information for IPv4 access lists.

```
device# 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 arp-guard

Displays all debug messages for ARP guard when enabled on LP.

Syntax

debug arp-guard [**all**]

no debug arp-guard [**all**]

Parameters

all

Displays all debug messages.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays all debug messages for ARP guard.

```
device# debug arp-guard
Jun 2 15:03:24.175 ARP_GUARD: Entered ag_ipc_callback_get_all_statistics()
Jun 2 15:03:24.175 ARP_GUARD: stats REQ revd for 2 ports & total buflen of 52
needed on this LP
Jun 2 15:03:24.175 ARP_GUARD: Stats REQ on port 2/3 for 2 VLANs rcvd
Jun 2 15:03:24.175 ARP_GUARD: REQ-ptr-offset = 0x0899f883, RESP-ptr-offset =
0x0803f780
Jun 2 15:03:24.175 ARP_GUARD:Equivalent (decimal) port=50,absolute_port_id=2
Jun 2 15:03:24.175 ARP_GUARD:Is[g_in_vlan_id ==
g_default_vlan_id]=YES,Is[untagged]=YES
Jun 2 15:03:24.175 ARP_GUARD: Entered ag_fill_statistics()
Jun 2 15:03:24.175 ARP_GUARD: stats-REQ rcvd for 1 VLANs
Jun 2 15:03:24.175 ARP_GUARD: Stats REQ on port 2/4 for 2 VLANs rcvd
Jun 2 15:03:24.175 ARP_GUARD: REQ-ptr-offset = 0x0899f890, RESP-ptr-offset =
0x0803f799
Jun 2 15:03:24.175 ARP_GUARD:Equivalent (decimal) port=51,absolute_port_id=3
Jun 2 15:03:24.175 ARP_GUARD:Is[g_in_vlan_id ==
g_default_vlan_id]=YES,Is[untagged]=YES
Jun 2 15:03:24.175 ARP_GUARD: Entered ag_fill_statistics()
Jun 2 15:03:24.175 ARP_GUARD: stats-REQ rcvd for 1 VLANs
Jun 2 15:03:24.175 ARP_GUARD: On VLAN 100, tot_arp_rcvd=10, arp_pkt_failed=10
Jun 2 15:03:24.175 2.Full-filled REQ for all(1) VLANs, so breaking
Jun 2 15:03:24.175 ARP_GUARD: Exiting ag_ipc_callback_get_all_statistics()
```

debug bfd

Displays information on BFD.

Syntax

debug bfd { **application** | **hitless-upgrade** | **ipc-error** | **ipc-event** | **itc** | **pbif** | **timer**}

no debug bfd { **application** | **hitless-upgrade** | **ipc-error** | **ipc-event** | **itc** | **pbif** | **timer**}

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example generates information about BFD application activity.

```
device# 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
```

The following example displays information about BFD hitless upgrade events.

```
device# 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
```

The following example generates information about BFD interprocess communication (IPC) errors.

```
device# debug ip ospf bfd
OSPF: BFD events debugging is on
device# debug bfd ipc-error
ipc-error: debugging is on
device# debug bfd ipc-event
ipc-event: debugging is on
device# 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
device#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
```

The following example displays information about BFD ITC activity.

```
device# debug bfd itc
itc: debugging is on
device# 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
```

The following example shows output from a session with **debug bfd pbif** and **debug bfd ipc-event** enabled.

```
device# debug bfd pbif
BFD: pbif debugging is on
device# 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
```

The following example shows the output from a session with **debug bfd timer**, **debug bfd ipc-event**, and **debug bfd pbif** enabled.

```
devuce# debug bfd timer
BFD: timer debugging is on
device# debug bfd pbif
BFD: pbif debugging is on
device# 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
```

debug cfm

Enables 802.1ag debugging.

Syntax

debug cfm {*level* | **ipc** | **packet** |**detail** }

no debug cfm {*level* | **ipc** | **packet** |**detail** }

Parameters

level

Specifies a value from 0 through 7. Setting the level to 7 enables debugging of all CFM packets.

ipc

Enables debugging and displays information for CFM IPC events.

packet

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

detail

Dumps the contents (time stamps) of the ETH-DM packet.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

```
device# 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
```

The following example displays information for CFM IPC events.

```
device# debug cfm ipc
device(config-cfm-md-test)# show vlan-group 1
device(config-cfm-md-test)# vlan-group 1
SYSLOG: <14>Dec 10 19:59:12 Dist1 VLAN Group 1: added by unknown from console
session.
device(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.
device(config-vlan-group-1)# exit
device(config)#cfm-enable
device(config-cfm)# domain-name test id 1 level 7
device(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
device(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
device(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
device(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
device(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
device(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
device(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
device(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
device(config-cfm-md-test-ma-3008)#end
device(config)#vlan-group 1
device(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
device(config-vlan-group-1)#end
SYSLOG: <14>Dec 10 20:01:06 Dist1 Security: running-config was changed from
console
```

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

```
device# debug cfm
device(config-cfm)# cfm-enable
device(config-cfm)# domain-name test id 1 level 7
device(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
device(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
device(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
device(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
device(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
device(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
```

debug cfm md

Enables debugging traces for the specified Maintenance Domain (MD).

Syntax

debug cfm md *md-name* **ma** *ma-name* [**mep-id** *mepid* | **rmep-id** *rmepid*]

no debug cfm md *md-name* **ma** *ma-name* [**mep-id** *mepid* | **rmep-id** *rmepid*]

Parameters

md-name
: Specifies the name of the Maintanance Domain.

ma *ma-name*
: 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).

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays debug logs for the specified MD at the MA level.

```
device# 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 following example displays debug logs for the specified MD at the MA and MEP level.

```
device# 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 following example displays debug logs for the specified MD at the MA and RMEP level.

```
device# 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 cluster actions

Displays the debug messages for MCT-related events, such as MCT FID port changes, and so on.

Syntax

debug cluster actions

no debug cluster actions

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays the debug messages for MCT-related events, such as MCT FID port changes, and so on.

```
device# 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

Displays the MCT active-passive messages for the cluster.

Syntax

debug cluster active-passive

no debug cluster active-passive

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays the MCT active-passive messages for the cluster.

```
device# 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

Displays all the CAM- or PRAM-related activities for MCT-related events in the LP.

Syntax

debug cluster cam

no debug cluster cam

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays all the CAM- or PRAM-related activities for MCT-related events in the LP.

```
device# 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 ccp fsm

Displays information related to Finite State Machine (FSM) transactions.

Syntax

debug cluster ccp fsm

no debug cluster ccp fsm

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays information related to Finite State Machine (FSM) transactions.

```
device# 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

Displays information specific to L2VPN timer packet exchange operation.

Syntax

debug cluster ccp packet

no debug cluster ccp packet

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays information specific to L2VPN timer packet exchange operation.

```
device# 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 config

Displays information about the Layer 2 Virtual Private Network (L2VPN) cluster configurations.

Syntax

debug cluster config

no debug cluster config

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays information about the Layer 2 Virtual Private Network (L2VPN) cluster configurations.

```
device# debug cluster config
device(config)#cluster c1
Nov 23 08:39:03 CLUSTER CONFIG: CLUSTER_CONFIG_ADD_CLUSTER ITC message received
device(config-cluster-c)# rbridge-id 4
Nov 23 08:39:03 CLUSTER CONFIG: CLUSTER_CONFIG_CLUSTER_RBRIDGE_ID ITC message
received
device(config-cluster-c)# session-vlan 31
Nov 23 08:39:03 CLUSTER CONFIG: CLUSTER_CONFIG_CLUSTER_MGMT_VLAN ITC message
received
device(config-cluster-c)# icl icl1 ethernet 1/3
Nov 23 08:39:03 CLUSTER CONFIG: CLUSTER_CONFIG_ADD_CLUSTER_ICL ITC message
received
device(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
device(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
device(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
device(config-cluster-c)# client c1
Nov 23 08:39:03 CLUSTER CONFIG: CLUSTER_CONFIG_ADD_CLUSTER_CLIENT ITC message
received
device(config-cluster-c-client-c1)# rbridge-id 101
Nov 23 08:39:03 CLUSTER CONFIG: CLUSTER_CONFIG_CLIENT_RBRIDGE_ID ITC message
received
device(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
device(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
device(config-cluster-c-client-c1)# client c2
Nov 23 08:39:03 CLUSTER CONFIG: CLUSTER_CONFIG_ADD_CLUSTER_CLIENT ITC message
received
device(config-cluster-c-client-c2)# rbridge-id 102
Nov 23 08:39:03 CLUSTER CONFIG: CLUSTER_CONFIG_CLIENT_RBRIDGE_ID ITC message
received
device(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 forwarding

Displays all the MCT forwarding-related events or messages in the MP that can affect traffic forwarding.

Syntax

debug cluster forwarding

no debug cluster forwarding

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays all the MCT forwarding-related events or messages in the MP that can affect traffic forwarding.

```
device# 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 fsm

Displays FSM information for the cluster.

Syntax

debug cluster fsm

no debug cluster fsm

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays FSM information for the cluster.

```
device# 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

Displays cluster FSM information for the client.

Syntax

debug cluster fsm client

no debug cluster fsm client

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays cluster FSM information for the client.

```
device# debug cluster fsm client 400
CLUSTER fsm debugging is now ON for client rbridge 400
device(config)#interface ethernet 3/13
device(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.
device(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)
device(config-if-e1000-3/13)#enable
SYSLOG: <46>Dec 5 15:55:55 Brocade PORT: 3/13 enabled by operator from console
session.
device(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 ipc

Debugs IPC to LP to sync forwarding information in the LP.

Syntax

debug cluster ipc

no debug cluster ipc

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

```
device# debug cluster ipc
CLUSTER IPC: MCT FID 0xa005 received for vlan 101
CLUSTER IPC: cluster 1, vlan mask change
```

debug cluster l2vpn

Displays the events specific to L2VPN operation.

Syntax

debug cluster l2vpn

no debug cluster l2vpn

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays the events specific to L2VPN operation

```
device# 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

Displays information related to MAC Database Update Protocol (MDUP) operation.

Syntax

debug cluster mdup

no debug cluster mdup

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays information related to MAC Database Update Protocol (MDUP) operation.

```
device# debug cluster mdup
CLUSTER mdup debugging is now ON
device# 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
```

debug destination

Allows you to select an output destination: Telnet, SSH, console, or logging (default).

Syntax

debug destination { **console** | **logging** | **telnet** *session-id* | **ssh** *session-id* } **task** *task-name*

no debug destination { **console** | **logging** | **telnet** *session-id* | **ssh** *session-id* } **task** *task-name*

Parameters

console
Configures to send the debug messages from the task specified to the console.

logging
Configures to send the debug messages from the task to the syslog server.

telnet *session-id*
Configures to send the debug messages from the task to a specified telnet session.

ssh *session-id*
Configures to send the debug messages from the task to a specified SSH session.

task *task-name*
Identifies the task from which the debug messages are configured to be sent to a particular destimation.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

The default is the system console, but you can redirect output to a syslog buffer, or a Telnet or SSH session.

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

Examples

The following example indicates that you have configured the console as the debug destination for the task OSPF.

```
device# debug destination console task ospf
Debug message task: ospf destination: Console
device# Apr 13 10:45:18.133 ip_fast_send: 10.44.44.44 -> 224.0.0.5: next hop
224.0.0.5, tx port 1793
Apr 13 10:45:18.133 ip_fast_send(224.0.0.5)pkt dropped : tx_port=1793 was=1793
Apr 13 10:45:18.133 IP(MEM): freeing packet at 0803fe42
```

The following example indicates that you have configured telnet session 2 as the debug destination for the task OSPF.

```
device# debug destination telnet 2 task ospf
Debug message task: ospf destination: telnet 2
device# Apr 13 10:45:18.133 ip_fast_send: 10.44.44.44 -> 224.0.0.5: next hop
224.0.0.5, tx port 1793
Apr 13 10:45:18.133 ip_fast_send(224.0.0.5)pkt dropped : tx_port=1793 was=1793
Apr 13 10:45:18.133 IP(MEM): freeing packet at 0803fe42
```

The following example indicates that you have configured SSH session 2 as the debug destination for the task OSPF.

```
device# debug destination ssh 2 task ospf
Debug message task: ospf destination: ssh 2
device# Apr 13 10:45:18.133 ip_fast_send: 10.44.44.44 -> 224.0.0.5: next hop
224.0.0.5, tx port 1793
Apr 13 10:45:18.133 ip_fast_send(224.0.0.5)pkt dropped : tx_port=1793 was=1793
Apr 13 10:45:18.133 IP(MEM): freeing packet at 0803fe42
```

The following example indicates that you have configured the syslog server as the debug destination for the task OSPF.

```
device# debug destination logging task ospf
Debug message task: ospf destination: logging
device# Apr 13 10:45:18.133 ip_fast_send: 10.44.44.44 -> 224.0.0.5: next hop
224.0.0.5, tx port 1793
Apr 13 10:45:18.133 ip_fast_send(224.0.0.5)pkt dropped : tx_port=1793 was=1793
Apr 13 10:45:18.133 IP(MEM): freeing packet at 0803fe42
```

debug destination buffer

Sends all debug messages to the buffer.

Syntax

debug destination buffer *ascii-string* [**create** | **show** [**lines** | **task** | **timestamp**] | **task**]

debug destination buffer list

no debug destination buffer list

Parameters

ascii-string
Sends all debug messages to the buffer specified by ASCII_string variable. The buffer must be pre-created.

create
Creates a buffer for collecting debug messages.

show
Displays the contents of the buffer.

lines
Filter the contents of the buffer based on number of lines to print.

task
Filter the contents of the buffer based on the task name.

timestamp
Filter the contents of the buffer based on timestamp.

task
Sets the debug destination for a task.

list
Lists all the buffers in the system along with the associated task names.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

You can use the **debug destination buffer create size** command to create a buffer of specific size. The buffer size can take a value ranging from 100 KB to 1 MB. You must specify the buffer size in bytes.

Examples

To create a buffer with name buff_1 of default size 500 KB, enter the following command.

```
device# debug destination buffer buff_1 create
```

To create a buffer with name buff_2 of size 1 MB (1048576 bytes), enter the following command.

```
device# debug destination buffer buff_2 create size 1048576
```

To send the debug messages from a specific task (OSPF) to a specific buffer (buff_2), enter a command such as the following.

```
device# debug destination buffer buff_2 task ospf
```

To display all the contents of a specific buffer (for example buff_2), enter the following command.

```
device# debug destination buffer buff_2 show
Buffer Dump: buff_2
Apr 13 10:44:29.702 IP/ARP: Pkt rcvd on port mgmt1, src IP = 10.37.73.142
Apr 13 10:44:29.703 IP/ARP: rcvd packet src 10.37.73.142 0000002a0800: dst
10.37.73.129 000000000000: Port mgmt1
Apr 13 10:44:29.703 find_arp_table_entry_and_index: ARP Entry not found for IP
10.37.73.142 Port mgmt1
Apr 13 10:44:29.703 find_arp_table_entry: ARP Entry not found for IP 10.37.73.142
Port mgmt1
Apr 13 10:44:30.986 IP/ARP: Pkt rcvd on port mgmt1, src IP = 10.37.73.174
Apr 13 10:44:30.986 IP/ARP: rcvd packet src 10.37.73.174 00000090e400: dst
10.37.73.129 000000000000: Port mgmt1
Apr 13 10:44:30.986 find_arp_table_entry_and_index: ARP Entry not found for IP
10.37.73.174 Port mgmt1
Apr 13 10:44:30.986 find_arp_table_entry: ARP Entry not found for IP 10.37.73.174
Port mgmt1
Apr 13 10:44:33.376 IP/ARP: Pkt rcvd on port mgmt1, src IP = 10.37.73.172
Apr 13 10:44:33.376 IP/ARP: rcvd packet src 10.37.73.172 00000090d000: dst
10.37.73.129 000000000000: Port mgmt1
Apr 13 10:44:33.376 find_arp_table_entry_and_index: ARP Entry not found for IP
10.37.73.172 Port mgmt1
Apr 13 10:44:33.376 find_arp_table_entry: ARP Entry not found for IP 10.37.73.172
Port mgmt1
Apr 13 10:44:35.933 ip_fast_send: 10.44.44.44 -> 224.0.0.5: next hop 224.0.0.5, tx
port 1793
Apr 13 10:44:35.933 ip_fast_send(224.0.0.5)pkt dropped : tx_port=1793 was=1793
Apr 13 10:44:35.933 IP(MEM): freeing packet at 0803fe42
Apr 13 10:44:38.734 MPLS: TNNL(a): Retry no. 1066, previously no router-id or
route
Apr 13 10:44:38.734 MPLS: TNNL(a): try signal LSP
Apr 13 10:44:44.634 ip_fast_send: 10.44.44.44 -> 224.0.0.5: next hop 224.0.0.5, tx
port 1793
Apr 13 10:44:44.634 ip_fast_send(224.0.0.5)pkt dropped : tx_port=1793 was=1793
Apr 13 10:44:44.634 IP(MEM): freeing packet at 0803fe42
Apr 13 10:44:56.034 ip_fast_send: 10.44.44.44 -> 224.0.0.5: next hop 224.0.0.5, tx
port 1793
Apr 13 10:44:56.034 ip_fast_send(224.0.0.5)pkt dropped : tx_port=1793 was=1793
Apr 13 10:44:56.034 IP(MEM): freeing packet at 0803fe42
Apr 13 10:44:57.267 IP/ARP: Pkt rcvd on port mgmt1, src IP = 10.37.73.165
Apr 13 10:44:57.267 IP/ARP: rcvd packet src 10.37.73.165 0000003b8600: dst
10.37.73.129 000000000000: Port mgmt1
Apr 13 10:44:57.268 find_arp_table_entry_and_index: ARP Entry not found for IP
10.37.73.165 Port mgmt1
Apr 13 10:44:57.268 find_arp_table_entry: ARP Entry not found for IP 10.37.73.165
Port mgmt1
Apr 13 10:45:06.834 ip_fast_send: 10.44.44.44 -> 224.0.0.5: next hop 224.0.0.5, tx
port 1793
Apr 13 10:45:06.834 ip_fast_send(224.0.0.5)pkt dropped : tx_port=1793 was=1793
Apr 13 10:45:06.834 IP(MEM): freeing packet at 0803fe42
Apr 13 10:45:08.734 MPLS: TNNL(a): Retry no. 1067, previously no router-id or
route
Apr 13 10:45:08.734 MPLS: TNNL(a): try signal LSP
```

To filter the contents of buffer (buff_2) based on task name (OSPF), enter the following command.

```
device# debug destination buffer buff_2 show task ospf
          Buffer: (buff_2) Filter: Task (ospf)
Apr 13 10:44:35.933 ip_fast_send: 10.44.44.44 -> 224.0.0.5: next hop 224.0.0.5, tx
port 1793
Apr 13 10:44:35.933 ip_fast_send(224.0.0.5)pkt dropped : tx_port=1793 was=1793
Apr 13 10:44:35.933 IP(MEM): freeing packet at 0803fe42
Apr 13 10:44:44.634 ip_fast_send(224.0.0.5)pkt dropped : tx_port=1793 was=1793
Apr 13 10:44:44.634 IP(MEM): freeing packet at 0803fe42
Apr 13 10:44:56.034 ip_fast_send: 10.44.44.44 -> 224.0.0.5: next hop 224.0.0.5, tx
port 1793
Apr 13 10:44:56.034 ip_fast_send(224.0.0.5)pkt dropped : tx_port=1793 was=1793
Apr 13 10:44:56.034 IP(MEM): freeing packet at 0803fe42
Apr 13 10:45:06.834 ip_fast_send: 10.44.44.44 -> 224.0.0.5: next hop 224.0.0.5, tx
port 1793
Apr 13 10:45:06.834 ip_fast_send(224.0.0.5)pkt dropped : tx_port=1793 was=1793
Apr 13 10:45:06.834 IP(MEM): freeing packet at 0803fe42
```

To filter the contents of buffer (buff_2) based on timestamp, enter a command such as the following.

```
device# debug destination buffer buff_2 show timestamp 04:13:10:45:06
         Buffer: (buff_2) Filter: Time Stamp (04:13:10:45:06)
Apr 13 10:45:06.834 ip_fast_send: 10.44.44.44 -> 224.0.0.5: next hop 224.0.0.5, tx
port 1793
Apr 13 10:45:06.834 ip_fast_send(224.0.0.5)pkt dropped : tx_port=1793 was=1793
Apr 13 10:45:06.834 IP(MEM): freeing packet at 0803fe42
Apr 13 10:45:08.734 MPLS: TNNL(a): Retry no. 1067, previously no router-id or
route
Apr 13 10:45:08.734 MPLS: TNNL(a): try signal LSP
```

To filter the contents of buffer (buff_2) based on number of lines to print (for example first five lines), enter a command such as the following.

```
device# debug destination buffer buff_2 show lines first 5
               Buffer: (buff_2) Dump First (5) lines
Apr 13 10:44:29.702 IP/ARP: Pkt rcvd on port mgmt1, src IP = 10.37.73.142
Apr 13 10:44:29.703 IP/ARP: rcvd packet src 10.37.73.142 0000002a0800: dst
10.37.73.129 000000000000: Port mgmt1
Apr 13 10:44:29.703 find_arp_table_entry_and_index: ARP Entry not found for IP
10.37.73.142 Port mgmt1
Apr 13 10:44:29.703 find_arp_table_entry: ARP Entry not found for IP 10.37.73.142
Port mgmt1
Apr 13 10:44:30.986 IP/ARP: Pkt rcvd on port mgmt1, src IP = 10.37.73.174
```

To filter the contents of buffer (buff_2) based on number of lines to print (for example last five lines), enter a command such as the following.

```
device# debug destination buffer buff_2 show lines last 5
Buffer: (buff_2) Dump Last (5) lines
Apr 13 10:45:06.834 ip_fast_send: 10.44.44.44 -> 224.0.0.5: next hop 224.0.0.5, tx
port 1793
Apr 13 10:45:06.834 ip_fast_send(224.0.0.5)pkt dropped : tx_port=1793 was=1793
Apr 13 10:45:06.834 IP(MEM): freeing packet at 0803fe42
Apr 13 10:45:08.734 MPLS: TNNL(a): Retry no. 1067, previously no router-id or
route
Apr 13 10:45:08.734 MPLS: TNNL(a): try signal LSP
```

The following example lists all the buffers in the system along with the associated task names.

```
device# debug destination buffer list
Buffer Name    Size       GLOBAL/Tasks
-----------    ----       --------------
buff_1         512000     ospf
buff_2         1048576   *GLOBAL*
```

debug destination show

Displays the configured debug destination for the specified task.

Syntax

debug destination show task *task-name*

no debug destination show task *task-name*

Parameters

task *task-name*

Identifies the task from which the debug messages are configured to be sent to a particular destimation.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays the configured debug destination for the specified task.

```
device# debug destination show task ospf
Task: ospf Debug Dest: Console
```

debug dot1x

Displays information about 802.1x authentication events, activity, and settings.

Syntax

debug dot1x [**all** | **dumpclass** | **events** | **fault** | **packets** | **port** [**all** | **event** | **timer**] *slot/port*| **state** | **timers**]

no debug dot1x [**all** | **dumpclass** | **events** | **fault** | **packets** | **port** [**all** | **event** | **timer**] *slot/port*| **state** | **timers**]

Parameters

all
: Displays general information about 802.1x activity for all ports. Provides a complete profile of the authentication process, including events, faults, timers, and packets and works globally across 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.

all
: Displays 802.1x event and timer information for a specified port.

event
: Displays 802.1x event information for a specified port.

timer
: Displays 802.1x timer settings for a specified port.

slot/port
: Specifies the interface.

state
: Displays 802.1x port state information.

timers
: Displays 802.1x timer information.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays a complete profile of the authentication process, including events, faults, timers, and packets.

```
device# 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
```

The following example displays the internal data structure.

```
device# 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
```

The following example 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.

```
device# 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)
```

The following example reports any kind of internal errors, such as out-of-memory, invalid RADIUS-response, and so on. This command works globally across all ports.

```
device# 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
```

The following example displays information about 802.1x packets.

```
device# 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
```

The following example displays event and timer information for a specified port.

```
device# 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
```

The following example displays information about the 802.1x state of a specified port.

```
device# 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
```

The following example enables debugging and displays information about 802.1x timers.

```
device# 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
```

debug dot1x-mka

Enables debugging of MACsec Key Agreement (MKA) and MAC security configurations settings.

Syntax

debug dot1x-mka [**all** | **cli** | **events** | **protocol** | **mkpdu-rx** | **mkpdu-tx** | **port** | **show** | **timers**]

Parameters

all
Displays information about MKA and MAC security configurations.

cli
Enables or disables MKA CLI debugging.

events
Enables or disables MKA event debugging.

protocol
Enables or disables MKA protocol debugging.

mkpdu-rx
Enables or disables MKA protocol data unit receive debugging.

mkpdu-tx
Enables or disables MKA protocol data unit transmit debugging.

port
Enables or disables MKA port debugging.

show
Displays the current MKA debugging settings.

timers
Enables or disables MKA timer debugging.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays information about MKA and MAC security configurations

```
device# debug dot1x-mka all
802.1X MKA All debugging ON
DUT5_8#Oct 27 06:38:31.395 [port 2/3] PORT UP event
Oct 27 06:38:31.395 [port 2/3] Activate Participant for cak, ckn
Oct 27 06:38:31.395 [port 2/3] Initializing MI and MN for the participant
Oct 27 06:38:31.395 Actor MI cd858915091cc5190795e2da
Oct 27 06:38:31.395 [port 2/3] Start Hello timer for transmitting the MKPDUs
Oct 27 06:38:31.395 [port 2/3] {Tx MKPDU } { Basic Param } Version: 1, Priority:
16, Desired: 1, Capable: 2, Length: 60
Oct 27 06:38:31.395 [port 2/3] { Tx MKPDU } { Basic Param } SCI: 0024389e2d320001
Oct 27 06:38:31.395 [port 2/3] { Tx MKPDU } { Basic Param } MI:
cd858915091cc5190795e2da, MN: 1, Agility: 0x0080c201
Oct 27 06:38:31.395 [port 2/3] { Tx MKPDU } { Basic Param } CKN:
0102030405060708090a0b0c0d0e0f10010203040506070809 0a0b0c0d0e0f10
Oct 27 06:38:31.395 [port 2/3] { Tx MKPDU } { SAK Use Param } Type: 3, LAN: 0,
OAN: 0, Delay: 0, Len: 0
Oct 27 06:38:31.395 [port 2/3] { Tx MKPDU } { SAK Use Param } LTX: 0, LRX: 0, OTX:
0, ORX: 0, PlainTX: 0, PlainRX: 0
Oct 27 06:38:31.395 [port 2/3] { Tx MKPDU } { SAK Use Param } LKI:
00000000000000000000000000000000, LLPN: 0x00000000
Oct 27 06:38:31.395 [port 2/3] { Tx MKPDU } { SAK Use Param } OKI:
00000000000000000000000000000000, OLPN: 0x00000000
Oct 27 06:38:31.395 Port 2/3 moved to blocking
Oct 27 06:38:32.256 [port 2/3] Received MKPDU - mkpdu len 84
Oct 27 06:38:32.256 [port 2/3] Key length 16 icv_pdu_len 86
Oct 27 06:38:32.257 [port 2/3] ICV Validation successful
Oct 27 06:38:32.257 [port 2/3] { Rx MKPDU } { Basic Param } Version: 1, Priority:
16, Desired: 1, Capable: 2, Length: 60
Oct 27 06:38:32.257 [port 2/3] { Rx MKPDU } { Basic Param } SCI: 0024388f6b920001
Oct 27 06:38:32.257 [port 2/3] { Rx MKPDU } { Basic Param } MI:
9b6742dce9f81a71d350e5c2, MN: 1, Agility: 0x0080c201
Oct 27 06:38:32.257 [port 2/3] { Rx MKPDU } { Basic Param } CKN:
0102030405060708090a0b0c0d0e0f10010203040506070809 0a0b0c0d0e0f10
Oct 27 06:38:32.257 [port 2/3] MKPDU - Processing basic parameter info len 60
Oct 27 06:38:32.257 [port 2/3] Received Peer mn = 1
Oct 27 06:38:32.257 no peers found for the participant
Oct 27 06:38:32.257 no peers found for the participant
Oct 27 06:38:32.257 Adding a potential peer
Oct 27 06:38:32.257 Peer 9b6742dce9f81a71d350e5c2
Oct 27 06:38:32.257 [port 2/3] {Tx MKPDU } { Basic Param } Version: 1, Priority:
16, Desired: 1, Capable: 2, Length: 60
Oct 27 06:38:32.257 [port 2/3] { Tx MKPDU } { Basic Param } SCI: 0024389e2d320001
Oct 27 06:38:32.257 [port 2/3] { Tx MKPDU } { Basic Param } MI:
cd858915091cc5190795e2da, MN: 2, Agility: 0x0080c201
Oct 27 06:38:32.257 [port 2/3] { Tx MKPDU } { Basic Param } CKN:
0102030405060708090a0b0c0d0e0f10010203040506070809 0a0b0c0d0e0f10
Oct 27 06:38:32.257 [port 2/3] { Tx MKPDU } { Potential Peer Param } MI:
9b6742dce9f81a71d350e5c2, MN: 1
Oct 27 06:38:32.257 [port 2/3] Include Potential peer with mn 1 to MKPDU
Oct 27 06:38:32.257 [port 2/3] { Tx MKPDU } { SAK Use Param } Type: 3, LAN: 0,
OAN: 0, Delay: 0, Len: 0
Oct 27 06:38:32.257 [port 2/3] { Tx MKPDU } { SAK Use Param } LTX: 0, LRX: 0, OTX:
0, ORX: 0, PlainTX: 0, PlainRX: 0
Oct 27 06:38:32.257 [port 2/3] { Tx MKPDU } { SAK Use Param } LKI:
00000000000000000000000000000000, LLPN: 0x00000000
Oct 27 06:38:32.257 [port 2/3] { Tx MKPDU } { SAK Use Param } OKI:
00000000000000000000000000000000, OLPN: 0x00000000
Oct 27 06:38:32.257 [port 2/3] CP current state change
Oct 27 06:38:32.280 [port 2/3] Received MKPDU - mkpdu len 104
Oct 27 06:38:32.280 [port 2/3] Key length 16 icv_pdu_len 106
Oct 27 06:38:32.280 [port 2/3] ICV Validation successful
Oct 27 06:38:32.280 [port 2/3] { Rx MKPDU } { Basic Param } Version: 1, Priority:
16, Desired: 1, Capable: 2, Length: 60
Oct 27 06:38:32.280 [port 2/3] { Rx MKPDU } { Basic Param } SCI: 0024388f6b920001
Oct 27 06:38:32.280 [port 2/3] { Rx MKPDU } { Basic Param } MI:
9b6742dce9f81a71d350e5c2, MN: 2, Agility: 0x0080c201
Oct 27 06:38:32.280 [port 2/3] { Rx MKPDU } { Basic Param } CKN:
```

```
0102030405060708090a0b0c0d0e0f1001020304050607080 90a0b0c0d0e0f10
Oct 27 06:38:32.280 [port 2/3] MKPDU - Processing basic parameter info len 60
Oct 27 06:38:32.280 [port 2/3] Received Peer mn = 2
Oct 27 06:38:32.280 [port 2/3] Re-start participant peer life timer
Oct 27 06:38:32.281 [port 2/3] Peer moved from potential to live
Oct 27 06:38:32.281 [port 2/3] {Tx MKPDU } { Basic Param } Version: 1, Priority:
16, Desired: 1, Capable: 2, Length: 60
Oct 27 06:38:32.281 [port 2/3] { Tx MKPDU } { Basic Param } SCI: 0024389e2d320001
Oct 27 06:38:32.281 [port 2/3] { Tx MKPDU } { Basic Param } MI:
cd858915091cc5190795e2da, MN: 3, Agility: 0x0080c201
Oct 27 06:38:32.281 [port 2/3] { Tx MKPDU } { Basic Param } CKN:
0102030405060708090a0b0c0d0e0f1001020304050607080 90a0b0c0d0e0f10
Oct 27 06:38:32.281 [port 2/3] { Tx MKPDU } { Live Peer Param } MI:
9b6742dce9f81a71d350e5c2, MN: 2
Oct 27 06:38:32.281 [port 2/3] Include Live peer with mn 2 to MKPDU
Oct 27 06:38:32.281 [port 2/3] { Tx MKPDU } { SAK Use Param } Type: 3, LAN: 0,
OAN: 0, Delay: 0, Len: 0
Oct 27 06:38:32.281 [port 2/3] { Tx MKPDU } { SAK Use Param } LTX: 0, LRX: 0, OTX:
0, ORX: 0, PlainTX: 0, PlainRX: 0
Oct 27 06:38:32.281 [port 2/3] { Tx MKPDU } { SAK Use Param } LKI:
00000000000000000000000000000000, LLPN: 0x00000000
Oct 27 06:38:32.281 [port 2/3] { Tx MKPDU } { SAK Use Param } OKI:
00000000000000000000000000000000, OLPN: 0x00000000
Oct 27 06:38:32.281 Stop MKA SAK delay timer
Oct 27 06:38:32.281 Current Key Server SCI 0000000000000000
Oct 27 06:38:32.281 key_server_priority 16 curr_key_server_priority 0
Oct 27 06:38:32.281 Key Server has been changed
Oct 27 06:38:32.281 macsec_capable 2, actor_macsec_capable 2
Oct 27 06:38:32.281 Selected MACsec capability 2
Oct 27 06:38:32.281 Oper cipher offset - 1
Oct 27 06:38:32.281 [port 2/3] MKA state is SECURED
Oct 27 06:38:32.281 [port 2/3] CP current state change
Oct 27 06:38:32.281 [port 2/3] CP SM state change from [change] to [secure]
Oct 27 06:38:32.281 Peer has better priorities so he will be ELECTED KEY SERVER
-priority 16
Oct 27 06:38:32.281 Key Server SCI 0024388f6b920001
Oct 27 06:38:32.281 [port 2/3] MKPDU - Processing potential peer parameter info
Oct 27 06:38:32.281 [port 2/3] {Rx MKPDU } { Potential Peer Param } MI:
cd858915091cc5190795e2da, MN: 2
Oct 27 06:38:32.281 [port 2/3] CP current state secure
Oct 27 06:38:32.282 [port 2/3] Received MKPDU - mkpdu len 104
Oct 27 06:38:32.282 [port 2/3] Key length 16 icv_pdu_len 106
Oct 27 06:38:32.282 [port 2/3] ICV Validation successful
Oct 27 06:38:32.282 [port 2/3] { Rx MKPDU } { Basic Param } Version: 1, Priority:
16, Desired: 1, Capable: 2, Length: 60
Oct 27 06:38:32.282 [port 2/3] { Rx MKPDU } { Basic Param } SCI: 0024388f6b920001
Oct 27 06:38:32.282 [port 2/3] { Rx MKPDU } { Basic Param } MI:
9b6742dce9f81a71d350e5c2, MN: 3, Agility: 0x0080c201
Oct 27 06:38:32.282 [port 2/3] { Rx MKPDU } { Basic Param } CKN:
0102030405060708090a0b0c0d0e0f1001020304050607080 90a0b0c0d0e0f10
Oct 27 06:38:32.282 [port 2/3] MKPDU - Processing basic parameter info len 60
Oct 27 06:38:32.282 [port 2/3] Received Peer mn = 3
Oct 27 06:38:32.282 [port 2/3] Re-start participant peer life timer
Oct 27 06:38:32.282 [port 2/3] MKPDU - Processing Live peer parameter info
Oct 27 06:38:32.283 [port 2/3] {Rx MKPDU } { Live Peer Param } MI:
cd858915091cc5190795e2da, MN: 3
Oct 27 06:38:32.283 [port 2/3] Peer has my recent MI and MN info, peer_mn 3
actor_mn 3
Oct 27 06:38:32.283 [port 2/3] CP current state secure
Oct 27 06:38:32.325 [port 2/3] Received MKPDU - mkpdu len 180
Oct 27 06:38:32.325 [port 2/3] Key length 16 icv_pdu_len 182
Oct 27 06:38:32.325 [port 2/3] ICV Validation successful
Oct 27 06:38:32.326 [port 2/3] { Rx MKPDU } { Basic Param } Version: 1, Priority:
16, Desired: 1, Capable: 2, Length: 60
Oct 27 06:38:32.326 [port 2/3] { Rx MKPDU } { Basic Param } SCI: 0024388f6b920001
Oct 27 06:38:32.326 [port 2/3] { Rx MKPDU } { Basic Param } MI:
9b6742dce9f81a71d350e5c2, MN: 4, Agility: 0x0080c201
Oct 27 06:38:32.326 [port 2/3] { Rx MKPDU } { Basic Param } CKN:
0102030405060708090a0b0c0d0e0f1001020304050607080 90a0b0c0d0e0f10
Oct 27 06:38:32.326 [port 2/3] MKPDU - Processing basic parameter info len 60
Oct 27 06:38:32.326 [port 2/3] Received Peer mn = 4
Oct 27 06:38:32.326 [port 2/3] Re-start participant peer life timer
```

```
Oct 27 06:38:32.326 [port 2/3] MKPDU - Processing Live peer parameter info
Oct 27 06:38:32.326 [port 2/3] {Rx MKPDU } { Live Peer Param } MI:
cd858915091cc5190795e2da, MN: 3
Oct 27 06:38:32.326 [port 2/3] Peer has my recent MI and MN info, peer_mn 3
actor_mn 3
Oct 27 06:38:32.326 [port 2/3] MKPDU - Processing SAK use parameter info
Oct 27 06:38:32.326 [port 2/3] { Rx MKPDU } { SAK Use Param } Type: 3, LAN: 0,
OAN: 0, Delay: 0, Len: 40
Oct 27 06:38:32.326 [port 2/3] { Rx MKPDU } { SAK Use Param } LTX: 0, LRX: 0, OTX:
0, ORX: 0, PlainTX: 0, PlainRX: 0
Oct 27 06:38:32.326 [port 2/3] { Rx MKPDU } { SAK Use Param } LKI:
9b6742dce9f81a71d350e5c200000001, LLPN: 0x00000001
Oct 27 06:38:32.326 [port 2/3] { Rx MKPDU } { SAK Use Param } OKI:
00000000000000000000000000000000, OLPN: 0x00000000
Oct 27 06:38:32.326 [port 2/3] SAK use info - no change of KI and AN
Oct 27 06:38:32.326 [port 2/3] Default cipher suite GCM-AES-128 to be used with
conf-offset 1
Oct 27 06:38:32.326 [port 2/3] {Rx MKPDU } { Distr SAK Param } Type: 4, AN: 0,
Offset: 1, Len: 28
Oct 27 06:38:32.326 [port 2/3] {Rx MKPDU } { Distr SAK Param } KN: 1, SAK: <Not
displayed>
Oct 27 06:38:32.326 Rx Distr SAK d73d455028d263e2ea1b38c89ce6ba13ba41c66b09217616
Oct 27 06:38:32.326 Unwrap SAK 117c078f206e50827537afe6b89caf3c
Oct 27 06:38:32.326 [port 2/3] Installing a new SAK Key - Distr AN 0
Oct 27 06:38:32.326 [port 2/3] CP current state secure
Oct 27 06:38:32.326 [port 2/3]Sending IPC to LP for creating rcv SA an 0
Oct 27 06:38:32.326 [port 2/3] Sending IPC to LP for creating transmit SA an 0
Oct 27 06:38:32.326 [port 2/3] CP SM state change from [secure] to [receive]
Oct 27 06:38:32.338 [port 2/3] Received MKPDU - mkpdu len 180
Oct 27 06:38:32.338 [port 2/3] Key length 16 icv_pdu_len 182
Oct 27 06:38:32.338 [port 2/3] ICV Validation successful
Oct 27 06:38:32.338 [port 2/3] { Rx MKPDU } { Basic Param } Version: 1, Priority:
16, Desired: 1, Capable: 2, Length: 60
Oct 27 06:38:32.338 [port 2/3] { Rx MKPDU } { Basic Param } SCI: 0024388f6b920001
Oct 27 06:38:32.338 [port 2/3] { Rx MKPDU } { Basic Param } MI:
9b6742dce9f81a71d350e5c2, MN: 5, Agility: 0x0080c201
Oct 27 06:38:32.338 [port 2/3] { Rx MKPDU } { Basic Param } CKN:
0102030405060708090a0b0c0d0e0f10010203040506070809 0a0b0c0d0e0f10
Oct 27 06:38:32.338 [port 2/3] MKPDU - Processing basic parameter info len 60
Oct 27 06:38:32.338 [port 2/3] Received Peer mn = 5
Oct 27 06:38:32.338 [port 2/3] Re-start participant peer life timer
Oct 27 06:38:32.338 [port 2/3] MKPDU - Processing Live peer parameter info
Oct 27 06:38:32.338 [port 2/3] {Rx MKPDU } { Live Peer Param } MI:
cd858915091cc5190795e2da, MN: 3
Oct 27 06:38:32.338 [port 2/3] Peer has my recent MI and MN info, peer_mn 3
actor_mn 3
Oct 27 06:38:32.338 [port 2/3] MKPDU - Processing SAK use parameter info
Oct 27 06:38:32.338 [port 2/3] { Rx MKPDU } { SAK Use Param } Type: 3, LAN: 0,
OAN: 0, Delay: 0, Len: 40
Oct 27 06:38:32.338 [port 2/3] { Rx MKPDU } { SAK Use Param } LTX: 0, LRX: 1, OTX:
0, ORX: 0, PlainTX: 0, PlainRX: 0
Oct 27 06:38:32.338 [port 2/3] { Rx MKPDU } { SAK Use Param } LKI:
9b6742dce9f81a71d350e5c200000001, LLPN: 0x00000001
Oct 27 06:38:32.338 [port 2/3] { Rx MKPDU } { SAK Use Param } OKI:
00000000000000000000000000000000, OLPN: 0x00000000
Oct 27 06:38:32.338 [port 2/3] SAK use info - no change of KI and AN
Oct 27 06:38:32.338 [port 2/3] Default cipher suite GCM-AES-128 to be used with
conf-offset 1
Oct 27 06:38:32.338 [port 2/3] {Rx MKPDU } { Distr SAK Param } Type: 4, AN: 0,
Offset: 1, Len: 28
Oct 27 06:38:32.339 [port 2/3] {Rx MKPDU } { Distr SAK Param } KN: 1, SAK: <Not
displayed>
Oct 27 06:38:32.339 [port 2/3] Recieved same SAK Key - Distr AN 0
Oct 27 06:38:32.339 [port 2/3] CP current state receive
Oct 27 06:38:32.347 [port 2/3] IPC response - RX SA is ready to receive
Oct 27 06:38:32.347 [port 2/3] CP current state receive
Oct 27 06:38:32.347 [port 2/3] CP SM state change from [receive] to [receiving]
Oct 27 06:38:32.347 [port 2/3] CP current state receiving
Oct 27 06:38:32.347 [port 2/3] CP SM state change from [receiving] to [ready]
Oct 27 06:38:32.347 [port 2/3]CP has set new_info, so transmit MKPDU
Oct 27 06:38:32.348 [port 2/3] {Tx MKPDU } { Basic Param } Version: 1, Priority:
16, Desired: 1, Capable: 2, Length: 60
```

```
Oct 27 06:38:32.348 [port 2/3] { Tx MKPDU } { Basic Param } SCI: 0024389e2d320001
Oct 27 06:38:32.348 [port 2/3] { Tx MKPDU } { Basic Param } MI:
cd858915091cc5190795e2da, MN: 4, Agility: 0x0080c201
Oct 27 06:38:32.348 [port 2/3] { Tx MKPDU } { Basic Param } CKN:
0102030405060708090a0b0c0d0e0f10010203040506070809 0a0b0c0d0e0f10
Oct 27 06:38:32.348 [port 2/3] { Tx MKPDU } { Live Peer Param } MI:
9b6742dce9f81a71d350e5c2, MN: 5
Oct 27 06:38:32.348 [port 2/3] Include Live peer with mn 5 to MKPDU
Oct 27 06:38:32.348 [port 2/3] { Tx MKPDU } { SAK Use Param } Type: 3, LAN: 0,
OAN: 0, Delay: 0, Len: 40
Oct 27 06:38:32.348 [port 2/3] { Tx MKPDU } { SAK Use Param } LTX: 0, LRX: 1, OTX:
0, ORX: 0, PlainTX: 0, PlainRX: 0
Oct 27 06:38:32.348 [port 2/3] { Tx MKPDU } { SAK Use Param } LKI:
9b6742dce9f81a71d350e5c200000001, LLPN: 0x00000000
Oct 27 06:38:32.348 [port 2/3] { Tx MKPDU } { SAK Use Param } OKI:
00000000000000000000000000000000, OLPN: 0x00000000
Oct 27 06:38:32.486 [port 2/3] Received MKPDU - mkpdu len 148
Oct 27 06:38:32.486 [port 2/3] Key length 16 icv_pdu_len 150
Oct 27 06:38:32.486 [port 2/3] ICV Validation successful
Oct 27 06:38:32.486 [port 2/3] { Rx MKPDU } { Basic Param } Version: 1, Priority:
16, Desired: 1, Capable: 2, Length: 60
Oct 27 06:38:32.486 [port 2/3] { Rx MKPDU } { Basic Param } SCI: 0024388f6b920001
Oct 27 06:38:32.486 [port 2/3] { Rx MKPDU } { Basic Param } MI:
9b6742dce9f81a71d350e5c2, MN: 6, Agility: 0x0080c201
Oct 27 06:38:32.486 [port 2/3] { Rx MKPDU } { Basic Param } CKN:
0102030405060708090a0b0c0d0e0f10010203040506070809 0a0b0c0d0e0f10
Oct 27 06:38:32.486 [port 2/3] MKPDU - Processing basic parameter info len 60
Oct 27 06:38:32.486 [port 2/3] Received Peer mn = 6
Oct 27 06:38:32.486 [port 2/3] Re-start participant peer life timer
Oct 27 06:38:32.486 [port 2/3] MKPDU - Processing Live peer parameter info
Oct 27 06:38:32.486 [port 2/3] {Rx MKPDU } { Live Peer Param } MI:
cd858915091cc5190795e2da, MN: 4
Oct 27 06:38:32.486 [port 2/3] Peer has my recent MI and MN info, peer_mn 4
actor_mn 4
Oct 27 06:38:32.486 [port 2/3] MKPDU - Processing SAK use parameter info
Oct 27 06:38:32.486 [port 2/3] { Rx MKPDU } { SAK Use Param } Type: 3, LAN: 0,
OAN: 0, Delay: 0, Len: 40
Oct 27 06:38:32.486 [port 2/3] { Rx MKPDU } { SAK Use Param } LTX: 1, LRX: 1, OTX:
0, ORX: 0, PlainTX: 0, PlainRX: 0
Oct 27 06:38:32.486 [port 2/3] { Rx MKPDU } { SAK Use Param } LKI:
9b6742dce9f81a71d350e5c200000001, LLPN: 0x00000001
Oct 27 06:38:32.486 [port 2/3] { Rx MKPDU } { SAK Use Param } OKI:
00000000000000000000000000000000, OLPN: 0x00000000
Oct 27 06:38:32.486 [port 2/3] SAK use info - no change of KI and AN
Oct 27 06:38:32.486 [port 2/3] Server has started transmitting, peers too need to
start tx
Oct 27 06:38:32.486 [port 2/3] CP current state ready
Oct 27 06:38:32.486 Port 2/3 moved to non-blocking
Oct 27 06:38:32.486 [port 2/3] Enabling transmit SA an 0
Oct 27 06:38:32.486 [port 2/3] CP SM state change from [ready] to [transmit]
Oct 27 06:38:32.487 [port 2/3] IPC response- TX SA is transmittng
Oct 27 06:38:32.487 [port 2/3] CP current state transmit
Oct 27 06:38:32.487 [port 2/3] CP SM state change from [transmit] to
[transmitting]
Oct 27 06:38:32.487 [port 2/3]CP has set new_info, so transmit MKPDU
Oct 27 06:38:32.487 [port 2/3] {Tx MKPDU } { Basic Param } Version: 1, Priority:
16, Desired: 1, Capable: 2, Length: 60
Oct 27 06:38:32.487 [port 2/3] { Tx MKPDU } { Basic Param } SCI: 0024389e2d320001
Oct 27 06:38:32.487 [port 2/3] { Tx MKPDU } { Basic Param } MI:
cd858915091cc5190795e2da, MN: 5, Agility: 0x0080c201
Oct 27 06:38:32.487 [port 2/3] { Tx MKPDU } { Basic Param } CKN:
0102030405060708090a0b0c0d0e0f10010203040506070809 0a0b0c0d0e0f10
Oct 27 06:38:32.487 [port 2/3] { Tx MKPDU } { Live Peer Param } MI:
9b6742dce9f81a71d350e5c2, MN: 6
Oct 27 06:38:32.487 [port 2/3] Include Live peer with mn 6 to MKPDU
Oct 27 06:38:32.487 [port 2/3] { Tx MKPDU } { SAK Use Param } Type: 3, LAN: 0,
OAN: 0, Delay: 0, Len: 40
Oct 27 06:38:32.487 [port 2/3] { Tx MKPDU } { SAK Use Param } LTX: 1, LRX: 1, OTX:
0, ORX: 0, PlainTX: 0, PlainRX: 0
Oct 27 06:38:32.487 [port 2/3] { Tx MKPDU } { SAK Use Param } LKI:
9b6742dce9f81a71d350e5c200000001, LLPN: 0x00000000
Oct 27 06:38:32.487 [port 2/3] { Tx MKPDU } { SAK Use Param } OKI:
```

```
00000000000000000000000000000000, OLPN: 0x00000000
Oct 27 06:38:32.488 [port 2/3] CP current state transmitting
Oct 27 06:38:32.488 [port 2/3] CP SM state change from [transmitting] to [retire]
```

The following example enables MKA CLI debugging.

```
device# debug dot1x-mka cli
DOT1X-MKA cli debugging ON
device#conf t
Oct 27 06:26:47.953 MKA ckn - ports ethe 2/2 to 2/16 ethe 2/19 to 2/20
Oct 27 06:26:47.953 MKA ckn - ports ethe 2/17 to 2/18
Oct 27 06:26:47.953 MKA no group - ports ethe 4/3
device(config-dot1x-mka-2/8)#mka-cfg-group 15
device(config-dot1x-mka-2/8)#pre-shared-key 0102030405060708090a0b0c0d0e0f12
key-name 0102030405060708090a0b0c0d0e0f1001020304050607080 90a0b0c0d0e0f10
Oct 27 06:29:55.303 Start dot1x_cu_send_config_request
Oct 27 06:29:55.303 Start DOT1X_MKA_PORT_CAK_CKN_CONFIG
Oct 27 06:29:55.304 End DOT1X_MKA_PORT_CAK_CKN_CONFIG
```

The following example enables MKA protocol debugging.

```
device# debug dot1x-mka protocol
DOT1X-MKA proto debugging ON
device#Oct 27 06:33:40.876 [port 2/3] Update Next PN 81239
device#Oct 27 06:33:41.589 [port 2/3] CP current state retire
Oct 27 06:33:42.882 [port 2/3] Update Next PN 81240
device#Oct 27 06:33:43.620 [port 2/3] CP current state retire
device#Oct 27 06:33:44.876 [port 2/3] Update Next PN 81242
Oct 27 06:33:45.587 [port 2/3] CP current state retire
Oct 27 06:33:46.876 [port 2/3] Update Next PN 81243
coOct 27 06:33:47.589 [port 2/3] CP current state retirenf t
device(config)#Oct 27 06:33:48.880 [port 2/3] Update Next PN 81244
Oct 27 06:33:49.620 [port 2/3] CP current state retire
Oct 27 06:33:50.876 [port 2/3] Update Next PN 81245
Oct 27 06:33:51.588 [port 2/3] CP current state retire
Oct 27 06:33:52.876 [port 2/3] Update Next PN 81246
Oct 27 06:33:53.589 [port 2/3] CP current state retire
Oct 27 06:33:54.880 [port 2/3] Update Next PN 81247
Oct 27 06:33:55.621 [port 2/3] CP current state retire
Oct 27 06:33:56.877 [port 2/3] Update Next PN 81248
Oct 27 06:33:57.589 [port 2/3] CP current state retire
device(config)#iOct 27 06:33:58.877 [port 2/3] Update Next PN 81249
nt etOct 27 06:33:59.589 [port 2/3] CP current state retireh 2/3
device(config-if-e10000-2/3)#Oct 27 06:34:00.881 [port 2/3] Update Next PN 81250
device(config-if-e10000-2/3)#
device(config-if-e10000-2/3)#Oct 27 06:34:01.621 [port 2/3] CP current
stateretire
device(config-if-e10000-2/3)#
device(config-if-e10000-2/3)#dis
device(config-if-e10000-2/3)#Oct 27 06:34:02.825 [port 2/3] PORT DOWN event
Oct 27 06:34:02.825 Port 2/3 moved to blocking
Oct 27 06:34:02.825 [port 2/3] Delete participant for port 2/3
Oct 27 06:34:02.825 Deleting all peers
Oct 27 06:34:02.825 [port 2/3] Logon process initialized
Oct 27 06:34:02.825 Port 2/3 moved to blocking
Oct 27 06:34:02.825 [port 2/3] CP current state init
Oct 27 06:34:02.825 [port 2/3] Deleting all SAs
Oct 27 06:34:02.825 [port 2/3] CP SM state change from [init] to [change]
ena
device(config-if-e10000-2/3)#Oct 27 06:34:03.927 [port 2/3] PORT UP event
Oct 27 06:34:03.928 [port 2/3] Activate Participant for cak, ckn
Oct 27 06:34:03.928 [port 2/3] Initializing MI and MN for the participant
Oct 27 06:34:03.928 Actor MI f2570d864df4107aff60f591
Oct 27 06:34:03.928 Port 2/3 moved to blocking
Oct 27 06:34:04.771 no peers found for the participant
Oct 27 06:34:04.771 no peers found for the participant
Oct 27 06:34:04.771 [port 2/3] CP current state change
Oct 27 06:34:04.775 Current Key Server SCI 0000000000000000
Oct 27 06:34:04.775 key_server_priority 16 curr_key_server_priority 0
Oct 27 06:34:04.775 Key Server has been changed
Oct 27 06:34:04.775 macsec_capable 2, actor_macsec_capable 2
Oct 27 06:34:04.775 Selected MACsec capability 2
Oct 27 06:34:04.775 Oper cipher offset - 1
Oct 27 06:34:04.775 [port 2/3] MKA state is SECURED
Oct 27 06:34:04.775 [port 2/3] CP current state change
Oct 27 06:34:04.775 [port 2/3] CP SM state change from [change] to [secure]
Oct 27 06:34:04.775 Peer has better priorities so he will be ELECTED KEY SERVER
-priority 16
Oct 27 06:34:04.775 Key Server SCI 0024388f6b920001
Oct 27 06:34:04.775 [port 2/3] CP current state secure
Oct 27 06:34:04.858 [port 2/3] CP current state secure
Oct 27 06:34:04.889 [port 2/3] CP current state secure
Oct 27 06:34:04.889 [port 2/3]Sending IPC to LP for creating rcv SA an 0
Oct 27 06:34:04.889 [port 2/3] Sending IPC to LP for creating transmit SA an 0
Oct 27 06:34:04.889 [port 2/3] CP SM state change from [secure] to [receive]
Oct 27 06:34:04.889 [port 2/3] CP current state receive
Oct 27 06:34:04.898 [port 2/3] IPC response - RX SA is ready to receive
Oct 27 06:34:04.898 [port 2/3] CP current state receive
Oct 27 06:34:04.898 [port 2/3] CP SM state change from [receive] to [receiving]
Oct 27 06:34:04.898 [port 2/3] CP current state receiving
```

```
Oct 27 06:34:04.898 [port 2/3] CP SM state change from [receiving] to [ready]
Oct 27 06:34:05.042 [port 2/3] CP current state ready
Oct 27 06:34:05.042 Port 2/3 moved to non-blocking
Oct 27 06:34:05.042 [port 2/3] Enabling transmit SA an 0
Oct 27 06:34:05.042 [port 2/3] CP SM state change from [ready] to [transmit]
Oct 27 06:34:05.043 [port 2/3] IPC response- TX SA is transmittng
Oct 27 06:34:05.043 [port 2/3] CP current state transmit
Oct 27 06:34:05.043 [port 2/3] CP SM state change from [transmit] to
[transmitting]
Oct 27 06:34:05.043 [port 2/3] CP current state transmitting
Oct 27 06:34:05.043 [port 2/3] CP SM state change from [transmitting] to [retire]
Oct 27 06:34:06.608 [port 2/3] CP current state retire
Oct 27 06:34:06.881 [port 2/3] Update Next PN 2
Oct 27 06:34:08.608 [port 2/3] CP current state retire
Oct 27 06:34:08.870 [port 2/3] Update Next PN 5
Oct 27 06:34:10.609 [port 2/3] CP current state retire
Oct 27 06:34:10.870 [port 2/3] Update Next PN 6
Oct 27 06:34:12.608 [port 2/3] CP current state retire
Oct 27 06:34:12.882 [port 2/3] Update Next PN 7
Oct 27 06:34:14.609 [port 2/3] CP current state retire
Oct 27 06:34:14.870 [port 2/3] Update Next PN 8
Oct 27 06:34:16.622 [port 2/3] CP current state retire
Oct 27 06:34:16.870 [port 2/3] Update Next PN 9
```

The following example enables MKA protocol data unit receive debugging.

```
device# debug dot1x-mka mkpdu-rx
DOT1X-MKA rx debugging ON
DUT5_8#Oct 27 06:36:18.617 [port 2/3] Received MKPDU - mkpdu len 148
Oct 27 06:36:18.617 [port 2/3] Key length 16 icv_pdu_len 150
Oct 27 06:36:18.617 [port 2/3] ICV Validation successful
Oct 27 06:36:18.617 [port 2/3] { Rx MKPDU } { Basic Param } Version: 1, Priority:
16, Desired: 1, Capable: 2, Length: 60
Oct 27 06:36:18.617 [port 2/3] { Rx MKPDU } { Basic Param } SCI: 0024388f6b920001
Oct 27 06:36:18.617 [port 2/3] { Rx MKPDU } { Basic Param } MI:
e7271c1c5133cef5cd14455d, MN: 73, Agility: 0x0080c201
Oct 27 06:36:18.617 [port 2/3] { Rx MKPDU } { Basic Param } CKN:
0102030405060708090a0b0c0d0e0f100102030405060708090a0b0c0d0e0f10
Oct 27 06:36:18.617 [port 2/3] MKPDU - Processing basic parameter info len 60
Oct 27 06:36:18.617 [port 2/3] Received Peer mn = 73
Oct 27 06:36:18.617 [port 2/3] MKPDU - Processing Live peer parameter info
Oct 27 06:36:18.617 [port 2/3] {Rx MKPDU } { Live Peer Param } MI:
f2570d864df4107aff60f591, MN: 72
Oct 27 06:36:18.617 [port 2/3] Peer has my recent MI and MN info, peer_mn 72
actor_mn 72
Oct 27 06:36:18.617 [port 2/3] MKPDU - Processing SAK use parameter info
Oct 27 06:36:18.617 [port 2/3] { Rx MKPDU } { SAK Use Param } Type: 3, LAN: 0,
OAN: 0, Delay: 0, Len: 40
Oct 27 06:36:18.617 [port 2/3] { Rx MKPDU } { SAK Use Param } LTX: 0, LRX: 0, OTX:
1, ORX: 1, PlainTX: 0, PlainRX: 0
Oct 27 06:36:18.617 [port 2/3] { Rx MKPDU } { SAK Use Param } LKI:
00000000000000000000000000000000, LLPN: 0x00000055
Oct 27 06:36:18.617 [port 2/3] { Rx MKPDU } { SAK Use Param } OKI:
e7271c1c5133cef5cd14455d00000001, OLPN: 0x00000000
Oct 27 06:36:18.617 [port 2/3] SAK use info - no change of KI and AN
```

Debug commands H-P

debug ikev2

Enables the Internet Key Exchange Protocol version 2 (IKEv2) debugging.

Syntax

On LP

debug ikev2 [**back-up** | **error** | **event** | **packet** { **receive** | **send** } [**detail**] | **peer** { *ip-address* | *ipv6-address* } | **trace** | **tunnel** *tunnel-id*]

no debug ikev2 [**back-up** | **error** | **event** | **packet** { **receive** | **send** } [**detail**] | **peer** { *ip-address* | *ipv6-address* } | **trace** | **tunnel** *tunnel-id*]

On MP

debug ikev2 [**back-up** | **error** | **event** | **extended-log** | **trace** | **tunnel** *tunnel-id*]

no debug ikev2 [**back-up** | **error** | **event** | **extended-log** | **trace** | **tunnel** *tunnel-id*]

Parameters

back-up
Enables IKEv2 backup events debugging.

error
Enables IKEv2 error debugging.

event
Enables IKEv2 event debugging.

packet
Enables IKEv2 packet debugging.

receive
Enables IKEv2 packet debugging in receive mode.

send
Enables IKEv2 packet debugging in send mode.

detail
Displays detailed IKEv2 packet debugging information.

peer
Displays IKEv2 debugging information for the specified peer.

ip-address
Specifies the IP adderss of the peer.

ipv6-address
Specifies the IPv6 address of the peer.

trace

Enables IKEv2 trace debugging.

tunnel *tunnel-id*

Enables IKEv2 events debugging related to a specific Internet Protocolsecurity (IPsec) tunnel. The value can be from 1 through 8192. The configured tunnel ID value must be valid, existing, and an IPsec tunnel ID.

extended-log

Enables IKEv2 extended log debugging.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

On the LP, the **debug ikev2 packet** command output resembles the following example.

```
device# debug ikev2 packet
Jun 25 09:22:27.000 IKE: Sending request to 10.100.113.2 [500]
Jun 25 09:22:27.000 IKE: Sending message 10.100.113.2[500]: IKE_SA_INIT, #1(4), ID
00000000
Jun 25 09:22:27.000 lp_ipsec_ike_tx_ipv4_ike_packet send packet to
10.100.113.2:500 from 10.52.23.51:500 length 248
Jun 25 09:22:32.000 IKE: Resending message 52.100.113.2[500], IKE_SA_INIT, #1(4),
ID 00000000, 1(3)
Jun 25 09:22:32.000 lp_ipsec_ike_tx_ipv4_ike_packet send packet to
10.100.113.2:500 from 10.52.23.51:500 length 248
Jun 25 09:22:42.000 IKE: Resending message 10.100.113.2[500], IKE_SA_INIT, #1(4),
ID 00000000, 2(3)
Jun 25 09:22:42.000 lp_ipsec_ike_tx_ipv4_ike_packet send packet 10.100.113.2:500
from 10.52.23.51:500 length 248
```

On the LP, the **debug ikev2 packet detail** command output resembles the following example.

```
device# debug ikev2 packet detail
Jun 25 09:23:51.375 +++ ISAKMP package start (message # 1) +++
Jun 25 09:23:51.375 < IKE Header >
Jun 25 09:23:51.375 IKE_SA Initiator's SPI: 0x9499cb7fcf1160a4
Jun 25 09:23:51.375 IKE_SA Responder's SPI: 0x0000000000000000
Jun 25 09:23:51.375 Next Payload: security association (33)
Jun 25 09:23:51.375 Version: 2.0
Jun 25 09:23:51.375 Exchange Type: IKE_SA_INIT
Jun 25 09:23:51.375 Flags: [ INITIATOR ]
Jun 25 09:23:51.375 Message ID: 0
Jun 25 09:23:51.375 Length: 216 (#x)
Jun 25 09:23:51.375 < security association >
Jun 25 09:23:51.375 Next Payload: key exchange (34)
Jun 25 09:23:51.375 Payload Length: 48 (#x)
Jun 25 09:23:51.375 << proposal >>
Jun 25 09:23:51.375 Proposal #: 1
Jun 25 09:23:51.375 Protocol ID: ISAKMP
Jun 25 09:23:51.375 SPI Size: 0
Jun 25 09:23:51.375 # of Transforms: 4
Jun 25 09:23:51.375 <<< transform >>>
Jun 25 09:23:51.375 Type: encr
Jun 25 09:23:51.375 ID: aes
Jun 25 09:23:51.375 Attr: 800e0100
Jun 25 09:23:51.375 <<< transform >>>
Jun 25 09:23:51.375 Type: integ
Jun 25 09:23:51.375 ID: sha384
Jun 25 09:23:51.375 <<< transform >>>
Jun 25 09:23:51.375 Type: prf
Jun 25 09:23:51.375 ID: sha384
Jun 25 09:23:51.375 <<< transform >>>
Jun 25 09:23:51.375 Type: dh
Jun 25 09:23:51.375 ID: ecp384
Jun 25 09:23:51.375 < key exchange >
Jun 25 09:23:51.375 Next Payload: nonce (40)
Jun 25 09:23:51.375 Payload Length: 104 (#x)
Jun 25 09:23:51.375 dh group: 'ecp384'
Jun 25 09:23:51.375 data:
f9648262 be296717 a90476fb 8433915e 919c15cb 5c774fc1 335bd463 cfbacf67
9762b6be 05c04f41 70f4c45c 5e2b746a c3acded6 2f86b16e c3eb102a 547c1b5c
83a8515b 2ca67cc2 15f1e1c7 b56811e9 0eee4f2f 557f1f11 4b57d646 15c30c55
Jun 25 09:23:51.375 < nonce >
Jun 25 09:23:51.375 Next Payload: none (0)
Jun 25 09:23:51.375 Payload Length: 36 (#x)
Jun 25 09:23:51.375 data:
dbe908e0 42e89912 fcd45200 94fc988e 875ea6a2 b56cda27 ad8c0e5f e18ec8df
Jun 25 09:23:51.375 --- ISAKMP package end (message # 1) ---
```

On the LP, the **debug ikev2 error** command output resembles the following example.

```
device# debug ikev2 error
Jun 25 09:25:11.525 IKE: ike_wr_check_matching_condition not found #1
Jun 25 09:25:11.525 IKE: IKE match condition not found for tunnel id 203
Jun 25 09:25:11.525 IKE: ipike_isakmp_handle_error: error: 1
Jun 25 09:25:11.525 IKE: IKE error: 12
Jun 25 09:25:11.525 ike sa not present on primary
```

On the LP, the **debug ikev2 tunnel** command output resembles the following example.

```
device# debug ikev2 tunnel 8
Oct 13 07:39:09.525 IKEV2 EVENT: received IPC Msg IKE_MP_LP_IKE_CLEAR and length
46
Oct 13 07:39:09.525 IKE: ike_wr_sa_create_delete(): delete initiator cookie:
0xa66857626c451525, responder cookie: 0xffb12e0b5dca5624
Oct 13 07:39:09.525 IKEV2 EVENT: IKE IPC send message IKE_LP_MP_IKE_SA_DEL_E and
len 47 to MP
Oct 13 07:39:09.525 IKE: ipike_isakmp_hash_obj_cookie() :: hash key: id=0,
type=37, init=1 -> val = e327622b
Oct 13 07:39:09.525 IKE: ipike_isakmp_hash_add() :: hash: id=0, type=37, init=1,
phase1=0
Oct 13 07:39:09.525 IKE: ipike_isakmp_hash_add() :: elements: cookie=2 id=2
Oct 13 07:39:09.525 IKE: Message encrypted 88 bytes
Oct 13 07:39:09.525 IKEV2 IKEv2:ipcom_hash_for_each took 1 ms
Oct 13 07:39:09.525 IKEV2 IKEv2:ipcom_hash_for_each longest CPU holdtime was 0 ms
Oct 13 07:39:09.525 IKE: ipike_isakmp_free_exchange() :: exchange 736180768 type -
INFORMATIONAL
Oct 13 07:39:09.550 IKE: ipike_isakmp_hash_obj_cookie() :: hash key: id=0, type=0,
init=0 -> val = 61b21908
Oct 13 07:39:09.550 IKE: ipike_isakmp_hash_remove() :: hash elements: 1 1
Oct 13 07:39:09.550 IKEV2 IKEv2:ipcom_hash_for_each took 0 ms
Oct 13 07:39:09.550 IKEV2 IKEv2:ipcom_hash_for_each longest CPU holdtime was 0 ms
Oct 13 07:39:09.550 IKE: IKE SA freed (0x2be46820)
Oct 13 07:39:09.550 IKE: initiator cookie: 0xa66857626c451525
Oct 13 07:39:09.550 IKE: responder cookie: 0xffb12e0b5dca5624
Oct 13 07:39:09.550 IKE: Deleting IPsec SA
Oct 13 07:39:09.550 IKE: Outbound IPSec SA ESP, spi 000024fc
Oct 13 07:39:09.550 IKE: proto: any
Oct 13 07:39:09.550 IKE: src: 0.0.0.0-255.255.255.255:0-65535
Oct 13 07:39:09.550 IKE: dst: 0.0.0.0-255.255.255.255:0-65535
Oct 13 07:39:09.550 IKE: ipike_ipsec_sadb_delete_sa - Hash child elements: 3 2
Oct 13 07:39:09.550 IKE: ipike_pfkeyv2_remove_sa: calling fpga remove-entry
Oct 13 07:39:09.550 IKE: ike_wr_update_ipsec_sa(): ipsec delete with in spi-id
0x0000ba66 for tunnel 8
Oct 13 07:39:09.550 IKEV2 EVENT: IKE IPC send message IKE_LP_MP_IPSEC_SA_DEL_E and
len 74 to MP
Oct 13 07:39:09.550 IKE: Deleting IPsec SA
Oct 13 07:39:09.550 IKE: Inbound IPSec SA ESP, spi 0000632c
Oct 13 07:39:09.550 IKE: proto: any
Oct 13 07:39:09.550 IKE: src: 0.0.0.0-255.255.255.255:0-65535
Oct 13 07:39:09.550 IKE: dst: 0.0.0.0-255.255.255.255:0-65535
Oct 13 07:39:09.550 IKE: ipike_ipsec_sadb_delete_sa - Hash child elements: 0 0
Oct 13 07:39:09.550 IKE: ipike_pfkeyv2_remove_sa: calling fpga remove-entry
Oct 13 07:39:09.550 IKE: ike_wr_update_ipsec_sa(): ipsec delete with in spi-id
0x0000632c for tunnel 8
Oct 13 07:39:09.550 IKEV2 EVENT: IKE IPC send message IKE_LP_MP_IPSEC_SA_DEL_E and
len 74 to MP
Oct 13 07:39:09.550 IKEV2 IKEv2:ipcom_hash_for_each took 2 ms
Oct 13 07:39:09.550 IKEV2 IKEv2:ipcom_hash_for_each longest CPU holdtime was 0 ms
Oct 13 07:39:09.550 IKEV2 lp_ipsec_ike_check_and_process_ipv4_ike_packet
Oct 13 07:39:09.550 IKEV2 ike packet received with pkt type 1 length 96
Oct 13 07:39:09.550 IKE: ipike_isakmp_process_message_request():
Oct 13 07:39:09.550 IKE: ipike_isakmp_hash_get_from_cookie() :: hash key: id=0,
type=37, init=00000001
Oct 13 07:39:09.550 IKE: ipike_isakmp_hash_get_from_cookie() :: hash key: id=0,
type=37, init=00000000
Oct 13 07:39:09.550 IKE: ipike_isakmp_new_exchange_responder() :: message is a
response, probably a retransmit for local add 10.10.6.2 remote addr 10.10.6.1->
ignore
Oct 13 07:39:09.550 IKE: ipike_isakmp_handle_error: error: 76
```

On the MP, the **debug ikev2 error** command output resembles the following example.

```
device# debug ikev2 error
IKEV2: Error debugging is on
Jan 13 18:01:36.102 received IPC Msg from LP 0, MsgType IKE_LP_MP_
IKE_SA_DEL_E and Length 43
Jan 13 18:01:36.102 IKE sa delete for tunnel id 23 initiator spi id dbd8945d0c3f
4ac2 and responder id 0000000000000000 , state 4, tlv count 0
Jan 13 18:01:36.102 received IPC Msg from LP 0, MsgType IKE_LP_MP_IKE_SA_DEL_E a
nd Length 43
Jan 13 18:01:36.102 IKE sa delete for tunnel id 104 initiator spi id 9bdfebdea9a
2cc95 and responder id 0000000000000000 , state 4, tlv count 0
Jan 13 18:01:36.102 received IPC Msg from LP 0, MsgType IKE_LP_MP_IKE_SA_DEL_E a
nd Length 43
Jan 13 18:01:36.102 IKE sa delete for tunnel id 105 initiator spi id 422d7510313
5491a and responder id 0000000000000000 , state 4, tlv count 0
```

On the MP, the **debug ikev2 tunnel** command output resembles the following example.

```
device# debug ikev2 tunnel 8
ike_ipc_send_to_lp: to LP mask -1 with MsgType IKE_MP_LP_IKE_CLEAR and length 46
delete ike sa with initiaor spi 66356fa6a8b659db and responder spi
a8d5d1d7b93b14c8 from the list
delete ipsec sa spi 0000d6cf
ike_ipsec_update_tunnel_nht for tunnel 8
ike_ipsec_update_tunnel_nht delete nht for tunnel 8
delete ipsec sa spi 0000769f
ike_ipsec_update_tunnel_nht for tunnel 8
delete tunnel 8 nht 2
```

debug ip aaa

Enables authentication, authorization, and accounting (AAA) debugging and displays debug information about AAA or RADIUS or TACACS authentication.

Syntax

debug ip aaa

no debug ip aaa

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays debug information about AAA or RADIUS authentication.

```
device# 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
```

The following example displays information about AAA or TACACS+ authentication.

```
device# 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.
```

debug ip arp

Generates information about Address Resolution (ARP)

Syntax

debug ip arp [**event** | **ipc** | **itc** | **packet**]

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

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays information about ARP transactions.

```
device# 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
```

The following example displays information about ARP events, and is useful in determining whether the router is sending and receiving ARP requests. The example shows send and receive activity for ARP packets.

```
device# 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
```

The following example generates information about ARP interprocess communication (IPC) activity.

```
device# 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
```

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

```
device# 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
```

The following example displays information about ARP packet activity. The output 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.

```
device# 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 ip bgp

Debugs and monitors the BGP environment.

Syntax

debug ip bgp [**all-vrfs** | *vrf-name*] [**bfd route-selection** | **dampening** | **events** | **graceful-restart** | **keepalives** | **neighbor** *address* | **ip-prefix** | **ipv6-prefix** | **ip-prefix-list** | **ipv6-prefix-list** | **route-map** | **route-updates** | **updates** [**rx** | **tx**] }

no debug ip bgp [**all-vrfs** | *vrf-name*] [**bfd route-selection** | **dampening** | **events** | **graceful-restart** | **keepalives** | **neighbor** *address* | **ip-prefix** | **ipv6-prefix** | **ip-prefix-list** | **ipv6-prefix-list** | **route-map** | **route-updates** | **updates** [**rx** | **tx**] }

Parameters

all-vrfs
Displays information for all virtual routing and forwarding events.

vrf-name
Displays information for a specific VRF event.

bfd
Displays information for BFD event.

route-selection
Displays BGP route selection debug information.

dampening
Displays BGP dampening activity.

events
Displays BGP event information.

graceful-restart
Displays information about graceful-restart events.

keepalives
Displays keepalive activity.

neighbor
Displays BGP neighbor activity.

address
Specifies the IPv4 or IPv6 address of the neighbor.

ip-prefix
Displays IPv4 prefix debug information.

ipv6-prefix
Displays IPv6 prefix debug information.

ip-prefix-list
Displays IPv4 prefix list debug information.

ipv6-prefix-list
Displays IPv6 prefix list debug information.

route-map
Displays BGP route map debug information

route-updates
Displays BGP route update debug information

updates
Displays BGP receive and transmit update messages about debug processing.

rx
Displays BGP receive update messages about debug processing.

tx
Displays BGP transmit update messages about debug processing.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example enables debugging for BFD.

```
device# 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)
```

The following example enables common BGP debugs to be displayed for all VRFs or for a specific VRF.

```
device# 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
```

The following example enables debugging of BGP route selection for all VRFs or for a specified VRF.

```
device# 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
```

The following example displays information about dampening process configurations, route penalties, durations, restraint, and release.

```
device# 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
```

The following example generates information about BGP events, such as connection attempts and keepalive timer activity.

```
device# 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
Sep 9 18:36:42 BGP: 192.168.1.1 rcv capability 65, len 4
```

The following example enables to receive information about BGP graceful restarts.

```
device# 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]
```

The following example specifies the IPv4 or IPv6 neighbor filter for BGP debugging for all VRFs or for a specified VRF.

```
device# 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
```

The following example specifies the IPv4 or IPv6 neighbor filter for BGP debugging for all VRFs or for a specified VRF.

```
device# debug ip bgp neighbor 2001:DB8::20
BGP: neighbor 2001:DB8::20 debugging is on
device# 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
```

The following is a sample output from the **debug ip bgp ip-prefix** command.

```
device# 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 following example reflects an IPv6 prefix filter, or a similar IPv6 prefix list along with AFI/SAFI as IPv6 and unicast.

```
device# 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
```

The following example displays the routes that have been shared with a neighbor. The route information includes the four-byte AS4_PATH attribute and the AS_PATH attribute.

```
device# 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
```

The following example enables debugging of BGP update message processing for all VRFs or for a specific VRF.

```
device# 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
```

debug ip icmp

Displays information about IPv4 Internet Control Message Protocol (ICMP) transactions.

Syntax

debug ip icmp [**events** | **external_loop** | **internal_loop** | **packets** | **port_loop**]

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

Parameters

events
Displays information about ICMP events.

external_loop
Displays ICMP external loop activity.

nternal_loop
Displays ICMP internal loop activity.

packets
Displays information about ICMP packets.

port_loop
Displays port loop activity.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

This command is useful in determining if a router is sending or receiving ICMP messages, and for troubleshooting end-to-end connections.

Examples

The following example generates information about ICMP events, such as sent and received echo (ping) requests, destination-unreachable messages, and redirect messages.

```
device# 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
```

The following example generates information about ICMP packets.

```
device# 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 ip igmp

Generates information about IGMP activity, including IGMP membership queries, membership responses, and the conversion of IGMP V2 to IGMP V3 through DNS lookup.

Syntax

debug ip igmp [**protocol query**]

no debug ip igmp [**protocol query**]

Parameters

protocol query

Displays information about the Internet Group Management Protocol (IGMP) query suppression state on the Cluster Client Edge Ports (CCEPs).

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example generates information about IGMP activity.

```
device# 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
```

The following example displays information about the Internet Group Management Protocol (IGMP) query suppression state on the Cluster Client Edge Ports (CCEPs).

```
device# 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 igmp mct-mdup

Generates debugging information about IGMP activity in relation to Multi-Chassis Trunking (MCT) Database Update (MDUP) settings.

Syntax

debug ip igmp mct-mdup [*stack*]

no debug ip igmp mct-mdup [*stack*]

Parameters

stack

Displays function call stack information along with other debugging information.

Modes

Privileged EXEC mode

Usage Guidelines

Use this command when troubleshooting IGMP MCT MDUP settings. MDUP can be a bottleneck between MCT peers.

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example generates information about IGMP MCT MDUP settings activity.

```
device# debug ip igmp mct-mdup

Dec 19 22:35:53.121 IGMP.VRF0.IPC: [ Port 2/5,v30. Grp 226.1.5.5 ] mcgrp_mark_one_group_for_mdup.
Dec 19 22:35:53.130 IGMP.VRF0.IPC: [ Port 2/5,v31. Grp 226.1.5.5 ] mcgrp_mark_one_group_for_mdup.
Dec 19 22:35:53.137 IGMP.VRF0.IPC: [ Port 2/5,v32. Grp 226.1.5.5 ] mcgrp_mark_one_group_for_mdup.
Dec 19 22:35:53.145 IGMP.VRF0.IPC: [ Port 2/5,v33. Grp 226.1.5.5 ] mcgrp_mark_one_group_for_mdup.
Dec 19 22:35:53.150 IGMP.VRF0.IPC: [ Port 4/1,v191. Grp 225.1.1.1 ] Not CCEP Port. MDUP flag 0
Dec 19 22:35:53.155 IGMP.VRF0.IPC: [ Port 4/1,v191. Grp 225.1.1.4 ] Not CCEP Port. MDUP flag 0
Dec 19 22:35:53.160 IGMP.VRF0.IPC: [ Port 4/1,v191. Grp 225.1.1.4 ] Not CCEP Port. MDUP flag 0
Dec 19 22:35:53.170 IGMP.VRF0.IPC: [ Port 3/1,v192. Grp 226.1.1.1 ] Not CCEP Port. MDUP flag 0
```

History

Release version	Command history
06.1.00	This command was introduced.

debug ip ipc

Generates information about interprocess communication (IPC) activity in IP modules.

Syntax

debug ip ipc

no debug ip ipc

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

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

Examples

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

```
device# 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
```

debug ip msdp

Generates information about Multicast Source Discovery Protocol (MSDP) alarms, events, and messages.

Syntax

debug ip msdp [**alarms** | **events** | **messages**]

no debug ip msdp [**alarms** | **events** | **messages**]

Parameters

alarms

Displays information about MSDP alarms.

events

Displays information about MSDP events.

messages

Displays information about MSDP messages.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example 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.

```
device# debug ip msdp alarms
MSDP: alarms debugging is on
MSDP: S=xxxxxxx P=0 Initiate Transport Connection to MSDP peer
```

The following example tracks originating SA advertisements, major peer events, and peer keepalive timer events.

```
device# 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
```

The following example generates information (including message contents) about MSDP messages received, transmitted, and forwarded, and flag errors in MSDP messages.

```
device# 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
```

debug ip multicast

Displays multicast information.

Syntax

debug ip multicast { **add-del-oif** | **error** | **events** | **ipc** | **protocol**}

no debug ip multicast { **add-del-oif** | **error** | **events** | **ipc** | **protocol**}

Parameters

add-del-oif
Displays information about addition or deletion of the outgoing interfaces (OIFs) to or from the mcache entry.

error
Displays information about any kind of unexpected error.

events
Displays information about system events such as VRF changes, interface changes, and so on.

ipc
Displays debug information about the IPC messages communicated between the MP and the LP.

protocol
Displays information related to the processing and handling of multicast query or reports.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays information about addition or deletion of the outgoing interfaces (OIFs) to or from the mcache entry.

```
device# debug ip multicast add-del-oif
L2MCASTV4.MLD.ADD_DEL: l2mcast_l2mdb_process_port_down
```

The following example displays error information.

```
device# debug ip multicast error
L2MCASTV4.IGMP.ERR: Rx packet has invalid checksum. Dropping packet
```

The following example displays information about system events.

```
device# 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
```

The following example displays debug information about the IPC messages communicated between the MP and the LP.

```
device# debug ip multicast ipc
L2MCASTV4.MLD.IPC: mgmt_send_enable_disable
```

The following example displays information related to the processing and handling of multicast query or reports.

```
device# 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
```

debug ip netconf

Enables NETCONF debugging.

Syntax

debug ip netconf [**content** | **engine** | **framer** | **operation** | **parser** | **rpc** | **transport**]

no debug ip netconf [**content** | **engine** | **framer** | **operation** | **parser** | **rpc** | **transport**]

Parameters

content
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.

engine
Enables NETCONF engine debugging.

framer
Displays the NETCONF XML framer level debugging information.

operation
Displays information about NETCONF operations, such as retrieve, configure, copy, delete, and so on.

parser
Enables NETCONF XML parser level debugging.

rpc
Displays debug messages related to the NETCONF Remote Procedure Call (RPC)layer.

transport
Displays debug messages related to the NETCONF transport layer.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example 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.

```
device# debug ip netconf content
NETCONF: content debugging is on
```

The following example enables NETCONF engine debugging.

```
device# 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
```

The following example displays the NETCONF XML framer level debugging information.

```
device# 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
```

The following example displays information about NETCONF operations, such as retrieve, configure, copy, delete, and so on.

```
device# 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
```

The following example enables NETCONF XML parser level debugging.

```
device# 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
```

The following example displays debug messages related to the NETCONF Remote Procedure Call (RPC) layer.

```
device# 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
```

The following example displays debug messages related to the NETCONF transport layer.

```
device# debug ip netconf transport
NETCONF: transport debugging is on
device# config t
device(config)# netconf server
Oct 26 13:11:16 NETCONF: Enabling the Netconf Server port
Oct 26 13:11:16 NETCONF: Received TCP listen callback
device(config)# no netconf server
Oct 26 13:11:23 NETCONF: Disabling the Netconf server port
device# no debug ip netconf transport
NETCONF: transport debugging is off
```

debug ip ntp

Displays information about NTP.

Syntax

debug ip ntp [**algorithms** | **association** | **broadcast** | **clockadjust** | **errors** | **packet** | **server**]

no debug ip ntp [**algorithms** | **association** | **broadcast** | **clockadjust** | **errors** | **packet** | **server**]

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays information about the NTP system algorithms.

```
device# 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
```

The following example displays information about the NTP server and peer association.

```
device# 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
```

The following example displays information about the NTP broadcast server and client.

```
device# 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
```

The following example displays information about the NTP clock-adjust process.

```
device# 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
```

The following example displays information about the NTP input and output packets.

```
device# 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
```

The following example displays information about the NTP server.

```
device# 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
```

debug ip ospf

Enables OSPF debugging.

Syntax

debug ip ospf [*ip-address* | **adj** | **all-vrfs** | **bfd** | **error** | **events** | **flood** | **graceful-restart** | **log-debug-message** | **log-empty-lsa** | **lsa-filtering** | **lsa-generation** | **max-metric** | **packet** | **packet-hello** | **packet-dd** | **packet-lsrequest** | **packet-lsupdate** | **packet-lsacknowledge** | **retransmission** | **route** *ip-address* | **sham-link** | **shortcuts** | **spf** | **vrf**]

no debug ip ospf [*ip-address* | **adj** | **all-vrfs** | **bfd** | **error** | **events** | **flood** | **graceful-restart** | **log-debug-message** | **log-empty-lsa** | **lsa-filtering** | **lsa-generation** | **max-metric** | **packet** | **packet-hello** | **packet-dd** | **packet-lsrequest** | **packet-lsupdate** | **packet-lsacknowledge** | **retransmission** | **route** *ip-address* | **sham-link** | **shortcuts** | **spf** | **vrf**]

Parameters

ip-address
Displays OSPF information for a specific IP address.

adj
Displays information about IP OSPF adjacencies.

all-vrfs
Enables OSPF debugging for all VPN routing and forwarding activity.

bfd
Displays information about OSPF BFD events.

error
Displays IP OSPF errors.

NOTE
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.

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. This option can be enabled on the standby MP as well.

max-metric
Displays information about a max-metric configuration.

packet
Displays IP OSPF packet information.

packet-hello
Displays IP OSPF hello packet information.

packet-dd
Displays OSPF Database Descriptor (DD) packet information.

packet-lsrequest
Displays IP OSPF LS request packet information.

packet-lsupdate
Displays IP OSPF LS update packet information.

packet-lsacknowledge
Displays IP OSPF LS acknowledge packet information.

retransmission
Displays IP OSPF retransmission information.

route *ip-address*
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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

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 the **error** option only when network traffic levels are low.

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.

Examples

The following example 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.

```
device# 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
```

The following example 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.

```
device# 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
```

The following example displays information about internal OSPF events related to configuration or interaction with the standby management processor and interface state transitions.

```
device# 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
```

The following example displays information about LSA flooding activity.

```
device# 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
```

The following command enables to receive information about OSPF graceful restart events, including restart phases, graceful LSA transmit and receive activity, and syslog messages.

```
device# 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 : ÿÿÿÿÿÿÿÿÿ
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
```

The following command logs instances when large (greater than MTU) LSA update messages are sent or received.

```
device# 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
```

The following command logs instances when empty or truncated LSA update messages are sent or received.

```
device# 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
```

The following example displays type-3 LSAs as and when a router generates and filters these LSAs.

```
device# 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
```

The following example generates information about LSAs. The 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.

```
device# 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
```

The following example displays all received and transmitted OSPF packets with decoded options and flags.

```
device# debug ip ospf packet
Aug 29 11:58:59.817 OSPF(red): recv from:10.50.50.10 Intf:ve 50 LS-Upd L:348 A:0
Rid:10.21.21.21 Cnt:10
Type:1 Age:1 LSID:10.21.21.21 Adv:10.21.21.21 Options:--------E- Links:0x01,
Flags:-Nt-VEB
Link ID: 0x6a323200, link Data: 0xffffff00
, type 3, TOS 0, metric 1
Type:1 Age:6 LSID:10.31.31.31 Adv:10.31.31.31 Options:--------E- Links:0x1,
Flags:-----E
Link ID: 0x6a32320a, link Data: 0x6a32328c
Aug 29 11:58:13.813 OSPF: send to:10.20.20.10 Intf:ve 20 DD L:32 A:0
Rid:10.11.11.11 Flags:IMoMa, 00002b43
Aug 29 11:58:13.814 OSPF: recv from:10.20.20.10 Intf:ve 20 DD L:32 A:0
Rid:10.21.21.21 Flags:IMoMa, 036290e6
Aug 29 11:58:13.814 OSPF: send to:10.20.20.10 Intf:ve 20 DD L:92 A:0
Rid:10.11.11.11 Flags:---Sl, 036290e6
LSType:1, LSID:10.11.11.11 Adv-Router:10.11.11.11 Options:--------ELSType:
3, LSID:10.40.40.0 Adv-Router:10.11.11.11 Options:--------ELSType:
3, LSID:10.30.30.0 Adv-Router:10.11.11.11 Options:--------EAug
29 11:58:13.815 OSPF: recv from:10.20.20.10 Intf:ve 20 DD L:292 A:0
Rid:10.21.21.21 Flags:---Ma, 036290e7
LSType:1, LSID:10.11.11.11 Adv-Router:10.11.11.11 Options:--------ELSType:
1, LSID:10.21.21.21 Adv-Router:10.21.21.21 Options:--------ESep
26 06:46:00.973 OSPF: recv from:10.30.30.10 Intf:ve 30 LS-Ack L:144 A:0
Rid:10.130.130.3
LSType:1, LSID:10.11.11.11 Adv-Router:10.11.11.11 Options:----------
LSType:3, LSID:10.20.20.0 Adv-Router:10.11.11.11 Options:----------
LSType:3, LSID:10.50.50.0 Adv-Router:10.11.11.11 Options:----------
LSType:3, LSID:10.10.10.0 Adv-Router:10.11.11.11 Options:----------
LSType:3, LSID:10.60.60.0 Adv-Router:10.11.11.11 Options:----------
LSType:3, LSID:10.40.40.0 Adv-Router:10.11.11.11 Options:----------
Sep 26 06:45:52.549 OSPF: send to:10.0.0.5 Intf:ve 20 Hello L:48 A:0
Rid:10.11.11.11 Options:--------E- DR:10.20.20.10 BDR:10.20.20.140
Sep 26 06:45:53.673 OSPF: recv from:10.10.10.10 Intf:ve 10 Hello L:48 A:0
Rid:10.21.21.21 Options:--------E- DR:10.10.10.10 BDR:10.10.10.140
eb
```

The following example displays sent and received OSPF hello packets.

```
device# debug ip ospf packet-hello
Sep 26 06:45:52.549 OSPF: send to:10.0.0.5 Intf:ve 20 Hello L:48 A:0
Rid:10.11.11.11 Options:--------E- DR:10.20.20.10 BDR:10.20.20.140
Sep 26 06:45:53.673 OSPF: recv from:10.10.10.10 Intf:ve 10 Hello L:48 A:0
Rid:10.21.21.21 Options:--------E- DR:10.10.10.10 BDR:10.10.10.140
```

The following example displays sent and received OSPF Database Descriptor (DD) packets.

```
device# debug ip ospf packet-dd
Aug 29 11:58:13.813 OSPF: send to:10.20.20.10 Intf:ve 20 DD L:32 A:0
Rid:10.11.11.11 Flags:IMoMa, 00002b43
Aug 29 11:58:13.814 OSPF: recv from:10.20.20.10 Intf:ve 20 DD L:32 A:0
Rid:10.21.21.21 Flags:IMoMa, 036290e6
Aug 29 11:58:13.814 OSPF: send to:10.20.20.10 Intf:ve 20 DD L:92 A:0
Rid:10.11.11.11 Flags:---Sl, 036290e6
LSType:1, LSID:10.11.11.11 Adv-Router:10.11.11.11 Options:--------ELSType:
3, LSID:10.40.40.0 Adv-Router:10.11.11.11 Options:--------ELSType:
3, LSID:10.30.30.0 Adv-Router:10.11.11.11 Options:--------E
Aug 29 11:58:13.815 OSPF: recv from:10.10.20.10 Intf:ve 20 DD L:292 A:0
Rid:10.21.21.21 Flags:---Ma, 036290e7
LSType:1, LSID:10.11.11.11 Adv-Router:10.11.11.11 Options:--------ELSType:
1, LSID:10.21.21.21 Adv-Router:10.21.21.21 Options:--------Edebug
```

The following example displays sent and received OSPF LS request packets.

```
device# debug ip ospf packet-lsrequest
Apr 18 11:30:06.705 OSPF: send to:10.2.45.1 Intf:eth 3/5 LS-Req L:60 A:0
Rid:10.2.2.2
Type:1, LSID:10.1.1.1 Adv:10.1.1.1
Type:2, LSID:10.2.45.2 Adv:10.2.2.2
Type:2, LSID:10.2.46.2 Adv:10.2.2.2
Apr 18 11:30:06.705 OSPF: recv from:10.2.45.1 Intf:eth 3/5 LS-Req L:36 A:0
Rid:10.1.1.1
Type:1, LSID:10.2.2.2 Adv:10.2.2.2
```

The following example displays sent and received OSPF LS update packets.

```
device# debug ip ospf packet-lsupdate
Aug 29 11:58:59.817 OSPF(red): recv from:10.50.50.10 Intf:ve 50 LS-Upd L:348 A:0
Rid:10.21.21.21 Cnt:10
Type:1 Age:1 LSID:10.21.21.21 Adv:10.21.21.21 Options:--------E- Links:0x01,
Flags:-Nt-VEB
Link ID: 0x6a323200, link Data: 0xffffff00
, type 3, TOS 0, metric 1
Type:1 Age:6 LSID:10.31.31.31 Adv:10.31.31.31 Options:--------E- Links:0x1,
Flags:-----E
Link ID: 0x6a32320a, link Data: 0x6a32328c
```

The following example displays sent and received OSPF LS acknowledge packets.

```
device# debug ip ospf packet-lsacknowledge
Sep 26 06:46:00.973 OSPF: recv from:10.30.30.10 Intf:ve 30 LS-Ack L:144 A:0
Rid:10.130.130.3
LSType:1, LSID:10.11.11.11 Adv-Router:10.11.11.11 Options:----------
LSType:3, LSID:10.20.20.0 Adv-Router:10.11.11.11 Options:----------
LSType:3, LSID:10.50.50.0 Adv-Router:10.11.11.11 Options:----------
LSType:3, LSID:10.10.10.0 Adv-Router:10.11.11.11 Options:----------
LSType:3, LSID:10.60.60.0 Adv-Router:10.11.11.11 Options:----------
LSType:3, LSID:10.40.40.0 Adv-Router:10.11.11.11 Options:----------
```

The following example generates internal information about OSPF retransmission of LSAs.

```
device# 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
```

The following example generates network-specific information during Dijkstra computation, routing table calculation, and LSA origination. This command is useful for tracking a specific OSPF prefix.

```
device# 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
```

The following example generates information about OSPF shortcuts for IP over MPLS.

```
device# 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
```

The following example generates information about OSPF SPF activity including SPF runs and calculations.

```
device# 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
```

debug ip pim mct

Generates IP Protocol-Independent Multicast (PIM) debugging information about Multi-Chassis Trunking (MCT) event and callback handling.

Syntax

debug ip pim mct

no debug ip pim mct

Modes

Privileged EXEC mode

Usage Guidelines

Use this command with PIM over MCT (PIMoMCT) when troubleshooting MCT Database Update (MDUP) event and callback messages. MDUP can be a bottleneck between MCT peers.

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays debugging information about MCT Database Update (MDUP) event and callback messages.

```
device# debug ip pim mct

Dec 19 22:43:21.686 PIMv4.ALL-VRF,Dbg: PIM_MCT.   mcast_mct_send_message_to_peer: msg_type 2, msg_len
1474, client 100, vlan 10
Dec 19 22:43:21.688 PIMv4.ALL-VRF,Dbg: PIM_MCT.   mcast_mct_send_message_to_peer: msg_type 2, msg_len
1474, client 100, vlan 10
Dec 19 22:43:21.690 PIMv4.ALL-VRF,Dbg: PIM_MCT.   mcast_mct_send_message_to_peer: msg_type 2, msg_len
1474, client 100, vlan 10
Dec 19 22:43:21.692 PIMv4.ALL-VRF,Dbg: PIM_MCT.   mcast_mct_send_message_to_peer: msg_type 2, msg_len
1374, client 100, vlan 10
Dec 19 22:43:21.694 PIMv4.ALL-VRF,Dbg: PIM_MCT.   mcast_mct_send_message_on_mdup: app_id 64, seg_size
1454, seg_base 0 returned 0
Dec 19 22:43:21.694 PIMv4.ALL-VRF,Dbg: PIM_MCT.   mcast_mct_send_message_on_mdup: app_id 64, seg_size
1454, seg_base 1450 returned 0
Dec 19 22:43:21.694 PIMv4.ALL-VRF,Dbg: PIM_MCT.   mcast_mct_send_message_on_mdup: app_id 64, seg_size
1454, seg_base 2900 returned 0
Dec 19 22:43:21.694 PIMv4.ALL-VRF,Dbg: PIM_MCT.   mcast_mct_send_message_on_mdup: app_id 64, seg_size
1454, seg_base 4350 returned 0
Dec 19 22:43:21.694 PIMv4.ALL-VRF,Dbg: PIM_MCT.   mcast_mct_send_message_on_mdup: app_id 64, seg_size
72, seg_base 5800 returned 0
Dec 19 22:43:23.221 PIMv4.ALL-VRF,Dbg: PIM_MCT.  mcast_mct_process_message_from_peer: Received mdup
with dest-ip 224.0.0.13 from peer
Dec 19 22:43:23.221 PIMv4.ALL-VRF,Dbg: PIM_MCT.  mcast_mct_process_ipv4_packet_from_peer: Received IPv4
packet on 2/3 VLAN 20 client-rbridge-id 200
Dec 19 22:43:23.223 PIMv4.ALL-VRF,Dbg: PIM_MCT.  mcast_mct_process_message_from_peer: Received mdup
with dest-ip 224.0.0.13 from peer
Dec 19 22:43:23.223 PIMv4.ALL-VRF,Dbg: PIM_MCT.  mcast_mct_process_ipv4_packet_from_peer: Received IPv4
packet on 2/3 VLAN 20 client-rbridge-id 200
```

History

Release version	Command history
06.1.00	This command was introduced.

debug ip pim-dvmrp

Enables monitoring the DVMRP environment.

Syntax

debug ip pim-dvmrp [**clear** | **event** | **ipc** | **nbr-change** | **optimizations** | **rate-update** | **route-change** | **rp-change** | **sync-lib** | **timer-type**]

no debug ip pim-dvmrp [**clear** | **event** | **ipc** | **nbr-change** | **optimizations** | **rate-update** | **route-change** | **rp-change** | **sync-lib** | **timer-type**]

Parameters

clear
Clears PIM-DVMRP debug settings.

event
Displays information about infrastructure events and callback handling.

ipc
Displays information about IPC messages between the management processor and a line processor.

nbr-change
Displays information about neighbor port changes.

optimizations
Displays PIM-DVMRP optimizations.

rate-update
Displays IPC between LP and MP to communicate rates of PIM streams.

route-change
Displays information about route change events.

rp-change
Displays information about RP events.

sync-lib
Displays information about the impact that hitless upgrade and MP switchover have on multicast flows.

timer-type
Displays information regarding the events associated with the multicast protocol and hardware timers in the system.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following is the sample output from the **debug ip pim-dvmrp entry** command.

```
device# debug ip pim-dvmrp entry
PIM: KAT timer debugging is on
PIM: Entry Hw programming debugging is on
PIM: Entry creation/deletion debugging is on
```

The following example displays IPC messages between the management processor and a line processor and 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.

```
device# 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
```

The following example indicates that neighbor 10.30.30.1 has changed from port e2/1 to port e3/1.

```
device# debug ip pim-dvmrp 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
```

The following command displays information about the hardware forwarding resources used by all multicast flows in the system.

```
device# debug ip pim-dvmrp optimization
PIM: optimization debugging is on
```

The following command displays the IPC messages exchanged between the LP and the MP to communicate rates of PIM streams.

```
device# debug ip pim-dvmrp rate-update
PIM-INFO(HW rate update).VRF 0: (10.1.1.1, 225.0.0.1) Total pkts: 100
```

The following command displays information about route change events.

```
device# debug ip pim-dvmrp route-change
PIM: route-change debugging is on
device# 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-dvmrp entry

Enables debugging of PIM entries.

Syntax

debug ip pim-dvmrp entry [**add-del** | **hw** | **kat**]

no debug ip pim entry [**add-del** | **hw** | **kat**]

Parameters

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).

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following is the sample output from the **debug ip pim-dvmrp entry** command.

```
device# debug ip pim-dvmrp entry
        PIM: KAT timer debugging is on
        PIM: Entry Hw programming debugging is on
        PIM: Entry creation/deletion debugging is on
```

debug ip pim-dvmrp filter

Enables filtering of DVMRP debug options.

Syntax

debug ip pim-dvmrp filter [**group-prefix** | **level** *level-num* | **source-prefix** | **stack** | **vrf-index**]

no debug ip pim-dvmrp filter [**group-prefix** | **level** *level-num* | **source-prefix** | **stack** | **vrf-index**]

Parameters

group-prefix
Filters and displays debugs relevant to groups that are within the group prefix range.

level *level-num*
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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following is the sample output from the **debug ip pim-dvmrp filter group-prefix** command.

```
device# debug ip pim-dvmrp 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-dvmrp filter level** command.

```
device# debug ip pim-dvmrp filter level 2
PIM: Debug Level debugging is on, value = 2
```

debug ip pim-dvmrp oif

Enables PIM DM and PIM SM related debugging.

Syntax

debug ip pim-dvmrp oif [**add-del** | **fsm** | **timer**]

no debug ip pim-dvmrp oif [**add-del** | **fsm** | **timer**]

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following is the sample output from the **debug ip pim-dvmrp oif** command.

```
device# debug ip pim-dvmrp oif
  PIM: OIF FSM debugging is on
  PIM: OIF timer debugging is on
  PIM: OIF External add-del debugging is on
```

debug ip rip

Enables RIP debugging.

Syntax

debug ip rip [**all-vrfs** | **vrf** *vrf-name*] [**database** | **events** | **packet** | **trigger**]

no debug ip rip [**all-vrfs** | **vrf** *vrf-name*] [**database** | **events** | **packet** | **trigger**]

Parameters

all-vrfs
Displays information about all VRFs of the RIP.

vrf *vrf-name*
Displays information about a specific VRF 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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays information about RIP database events.

```
device# 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

device# 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
```

The following example displays information about RIP events.

```
device# 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
device(config-if-e1000-1/5)# ip rip v2-only
Feb 7 08:57:16.910 RIP(default-vrf): start running on interface 1/5
device(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
```

The following example displays information about RIP packets sent and received.

```
device# 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

device# 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)
```

The following example displays information about triggered updates which is being sent, and periodic updates.

```
device# debug ip rip trigger
Feb 7 09:10:35.964 RIP(default-vrf): triggered update sent on port 1793
device(config-if-e1000-1/5)# ip rip v2-only
device(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 rtm

Displays information about the routing table manager (RTM), including changes in the routing table.

Syntax

debug ip rtm [*ip-address* | **all** | **errors** | **inter-vrf-routing** | **nexthop**]

no debug ip rtm [*ip-address* | **all** | **errors** | **inter-vrf-routing** | **nexthop**]

Parameters

ip-address
: Displays RTM information for a specified IPv4 address.

all
: Displays all RTM information.

errors
: Displays RTM errors.

inter-vrf-routing
: Enables inter-VRF routing debugging for IPv4 routes.

nexthop
: Logs various next hop-related events to the console.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

This example indicates that OSPF route 10.11.11.0/24 has been deleted from the route table.

```
device# debug ip rtm
IP: rtm debugging is on
device# 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
```

The following example enables inter-VRF routing debugging for IPv4 routes.

```
device# 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
```

The following example enables debugging for the IPv4 nexthop table and displays information about various next-hop related events.

```
device# 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
```

debug ip ssh

Generates information about SSH connections.

Syntax

debug ip ssh [**msgs**]

no debug ip ssh [**msgs**]

Parameters

msgs

Indicates which keyexchange algorithm is negotiated.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example shows information about SSH connections for a user logging in.

```
device# 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.
```

The following example shows information about SSH connections and also indicates which key exchange algorithm that is negotiated.

```
device# debug ip ssh msgs
May 30 03:30:05.983 SSH[0]: ssh_is_connection_allowed..NEW CONNECTION
May 30 03:30:05.983 SSH[0]:ssh_socket_control: new connection
May 30 03:30:05.983 SSH[0]:ShtcpOpenPassive: WaitingForConnection (reusing)
May 30 03:30:05.983 SSH[0]:Awaiting child task init complete
May 30 03:30:05.985 SSH[0]:ssh_socket_control:Incoming connection ready
May 30 03:30:05.985 SSH:tcp incoming tcb 0x104689fc, handle 0xa000013e
May 30 03:30:05.985 SSH[0]: Data is ready to receive
May 30 03:30:05.985 SSH[0]:ShtcpConnectionStatus: connection established
May 30 03:30:05.985 SSH:tcp receive data tcb handle 0x00000002
May 30 03:30:06.480 SSH[0]:ShtcpReceiveStatus: string length 41
May 30 03:30:06.480 SSH: eShSessionInit state
May 30 03:30:06.480 SSH[0]:ShtcpSend: eSendComplete: sent 24/24
May 30 03:30:06.482 SSH[0]: Data is ready to receive
May 30 03:30:06.482 SSH[0]:ProcessPendingSend: send completion status=Complete,
result=0
May 30 03:30:06.482 SSH: eShSessionExpectVersion state
May 30 03:30:06.482 SSH[4] Length of the buffer received is 41
May 30 03:30:06.482 SSH[0]: Sending Message Id 20
May 30 03:30:06.482 SSH[0]: Sending Message length 320
May 30 03:30:06.482 SSH[0]:ShtcpSend: eSendComplete: sent 320/320
May 30 03:30:06.482 SSH:tcp receive data tcb handle 0x00000002
May 30 03:30:06.980 SSH[0]:ShtcpReceiveStatus: string length 1928
May 30 03:30:06.980 SSH[0]:ProcessPendingSend: send completion status=Complete,
result=0
May 30 03:30:06.980 SSH: eShSessionNegotiation state
May 30 03:30:06.980 SSH: eShSessionExpectClientNegotiation state
May 30 03:30:06.980 SSH: eShSessionKeyExchangeDhInit state
May 30 03:30:06.981 SSH: eShSessionExpectNewKeys state
May 30 03:30:06.981 SSH: eShSessionTerminate state
May 30 03:30:06.981 SSH[0] Length of the buffer received is 1928
May 30 03:30:06.981 Not protected by hmac
May 30 03:30:06.981 ShProcessMessage: 20
May 30 03:30:06.981 SSH[0]: Received Message 20
May 30 03:30:06.981 kShMsgKexInit message received
May 30 03:30:06.981 Match: diffie-hellman-group1-sha1
May 30 03:30:06.981 Match: ssh-rsa
May 30 03:30:06.981 Match: aes128-ctr
May 30 03:30:06.981 Match: aes128-ctr
May 30 03:30:06.981 Match: hmac-sha1
May 30 03:30:06.981 Match: hmac-sha1
May 30 03:30:06.981 Match: none
May 30 03:30:06.981 Match: none
May 30 03:30:06.981 Not protected by hmac
May 30 03:30:06.981 ShProcessMessage: 30
May 30 03:30:06.981 SSH[0]: Received Message 30
May 30 03:30:06.981 kShMsgKexDhInit message received
May 30 03:30:07.197 SSH[0]: Sending Message Id 31
May 30 03:30:07.311 SSH[0]: Sending Message length 704
May 30 03:30:07.312 SSH[0]:ShtcpSend: eSendComplete: sent 704/704
May 30 03:30:07.313 SSH[0]: Data is ready to receive
May 30 03:30:07.313 SSH[0]: Connection Task(21)
May 30 03:30:07.313 SSH[0]:ShtcpReceiveStatus: string length 16
May 30 03:30:07.313 SSH[0]:ProcessPendingSend: send completion status=Complete,
result=0
May 30 03:30:07.313 SSH: eShSessionExpectNewKeys state
May 30 03:30:07.313 SSH: eShSessionTerminate state
May 30 03:30:07.313 SSH[0] Length of the buffer received is 16
May 30 03:30:07.313 Not protected by hmac
May 30 03:30:07.313 ShProcessMessage: 21
May 30 03:30:07.313 SSH[0]: Received Message 21
May 30 03:30:07.313 kShMsgNewKeys message received
May 30 03:30:07.314 SSH[0]: Sending Message Id 21
May 30 03:30:07.314 SSH[0]: Sending Message length 16
May 30 03:30:07.314 SSH[0]:ShtcpSend: eSendComplete: sent 16/16
May 30 03:30:07.314 SSH:tcp receive data tcb handle 0x00000002
May 30 03:30:07.314 SSH[0]: Data is ready to receive
May 30 03:30:07.314 SSH[0]:ShtcpReceiveStatus: string length 52
```

```
May 30 03:30:07.314 SSH[0]:ProcessPendingSend: send completion status=Complete,
result=0
May 30 03:30:07.314 SSH: eShSessionEncrypted state
May 30 03:30:07.314 SSH[0] Length of the buffer received is 52
May 30 03:30:07.314 HMac matched incoming packet
May 30 03:30:07.314 ShProcessMessage: 5
May 30 03:30:07.314 SSH[0]: Received Message 5
May 30 03:30:07.314 kShMsgServiceRequest message received
May 30 03:30:07.314 Received Service request: ssh-userauth
May 30 03:30:07.314 SSH[0]: Sending Message Id 6
May 30 03:30:07.314 SSH[0]: Sending Message length 52
May 30 03:30:07.314 SSH[0]:ShtcpSend: eSendComplete: sent 52/52
May 30 03:30:07.314 SSH:tcp receive data tcb handle 0x00000002
May 30 03:30:07.315 SSH[0]: Data is ready to receive
May 30 03:30:07.315 SSH[0]:ShtcpReceiveStatus: string length 68
May 30 03:30:07.315 SSH[0]:ProcessPendingSend: send completion status=Complete,
result=0
May 30 03:30:07.315 SSH: eShSessionEncrypted state
May 30 03:30:07.315 SSH[0] Length of the buffer received is 68
May 30 03:30:07.315 HMac matched incoming packet
May 30 03:30:07.315 ShProcessMessage: 50
May 30 03:30:07.315 SSH[0]: Received Message 50
May 30 03:30:07.315 kShMsgUserAuthRequest message received
May 30 03:30:07.315 Received: UserAuth user=brocade service=ssh-connection
method=none
May 30 03:30:07.315 SSH[0]: Sending Message Id 51
May 30 03:30:07.315 SSH[0]: Sending Message length 84
May 30 03:30:07.315 SSH[0]:ShtcpSend: eSendComplete: sent 84/84
May 30 03:30:07.315 SSH:tcp receive data tcb handle 0x00000002
May 30 03:30:07.316 SSH[0]: Data is ready to receive
May 30 03:30:07.316 SSH[0]:ShtcpReceiveStatus: string length 100
May 30 03:30:07.316 SSH[0]:ProcessPendingSend: send completion status=Complete,
result=0
May 30 03:30:07.316 SSH: eShSessionEncrypted state
May 30 03:30:07.316 SSH[0] Length of the buffer received is 100
May 30 03:30:07.316 HMac matched incoming packet
May 30 03:30:07.316 ShProcessMessage: 50
May 30 03:30:07.316 SSH[0]: Received Message 50
May 30 03:30:07.316 kShMsgUserAuthRequest message received
May 30 03:30:07.316 Received: UserAuth user=brocade service=ssh-connection
method=keyboard-interactive
May 30 03:30:07.316 SSH[0]: Sending Message Id 60
May 30 03:30:07.316 SSH[0]: Sending Message length 68
May 30 03:30:07.316 SSH[0]:ShtcpSend: eSendComplete: sent 68/68
May 30 03:30:07.316 SSH:tcp receive data tcb handle 0x00000002
May 30 03:30:07.480 SSH[0]:ProcessPendingSend: send completion status=Complete,
result=0
May 30 03:30:07.480 SSH: eShSessionEncrypted state
May 30 03:30:07.480 SSH[0] Length of the buffer received is 0
May 30 03:30:07.980 SSH: eShSessionEncrypted state
May 30 03:30:07.980 SSH[0] Length of the buffer received is 0
May 30 03:30:08.480 SSH: eShSessionEncrypted state
May 30 03:30:08.480 SSH[0] Length of the buffer received is 0
May 30 03:30:08.980 SSH: eShSessionEncrypted state
May 30 03:30:08.980 SSH[0] Length of the buffer received is 0
May 30 03:30:12.480 SSH: eShSessionEncrypted state
May 30 03:30:15.480 SSH[0]: Connection Task(41)
May 30 03:30:15.480 SSH: eShSessionEncrypted state
May 30 03:30:15.480 SSH[0] Length of the buffer received is 0
May 30 03:30:17.980 SSH: eShSessionEncrypted state
May 30 03:30:19.480 SSH[0] Length of the buffer received is 0
May 30 03:30:19.980 SSH: eShSessionEncrypted state
May 30 03:30:19.980 SSH[0] Length of the buffer received is 0
May 30 03:30:20.480 SSH[0]: Connection Task(51)
May 30 03:30:20.480 SSH: eShSessionEncrypted state
May 30 03:30:20.480 SSH[0] Length of the buffer received is 0
May 30 03:30:20.980 SSH: eShSessionEncrypted state
May 30 03:30:20.980 SSH[0] Length of the buffer received is 0
May 30 03:30:21.480 SSH: eShSessionEncrypted state
May 30 03:30:21.480 SSH[0] Length of the buffer received is 0
May 30 03:30:21.980 SSH: eShSessionEncrypted state
May 30 03:30:21.980 SSH[0] Length of the buffer received is 0
```

```
May 30 03:30:22.480 SSH: eShSessionEncrypted state
May 30 03:30:22.480 SSH[0] Length of the buffer received is 0
May 30 03:30:22.980 SSH: eShSessionEncrypted state
May 30 03:30:22.980 SSH[0] Length of the buffer received is 0
May 30 03:30:23.480 SSH: eShSessionEncrypted state
May 30 03:30:23.480 SSH[0] Length of the buffer received is 0
May 30 03:30:26.324 SSH[0]: Data is ready to receive
May 30 03:30:26.324 SSH[0]:ShtcpReceiveStatus: string length 84
May 30 03:30:26.324 SSH: eShSessionEncrypted state
May 30 03:30:26.324 SSH[0] Length of the buffer received is 84
May 30 03:30:26.324 HMac matched incoming packet
May 30 03:30:26.324 ShProcessMessage: 61
May 30 03:30:26.324 SSH[0]: Received Message 61
May 30 03:30:26.324 kShMsgUserAuthInfoResponse message received
May 30 03:30:26.324 SSH[0]: 1 (1=Group1, 2=Group14, 3=GroupEx(Group14))
May 30 03:30:26.342 SSH[0]: Sending Message Id 52
May 30 03:30:26.342 SSH[0]: Sending Message length 36
May 30 03:30:26.342 SSH[0]:ShtcpSend: eSendComplete: sent 36/36
May 30 03:30:26.342 SSH:tcp receive data tcb handle 0x00000002
May 30 03:30:26.343 SSH[0]: Data is ready to receive
May 30 03:30:26.343 SSH[0]:ShtcpReceiveStatus: string length 68
May 30 03:30:26.343 SSH[0]:ProcessPendingSend: send completion status=Complete,
result=0
```

History

Release version	Command history
6.0.00a	This command was modified to include the keyword **msgs**.

debug ip ssl

Enables debugging of a Secure Socket Layer (SSL) connection.

Syntax

debug ip ssl

no debug ip ssl

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following command enables debugging of an SSL connection.

```
device# 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
```

debug ip telnet

Generates information about incoming Telnet connections.

Syntax

debug ip telnet

no debug ip telnet

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example generates information about incoming Telnet connections.

```
device# debug ip telnet
TELNET: Data is ready to receive
TELNET: Data is ready to receive
```

debug ip tunnel

Generates information about Generic Routing Encapsulation (GRE) and IPv6 tunnels.

Syntax

debug ip tunnel [**errors** | **events** | **ipc** | **keepalives** | **packets** | **statistics** | **tunnel-type** [**gre** | **ipv6**]] [**range** *lower-tunnel-id upper-tunnel-id*]

no debug ip tunnel [**errors** | **events** | **ipc** | **keepalives** | **packets** | **statistics** | **tunnel-type** [**gre** | **ipv6**]] [**range** *lower-tunnel-id upper-tunnel-id*]

Parameters

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 *lower-tunnel-id upper-tunnel-id*
Displays information for a range of tunnels (specify lower and higher tunnel ID values).

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example 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.

```
device# 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
```

The following example 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.

```
device# 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
```

The following example displays IPC messages related to debugging information for the specified range of tunnels.

```
device# 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
```

The following example displays messages about the receiving and transmitting of keepalive packets, keepalive timer actions, and bringing the tunnel up or down based on keepalive actions.

```
device# 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
```

The following example 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.

```
device# 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
```

The following example 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.

```
device# 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 ip vrf

Generates information about synchronization of VRF routing information to line cards.

Syntax

debug ip vrf

no debug ip vrf

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example shows a download request from the line card to start the tree download for VRF 1.

```
device# debug ip vrf
RTM (vrf): Processing tree download for vrf 1
```

debug ip vrrp

Enables debugging of VRRP.

Syntax

debug ip vrrp [**all** | **error** | **ethernet** *slot/port* | **events** | **packets** | **show** | **state** | **ve** *num* | **verbose** | **vrid** *num*]

no debug ip vrrp [**all** | **error** | **ethernet** *slot/port* | **events** | **packets** | **show** | **state** | **ve** *num* | **verbose** | **vrid** *num*]

Parameters

all
Displays information about all IPv4 VRRP instances (default).

error
Displays error conditions where a packet is not being processed.

ethernet *slot/port*
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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

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.

Examples

If debugging is enabled and MD5 authentication is configured on an interface, the **debug ip vrrp** command output resembles the following example.

```
device# 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:[00000000000000000000400010]
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.

```
device# 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:[00000000000000000000400010]
IpAddr: 2001:DB8::40:10
```

If debugging is enabled and MD5 authentication fails, the **debug ip vrrp** command output resembles the following example.

```
device# 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:[00000000000000000000400010]
IpAddr: 2001:DB8::40:10
```

debug ipsec

Enables debugging of Internet Protocol security (IPsec) operation.

Syntax

On LP

debug ipsec [**error** | **log** | **packet** | **trace**]

no debug ipsec [**error** | **log** | **packet** | **trace**]

On MP

debug ipsec [**esp** | **sa** | **policy** | **in** | **out**]

no debug ipsec [**esp** | **sa** | **policy** | **in** | **out**]

Parameters

error

Displays IPsec error conditions.

log

Displays IPsec logging information.

packet

Displays IPsec packet processing information.

trace

Displays trace information for IPsec.

esp

Enables 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).

sa

Displays debugging information related to the security associations used by IPsec for OSPFv3 packets.

policy

Displays debugging information for IPsec policy.

in

Displays debugging information related to inbound OSPFv3 packets with IPsec.

out

Displays debugging information related to outbound OSPFv3 packets with IPsec.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays IPsec packet processing information.

```
device# debug ipsec packet
IPSec: packet debugging is on
IPsec packet received: Packet size: 93, IPsec 4 Bytes: 0x80550000
Foundry header:
00: 05 f2 44 03 98 54 cf fa 3c 18 0c b8 0c 00 00 00
10: 80 55 00 00
Packet (starting with Ethernet Header):
00: 00 24 38 92 79 00 74 8e f8 31 f7 01 08 00 45 c0
10: 00 3b 0c e0 00 00 40 06 dc bf c8 2e 00 00 c8 2e
20: 00 01 00 b3 1f f1 8d cc d8 6d 8d 47 9d b0 50 18
30: fd e8 6b 89 00 00 ff ff ff ff ff ff ff ff ff ff
40: ff ff ff ff ff ff 00 13 04
IPsec packet received: Packet size: 80, IPsec 4 Bytes: 0x80160000
Foundry header:
00: 05 f2 44 03 98 54 cf fa 3c 18 0d 78 0c 00 00 00
10: 80 16 00 00
Packet (starting with Ethernet Header):
00: 00 24 38 92 79 00 74 8e f8 31 f7 01 08 00 45 c0
10: 00 2c 0c e1 00 00 40 06 dc cb c8 2f 00 00 c8 2f
20: 00 01 1f bc 00 b3 14 5f 8f 02 00 00 00 00 60 02
30: fd e8 46 0d 00 00 02 04 05 b4 00 00
IPsec packet received: Packet size: 80, IPsec 4 Bytes: 0x80160000
Foundry header:
00: 05 f2 44 03 98 54 cf fa 3c 18 0d 78 0c 00 00 00
10: 80 16 00 00
Packet (starting with Ethernet Header):
00: 00 24 38 92 79 00 74 8e f8 31 f7 01 08 00 45 c0
10: 00 2c 0c e2 00 00 40 06 dc ca c8 2f 00 00 c8 2f
20: 00 01 1f bc 00 b3 14 5f 8f 02 00 00 00 00 60 02
30: fd e8 46 0d 00 00 02 04 05 b4 00 00
```

The following example displays trace information for IPsec.

```
device# debug ipsec trace
IPSec: trace debugging is on
Jan 13 10:25:18.599 IPsec: Packet received for IPsec Tunnel 17
Jan 13 10:25:18.650 IPsec: Packet received for IPsec Tunnel 23
Jan 13 10:25:18.750 IPsec: Packet received for IPsec Tunnel 23
Jan 13 10:25:18.900 IPsec: Packet received for IPsec Tunnel 46
Jan 13 10:25:19.199 IPsec: Packet received for IPsec Tunnel 67
Jan 13 10:25:19.300 IPsec: Packet received for IPsec Tunnel 40
Jan 13 10:25:19.300 IPsec: Packet received for IPsec Tunnel 57
Jan 13 10:25:19.500 IPsec: Packet received for IPsec Tunnel 333
Jan 13 10:25:19.599 IPsec: Packet received for IPsec Tunnel 38
Jan 13 10:25:20.150 IPsec: Packet received for IPsec Tunnel 68
Jan 13 10:25:20.775 IPsec: Packet received for IPsec Tunnel 46
Jan 13 10:25:21.099 IPsec: Packet received for IPsec Tunnel 18
Jan 13 10:25:22.300 IPsec: Packet received for IPsec Tunnel 37
Jan 13 10:25:23.349 IPsec: Packet received for IPsec Tunnel 63
Jan 13 10:25:23.525 IPsec: Packet received for IPsec Tunnel 21
Jan 13 10:25:23.699 IPsec: Packet received for IPsec Tunnel 36
Jan 13 10:25:24.625 IPsec: Packet received for IPsec Tunnel 59
Jan 13 10:25:24.824 IPsec: Packet received for IPsec Tunnel 20
Jan 13 10:25:24.824 IPsec: Packet received for IPsec Tunnel 39
```

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.

```
device# debug ipsec esp
IPSec: esp debugging is on
device# 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::)
```

The following example displays debugging information related to the security associations used by IPsec for OSPFv3 packets.

```
device# debug ipsec sa
IPSec: sa debugging is on
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
```

The following example displays debugging information for IPsec policy.

```
device# debug ipsec policy
IPSec: policy debugging is on
devuce# conf t
device(config)# interface ethernet 1/8
device(config-if-e1000-1/8)# ipv6 ospf authentication ipsec spi 121212 esp sha1
no-encrypt 12345678901234567890123456789012345678901234567890
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
device(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
```

The following example displays debugging information related to inbound OSPFv3 packets with IPsec.

```
device# 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 ::
```

The following example displays debugging information related to outbound OSPFv3 packets with IPsec.

```
device# 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 access-list

Enables debugging of IPv6 access lists.

Syntax

debug ipv6 access-list [**error** | **ipv6** | **ipv6-cam** | **rate-limit** | **receive** [**time** | **cam** | **generic** | **packet**] | **stats**]

no debug ipv6 access-list [**error** | **ipv6** | **ipv6-cam** | **rate-limit** | **receive** [**time** | **cam** | **generic** | **packet**] | **stats**]

Parameters

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.

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.

stats
Displays statistical information about the IPv6 access list.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays error information about the IPv6 access list.

```
device# 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
```

The following example displays information about the IPv6 access list activity. In this example it is assumed that an IPv6 traffic filter is on incoming traffic to a virtual interface abc10.

```
device# debug ipv6 access-list ipv6
device# 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>>
```

The following example displays CAM information about the IPv6 access list.

```
device# 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
```

On MP, the **debug ipv6 access-list rate-limit** command output resembles the following example for a configuration flow when the rate limiting feature is enabled.

```
device# debug ipv6 access-list rate-limit
IPv6: ACL-based Rate-Limiting debugging is on
device(config-if-e1000-4/8)# rate-limit input access-group name ipv6 rl-perm
200000 10000
Dec 17 12:17:22.820 RL-ACL: port 4/8: RL config: attrib 5001, vlan_pri_flag 0x00,
rl_vrf_idx 0
Average rate is adjusted to 195456 bits per second.
Dec 17 12:17:22.821 RL: port 4/8: allocate temp_rl_config
```

On LP, the **debug ipv6 access-list rate-limit** command output resembles the following example for a configuration flow when the rate limiting feature is enabled.

```
device# 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
```

The following is the sample output from the **debug ipv6 access-list receive generic** command.

```
device# debug ipv6 access-list receive generic
device(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
```

In the following example, with **debug ipv6 access-list stats** command enabled, the **show ipv6 access-list accounting brief** command generates a brief accounting summary.

```
device# debug ipv6 access-list stats
device# 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
device# 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)
```

debug ipv6 dhcp

Enables IPv6 DHCP debugging.

Syntax

debug ipv6 dhcp [**pd all** | **pd flash** | **pd pd-option** | **pd static** | **sync**]

no debug ipv6 dhcp [**pd all** | **pd flash** | **pd pd-option** | **pd static** | **sync**]

Parameters

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).

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example enables all IPv6 PD debugging.

```
device# 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
```

The following example enables PD debugging for read and write to a flash file.

```
device# 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
```

The following example enables debugging for processing of PD options in DHCPv6 messages.

```
device# 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.
```

The following example enables debugging of DHCPv6 interface with IPv6 static route module.

```
device# 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
```

The following example enables debugging of DHCPv6 synchronization between active and standby MPs.

```
device# 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
```

debug ipv6 icmp

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.

Syntax

debug ipv6 icmp

no debug ipv6 icmp

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays information about ICMP activity.

```
device# 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
```

debug ipv6 mld mct-mdup

Generates debugging information about Multicast Listener Discovery (MLD) activity in relation to Multi-Chassis Trunking (MCT) Database Update (MDUP) settings.

Syntax

debug ipv6 mld mct-mdup [*stack*]

no debug ipv6 mld mct-mdup [*stack*]

Parameters

stack

Displays function call stack information along with other debugging information.

Modes

Privileged EXEC mode

Usage Guidelines

Use this command when troubleshooting MLD MCT MDUP settings. MDUP can be a bottleneck between MCT peers.

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example generates information about MLD MCT MDUP settings activity.

```
device# debug ipv6 mld mct-mdup

Aug 23 10:05:47.672 MLD.VRF0.IPC: [ Port 1/7,v10. Grp ff4e::1 ] Wait not set, set refresh flag. MDUP
flag 0
Aug 23 10:08:31.510 MLD.VRF0.IPC: [ Port 1/7,v10. Grp ff4e::1 ] mcgrp_mark_one_group_for_mdup.
Aug 23 10:08:31.512 MLD.VRF0.IPC: [ Port 1/7,v10. Grp ff4e::1 ] MDUP Wait set. MDUP flag 1
```

History

Release version	Command history
06.1.00	This command was introduced.

debug ipv6 mld protocol query

Generates MLD information.

Syntax

debug ipv6 mld protocol query

no debug ipv6 mld protocol query

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays information about MLD query suppression state on the CCEPs.

```
device# 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
```

debug ipv6 multicast

Displays multicast information.

Syntax

debug ipv6 multicast { **add-del-oif** | **error** | **events** | **ipc** | **protocol**}

no debug ipv6 multicast { **add-del-oif** | **error** | **events** | **ipc** | **protocol**}

Parameters

add-del-oif
Displays information about addition or deletion of the outgoing interfaces (OIFs) to or from the mcache entry.

error
Displays information about any kind of unexpected error.

events
Displays information about system events such as VRF changes, interface changes, and so on.

ipc
Displays debug information about the IPC messages communicated between the MP and the LP.

protocol
Displays information related to the processing and handling of multicast query or reports.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays information about addition or deletion of the outgoing interfaces (OIFs) to or from the mcache entry.

```
device# debug ipv6 multicast add-del-oif
L2MCASTV6.MLD.ADD_DEL: l2mcast_l2mdb_process_port_down
```

The following example displays debug information about the IPC messages communicated between the MP and the LP.

```
device# debug ipv6 multicast ipc
L2MCASTV6.MLD.IPC: mgmt_send_enable_disable
```

debug ipv6 nd

Enables debugging for all neighbors.

Syntax

debug ipv6 nd

no debug ipv6 nd

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

To enable debugging for a specific neighbor, enter the **debug ipv6 nd** command followed by the **debug ipv6 address** command.

Examples

The following example enables debugging of all neighbors.

```
device# 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)
```

debug ipv6 ospf

Enables IPv6 OSPF debugging.

Syntax

debug ipv6 ospf [**bfd** | **gr-helper** | **ipsec** | **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**]

no debug ipv6 ospf [**bfd** | **gr-helper** | **ipsec** | **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**]

Parameters

bfd
Displays information about OSPFv3 BFD events.

gr-helper
Displays information about graceful restart (GR) helper operation.

ipsec
Displays information about IPsec events.

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays information about graceful restart (GR) helper operation.

```
device# 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
```

The following example generates comprehensive information about OSPF ISM status changes. This example 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.

```
device# 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)
```

The following example displays IPv6 OSPF interface state machine (ISM) activity, such as an interface coming up or going down.

```
device# debug ipv6 ospf ism-events
OSPFv3 ISM[137]: Interfaces
OSPFv3 ISM[137]: BackupSeen goes up
```

The following example displays IPv6 OSPF ISM status information. This example indicates that ISM 137 has gone down and is waiting for a switch from the designated router (DR) to the backup designated router (BDR).

```
device# 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)
```

The following example displays information about OSPF LSAs.

```
device# 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
```

The following example displays IPv6 OSPF LSA flooding activity.

```
device# 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
```

The following example shows additions or deletions of LSAs from the link state database.

```
device# 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
```

The following example generates information about new LSAs that are installed in the link state database.

```
device# 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
```

The following example 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.

```
device# 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.
```

The following example displays information about IPsec events. This sample output from MP shows success in the attempts to provide various IPsec services to OSPFv3.

```
device# debug ipv6 ospf ipsec
device(config-if-e1000-1/8)# ipv6 ospf authentication ipsec spi 121212 esp sha1
no-encrypt 1234567890123456789012345678901234567890
device(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
```

debug ipv6 pim

Enables monitoring the PIM environment.

Syntax

debug ipv6 pim [**clear** | **event** | **ipc** | **nbr-change** | **optimizations** | **rate-update** | **route-change** | **rp-change** | **sync-lib** | **timer-type**]

no debug ipv6 pim [**clear** | **event** | **ipc** | **nbr-change** | **optimizations** | **rate-update** | **route-change** | **rp-change** | **sync-lib** | **timer-type**]

Parameters

clear
Clears PIM-DVMRP debug settings.

event
Displays information about infrastructure events and callback handling.

ipc
Displays information about IPC messages between the management processor and a line processor.

nbr-change
Displays information about neighbor port changes.

optimizations
Displays PIM-DVMRP optimizations.

rate-update
Displays IPC between LP and MP to communicate rates of PIM streams.

route-change
Displays information about route change events.

rp-change
Displays information about RP events.

sync-lib
Displays information about the impact that hitless upgrade and MP switchover have on multicast flows.

timer-type
Displays information regarding the events associated with the multicast protocol and hardware timers in the system.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example enables IPC debugging.

```
device# debug ipv6 pim ipc
PIM: ipc debugging is on
```

The following example enables nbr-change.

```
device# debug ipv6 pim-dvmrp nbr-change
PIM: nbr-change debugging is on
```

debug ipv6 pim filter

Enables filtering of PIM debug options.

Syntax

debug ipv6 pim filter [**group-prefix** | **level** *level-num* | **source-prefix** | **stack** | **vrf-index**]

no debug ipv6 pim filter [**group-prefix** | **level** *level-num* | **source-prefix** | **stack** | **vrf-index**]

Parameters

group-prefix
Filters and displays debugs relevant to groups that are within the group prefix range.

level *level-num*
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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following is the sample output from the **debug ipv6 pim filter level** command.

```
device# debug ipv6 pim filter level 2
PIM: Debug Level debugging is on, value = 2
```

debug ipv6 pim entry

Enables debugging of PIM entries.

Syntax

debug ipv6 pim entry [**add-del** | **hw** | **kat**]

no debug ipv6 pim entry [**add-del** | **hw** | **kat**]

Parameters

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).

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following is the sample output from the **debug ipv6 pim entry** command.

```
device# debug ipv6 pim entry
        PIM: KAT timer debugging is on
        PIM: Entry Hw programming debugging is on
        PIM: Entry creation/deletion debugging is on
```

debug ipv6 pim oif

Enables PIM DM and PIM SM related debugging.

Syntax

debug ipv6 pim oif [**add-del** | **fsm** | **timer**]

no debug ipv6 pim oif [**add-del** | **fsm** | **timer**]

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following is the sample output from the **debug ipv6 pim oif** command.

```
device# debug ipv6 pim oif
  PIM: OIF FSM debugging is on
  PIM: OIF timer debugging is on
  PIM: OIF External add-del debugging is on
```

debug ipv6 ra

Generates debugging information related to ND6 Router Advertisement (RA) messages.

Syntax

debug ipv6 ra

no debug ipv6 ra

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays debugging information related to ND6 Router Advertisement (RA) messages.

```
device# 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
```

debug ipv6 rip

Generates IPv6 RIP debugging messages.

Syntax

debug ipv6 rip [**all-vrfs** | **vrf** *vrf-name*] [**events** | **receive** | **transmit**]

no debug ipv6 rip [**all-vrfs** | **vrf** *vrf-name*] [**events** | **receive** | **transmit**]

Parameters

all-vrfs
Displays information about all VRFs of the RIPng.

vrf *vrf-name*
Displays information about a specific VRF of the RIPng.

events
Displays RIPng events.

receive
Displays received update packets in RIPng.

transmit
Displays transmitted update packets in RIPng.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays information about all VRFs of the RIPng for transmitted packets.

```
device# 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
```

The following example displays information about RIPng events.

```
device# 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
device(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
```

The following example displays information about RIPng received update packets.

```
device# 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
```

The following example displays information about RIPng transmitted update packets.

```
device# 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
```

The following example displays information about a particular VRF of the RIPng events.

```
device# 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
```

debug ipv6 rtm

Displays information about the IPv6 routing table manager (RTM), including changes in the routing table.

Syntax

debugv6 ip rtm [**inter-vrf-routing** | **nexthop**]

no debug ipv6 rtm [**inter-vrf-routing** | **nexthop**]

Parameters

inter-vrf-routing

Enables inter-VRF routing debugging for IPv6 routes.

nexthop

Logs various next hop-related events to the console.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example enables inter-VRF routing debugging for IPv6 routes.

```
device# 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
```

The following example enables debugging for the IPv6 nexthop table and displays information about various next-hop related events

```
device# 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 ipv6 vrrp

Enables debugging of VRRP.

Syntax

debug ipv6 vrrp [**all** | **error** | **ethernet** *slot/port* | **events** | **packets** | **show** | **state** | **ve** *num* | **verbose** | **vrid** *num*]

no debug ipv6 vrrp [**all** | **error** | **ethernet** *slot/port* | **events** | **packets** | **show** | **state** | **ve** *num* | **verbose** | **vrid** *num*]

Parameters

all
Displays information about all IPv6 VRRP instances (default).

error
Displays error conditions where a packet is not being processed.

ethernet *slot/port*
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
Shows 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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

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.

Examples

If debugging is enabled and MD5 authentication is configured on an interface, the **debug ipv6 vrrp** command output resembles the following example.

```
device# 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::20
```

debug isis

Enables debugging in an IS-IS environment.

Syntax

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** | **spf-stct** | **te** | **trace**]

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** | **spf-stct** | **te** | **trace**]

Parameters

adj
Displays information about IS-IS adjacencies.

bfd
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.

spf-stct
Displays information related to incremental shortcut LSP SPF optimizations.

te
Displays information about IS-IS traffic engineering.

trace
Displays trace information for the IS-IS code path.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example generates information about IS-IS adjacencies.

```
device# 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]
```

The following example displays information about level 1 complete sequence number PDUs (CSNPs) sent and received on the device.

```
device# 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
```

The following example generates information about level 1 hello PDUs sent and received.

```
device# 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
```

The following example generates information about level 1 link state PDUs (LSPs) sent and received.

```
device# 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
```

The following example generates information about level 1 partial sequence number PDUs (PSNPs) sent and received.

```
device# debug isis l1-psnp
ISIS: Received L1 PSNP on 2/24, length 256
ISIS: Received L1 PSNP on 2/24, length 35
```

The following example generates information about level 2 CSN PDUs sent and received.

```
device# 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
```

The following example generates information about level 2 hello PDUs sent and received.

```
device# 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
```

The following example generates information about level 2 link state PDUs sent and received.

```
device# 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
```

The following example generates information about level 2 PSN PDUs (PSNPs) sent and received.

```
device# debug isis l2-psnp
ISIS: Received L2 PSNP on 2/24, length 256
ISIS: Received L2 PSNP on 2/24, length 35
```

The following example generates information about IS-IS memory allocations and releases.

```
device# 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
```

The following example displays information related to LSP, neighbor syncing, and NSR state-related information.

```
device# 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
```

The following example displays information about point-to-point hello PDUs sent and received.

```
device# debug isis pp-hello
ISIS: Sending PTP IIH on 9/1, length 1492
ISIS: Received PTP IIH on 9/1, length 256
```

The following example generates information about OSI PPP packets sent and received.

```
device# 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
```

The following example generates information about IS-IS PSPF activity.

```
device# 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
```

The following example generates detailed information about IS-IS PSPF activity.

```
device# 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
```

The following example displays route redistribution information into or out of IS-IS routes. This output indicates several redistribution activities, including importing and unimporting connected routes, and adding and deleting external routes.

```
device# 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
```

The following example reports changes to the IS-IS route table.

```
device# 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
```

The following example generates information about SPF calculations made for IS-IS.

```
device# 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!
```

The following example displays debug information related to incremental shortcut LSP SPF optimizations.

```
device# 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
```

The following example generates information about internal IS-IS functions.

```
device# debug isis trace
ISIS:proc_SNPE
ISIS: build_csnp
ISIS: build_csnp
ISIS: sig_description
```

debug license

Displays the package information on which the license has been loaded.

Syntax

debug license

no debug license

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

The package information 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.

Examples

The following is a sample output of the **show license** command when debugging is not enabled.

```
device# show license
Index      Package Name         Lid          License Type    Status     License Period
1          NI-CES-2048-L3U      ucGNFOGHGO   normal          active     unlimited
device#show license 1
License information for license <1>:
        +package name:        NI-CES-2048-L3U
        +lid:                 ucGNFOGHGO
        +license type:        normal
        +status:              active
        +license period:      unlimited
```

The following is a sample output after **debug license** command is enabled.

```
device# debug license
License all debugging ON
device# show license 1
License information for license <1>:
        +package name:        NI-CES-2048-L3U
        +lid:                 ucGNFOGHGO
        +license type:        normal
        +status:              active
        +license period:      unlimited
License information:
        +pkg info:     0X00000003
```

debug link-keepalive

Enables UDLD debugging.

Syntax

debug link-keepalive { **packet** [**ethernet** *slot/port*] [**tx** | **rx**] | **show** }

no debug link-keepalive { **packet** [**ethernet** *slot/port*] [**tx** | **rx**] | **show** }

Parameters

packet
Enables debugging of PDUs.

ethernet *slot/port*
Specifies a specific port on which debugging should be enabled.

tx
Enables debugging for transmit PDUs.

rx
Enables debugging for all ports for receive PDUs.

show
Displays debugging parameters for link-keepalive protocol

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following command enables debugging for all ports for transmit PDUs and receive PDUs. This command is available only on LP.

```
device# debug link-keepalive packet
Link-Keepalive: All Ports RX TX debugging is on
device# Sending Local Port In New Version: 1/2 (00008042)
-----------------------------------------
Apr 3 09:39:06.050 UDLD PDU sent by 1/2 (00008042)
-----------------------------------------
Local Seq Num: 9
Remote Seq Num: 5
Local Port No: 2/1 (00008042)
Remote Port No: 2/1 (00008081)
-----------------------------------------
Apr 3 09:39:06.074 UDLD PDU Received by 1/2:
-----------------------------------------
Local Seq Num: 6
Remote Seq Num: 9
Local Port No: 2/1 (00008081)
Remote Port No: 1/2 (00008042)
-----------------------------------------
Remote Running New Version
Apr 3 09:39:06.074 Remote Port = 2/1 (00008081)
Apr 3 09:39:06.074 Last rx clock:7500
Apr 3 09:39:06.074 udld_lka->wrong_seq_cnt:0
Sending Local Port In New Version: 1/2 (00008042)
-----------------------------------------
Apr 3 09:39:12.050 UDLD PDU sent by 1/2 (00008042)
-----------------------------------------
Local Seq Num: 10
Remote Seq Num: 6
Local Port No: 2/1 (00008042)
Remote Port No: 2/1 (00008081)
-----------------------------------------
Apr 3 09:39:12.074 UDLD PDU Received by 1/2:
-----------------------------------------
Local Seq Num: 7
Remote Seq Num: 10
Local Port No: 2/1 (00008081)
Remote Port No: 1/2 (00008042)
-----------------------------------------
Remote Running New Version
Apr 3 09:39:12.074 Remote Port = 2/1 (00008081)
Apr 3 09:39:12.074 Last rx clock:7740
Apr 3 09:39:12.074 udld_lka->wrong_seq_cnt:0
```

The following is a sample output from the **debug link-keepalive packet tx** command.

```
device# debug link-keepalive packet tx
Link-Keepalive: All Ports TX debugging is on
device# Sending Local Port In New Version: 1/2 (00008042)
-----------------------------------------
Apr 3 09:41:48.050 UDLD PDU sent by 1/2 (00008042)
-----------------------------------------
Local Seq Num: 36
Remote Seq Num: 32
Local Port No: 2/1 (00008042)
Remote Port No: 2/1 (00008081)
-----------------------------------------
Sending Local Port In New Version: 1/2 (00008042)
-----------------------------------------
Apr 3 09:41:54.050 UDLD PDU sent by 1/2 (00008042)
-----------------------------------------
Local Seq Num: 37
Remote Seq Num: 33
Local Port No: 2/1 (00008042)
Remote Port No: 2/1 (00008081)
-----------------------------------------
Sending Local Port In New Version: 1/2 (00008042)
-----------------------------------------
Apr 3 09:42:00.050 UDLD PDU sent by 1/2 (00008042)
-----------------------------------------
Local Seq Num: 38
Remote Seq Num: 34
Local Port No: 2/1 (00008042)
Remote Port No: 2/1 (00008081)
-----------------------------------------
```

The following is a sample output from the **debug link-keepalive packet rx** command.

```
device# debug link-keepalive packet rx
Link-Keepalive: All Ports RX debugging is on
device# Apr 3 09:40:54.074 UDLD PDU Received by 1/2:
-----------------------------------------
Local Seq Num: 24
Remote Seq Num: 27
Local Port No: 2/1 (00008081)
Remote Port No: 1/2 (00008042)
-----------------------------------------
Remote Running New Version
Apr 3 09:40:54.074 Remote Port = 2/1 (00008081)
Apr 3 09:40:54.074 Last rx clock:11820
Apr 3 09:40:54.074 udld_lka->wrong_seq_cnt:0
    Apr 3 09:41:00.074 UDLD PDU Received by 1/2:
-----------------------------------------
Local Seq Num: 25
Remote Seq Num: 28
Local Port No: 2/1 (00008081)
Remote Port No: 1/2 (00008042)
-----------------------------------------
Remote Running New Version
Apr 3 09:41:00.074 Remote Port = 2/1 (00008081)
Apr 3 09:41:00.074 Last rx clock:12060
Apr 3 09:41:00.074 udld_lka->wrong_seq_cnt:0
no debug all
All possible debug messages have been turned off
```

The following example shows when debugging is enabled for a specific port for both transmit and receive PDUs.

```
device# debug link-keepalive packet eth 1/2
Link-Keepalive: Specified Ports RX TX debugging is on
device# Sending Local Port In New Version: 1/2 (00008042)
-----------------------------------------
Apr 3 09:43:30.050 UDLD PDU sent by 1/2 (00008042)
-----------------------------------------
Local Seq Num: 53
Remote Seq Num: 49
Local Port No: 2/1 (00008042)
Remote Port No: 2/1 (00008081)
-----------------------------------------
Apr 3 09:43:30.074 UDLD PDU Received by 1/2:
-----------------------------------------
Local Seq Num: 50
Remote Seq Num: 53
Local Port No: 2/1 (00008081)
Remote Port No: 1/2 (00008042)
-----------------------------------------
Remote Running New Version
Apr 3 09:43:30.074 Remote Port = 2/1 (00008081)
Apr 3 09:43:30.074 Last rx clock:18060
Apr 3 09:43:30.074 udld_lka->wrong_seq_cnt:0
Sending Local Port In New Version: 1/2 (00008042)
-----------------------------------------
Apr 3 09:43:36.050 UDLD PDU sent by 1/2 (00008042)
-----------------------------------------
Local Seq Num: 54
Remote Seq Num: 50
Local Port No: 2/1 (00008042)
Remote Port No: 2/1 (00008081)
-----------------------------------------
Apr 3 09:43:36.074 UDLD PDU Received by 1/2:
-----------------------------------------
Local Seq Num: 51
Remote Seq Num: 54
Local Port No: 2/1 (00008081)
Remote Port No: 1/2 (00008042)
-----------------------------------------
Remote Running New Version
Apr 3 09:43:36.074 Remote Port = 2/1 (00008081)
Apr 3 09:43:36.074 Last rx clock:18300
Apr 3 09:43:36.074 udld_lka->wrong_seq_cnt:0
```

The following is a sample output from the **debug link-keepalive packet eth** *slot/port* **tx** command.

```
device# debug link-keepalive packet eth 1/2 tx
Link-Keepalive: Specified Ports TX debugging is on
device# Sending Local Port In New Version: 1/2 (00008042)
-----------------------------------------
Apr 3 09:45:24.050 UDLD PDU sent by 1/2 (00008042)
-----------------------------------------
Local Seq Num: 72
Remote Seq Num: 68
Local Port No: 2/1 (00008042)
Remote Port No: 2/1 (00008081)
-----------------------------------------
Sending Local Port In New Version: 1/2 (00008042)
-----------------------------------------
Apr 3 09:45:30.050 UDLD PDU sent by 1/2 (00008042)
-----------------------------------------
Local Seq Num: 73
Remote Seq Num: 69
Local Port No: 2/1 (00008042)
Remote Port No: 2/1 (00008081)
-----------------------------------------
Sending Local Port In New Version: 1/2 (00008042)
-----------------------------------------
Apr 3 09:45:36.050 UDLD PDU sent by 1/2 (00008042)
-----------------------------------------
Local Seq Num: 74
Remote Seq Num: 70
Local Port No: 2/1 (00008042)
Remote Port No: 2/1 (00008081)
```

The following is a sample output from the **debug link-keepalive packet eth** *slot/port* **rx** command.

```
device# debug link-keepalive packet eth 1/2 rx
Link-Keepalive: Specified Ports RX debugging is on
device# Apr 3 09:44:18.074 UDLD PDU Received by 1/2:
-----------------------------------------
Local Seq Num: 58
Remote Seq Num: 61
Local Port No: 2/1 (00008081)
Remote Port No: 1/2 (00008042)
-----------------------------------------
Remote Running New Version
Apr 3 09:44:18.074 Remote Port = 2/1 (00008081)
Apr 3 09:44:18.074 Last rx clock:19980
Apr 3 09:44:18.074 udld_lka->wrong_seq_cnt:0
Apr 3 09:44:24.074 UDLD PDU Received by 1/2:
-----------------------------------------
Local Seq Num: 59
Remote Seq Num: 62
Local Port No: 2/1 (00008081)
Remote Port No: 1/2 (00008042)
-----------------------------------------
Remote Running New Version
Apr 3 09:44:24.074 Remote Port = 2/1 (00008081)
Apr 3 09:44:24.074 Last rx clock:20220
Apr 3 09:44:24.074 udld_lka->wrong_seq_cnt:0
Apr 3 09:44:30.074 UDLD PDU Received by 1/2:
-----------------------------------------
Local Seq Num: 60
Remote Seq Num: 63
Local Port No: 2/1 (00008081)
Remote Port No: 1/2 (00008042)
-----------------------------------------
Remote Running New Version
Apr 3 09:44:30.074 Remote Port = 2/1 (00008081)
Apr 3 09:44:30.074 Last rx clock:20460
Apr 3 09:44:30.074 udld_lka->wrong_seq_cnt:0
```

The following example displays debugging parameters for link-keepalive protocol.

```
device# debug link-keepalive show
Link-keepalive debug parameters
-------------------------------
RX debugging is on for ports ethe 1/2
TX debugging is on for ports ethe 1/2
```

debug lldp

Enables LLDP-related debugging.

Syntax

debug lldp [**port ethernet** *slot/port*]

no debug lldp [**port ethernet** *slot/port*]

Parameters

port ethernet *slot/port*

Limits the display of LLDP port state change information to the specific Ethernet port.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays information about LLDP port state changes.

```
device# 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
```

The following example limits the display of LLDP port state change information to the specific Ethernet port.

```
device# 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
```

debug loopdetect

Generates loop detection information.

Syntax

debug loopdetect [**detail** | **error** | **info**]

no debug loopdetect [**detail** | **error** | **info**]

Parameters

detail

Displays loop detect detail messages.

error

Displays loop detect error messages.

info

Displays loop detect information messages.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following command example displays loop detect information messages.

```
device# 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.

```
device# 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
```

debug mac

generates information about MAC address databases, actions, settings, MAC table management, and MAC learning.

Syntax

debug mac [**action** [**vlan** *vlan-id*] [*mac-address*] | **cam** | **error** | **info** | **learning** [**vlan** *vlan-id*] [*mac-address*] | **mport** | **port security**]

no debug mac [**action** [**vlan** *vlan-id*] [*mac-address*] | **cam** | **error** | **info** | **learning** [**vlan** *vlan-id*] [*mac-address*] | **mport** | **port security**]

Parameters

action
Displays information about the MAC database.

vlan *vlan-id*
Displays all MAC address-related actions for the specified VLAN.

mac-address
Displays specified MAC address-related actions for all VLANs.

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.

vlan *vlan-id*
Displays all MAC address-related learning for the specified VLAN.

mac-address
Displays specified MAC address-related learning for all VLANs.

mport
Displays Multiport MAC event messages.

port security
Displays MAC table management port security messages.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

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

Examples

The following example displays actions-related information for the MAC database.

```
device# 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.

```
device# 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
```

The following example displays 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.

```
device# 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
```

The following example generates information about MAC database activity, such as flushing and table distribution.

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

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

```
device# 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.

```
device# 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.

```
device# 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 mmrp

Enables MMRP debugging.

Syntax

debug mmrp [**cli** | **config** | **db-event** | **error-event** | **ethernet** | **event** | **itc** | **pdu** | **reset** | **rx-event** | **show** | **sm-event** | **timer** | **tx-event** | **verbose**]

no debug mmrp [**cli** | **config** | **db-event** | **error-event** | **ethernet** | **event** | **itc** | **pdu** | **reset** | **rx-event** | **show** | **sm-event** | **timer** | **tx-event** | **verbose**]

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays the current MMRP debugging parameters.

```
device# 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
```

debug mpls

Turns on the debugging output, while preserving the debugging configuration.

Syntax

debug mpls [**all** | **api** | **dynamic-bypass** | **error** | **lsp-convergence** | **oam** | **rsir**]

Parameters

all
Turns on debugging for all MPLS configurations.

api
Display debugging information about MPLS API.

dynamic-bypass
Enables MPLS dynamic bypass debugging.

error
Turns on error debugging for all the modules.

lsp-convergence
Enables LSP convergence debugging.

oam
Specifies MPLS OAM debug.

rsir
Specifies MPLS RSIR debug.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

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.

Examples

The following example turns on the debugging output.

```
device# debug mpls
```

This example turns on error debugging for all the modules.

```
device# debug mpls error
device# show debug
MPLS Debug Settings
  MPLS Debug is OFF
  Error
  All
```

The following example enables and displays the dynamic bypass-related logs on the console.

```
device# debug mpls dynamic-bypass all
MPLS: dynamic-bypass debugging is on
```

The following example displays debugging information about MPLS API.

```
device# debug mpls api
device# 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
Device# no router isis
Aug 18 21:32:10 mpls_app_deregister_igp_shortcut app_type MPLS_APP_ISIS is
de-registered
Device# 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
Device# 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
Device# 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
Device# 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
Device# next-hop-mpls
178 Device MLX Series and NetIron XMR Diagnostic Guide
53-1004194-02
5 MPLS API
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
Device# 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 cspf

Displays information on the Constrained Shortest Path First (CSPF) computations, mapping, TE databases, and errors.

Syntax

debug mpls cspf [**computation** | **mapping** | **ted** | **error** | **all**]

no debug mpls cspf [**computation** | **mapping** | **ted** | **error** | **all**]

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

This example displays all debug error messages related to the CSPF module.

```
device# debug mpls cspf all
```

debug mpls cspf computation

Displays CSPF computation information filtered by more specific criteria. If an LSP does not come up with the error code “No path found”, it means that CSPF could not calculate a path for the destination with the specified set of constraints.

Syntax

debug mpls cspf computation [**all** | **detail** | **lsp**]

no debug mpls cspf computation [**all** | **detail** | **lsp**]

Parameters

all

Displays all debug messages related to the CSPF computation.

detail

Displays more detailed information about the CSPF computation.

lsp

Displays CSPF computation messages for specific LSPs.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

This example displays CSPF computation information.

```
device# 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

Displays the CSPF computation information for specific LSPs.

Syntax

debug mpls cspf computation lsp [**name** *name* | **sess-object** *source-ipddress destination-ip-address tunnel-id*]

no debug mpls cspf computation lsp [**name** *name* | **sess-object** *source-ipddress destination-ip-address tunnel-id*]

Parameters

all

Limits the display of information to debug messages for the LSP that matches with the specified LSP name.

sess_obj *source-ipddress 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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

This example displays the CSPF computation information for specific LSPs.

```
device# 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
```

debug mpls forwarding

Displays the MPLS forwarding debugging information.

Syntax

debug mpls forwarding [**all** | **error** | **ldp** [**fec-key** | **all** | **prefix** | **vc**] | **rm** | **rsvp**]

no debug mpls forwarding [**all** | **error** | **ldp** [**fec-key** | **all** | **prefix** | **vc**] | **rm** | **rsvp**]

Parameters

all
Enables or disables all forwarding debugging.

error
Enables or disables logging of forwarding debug error messages.

ldp fec-key | **ldp all** | **ldp prefix** | **ldp vc**
Enables or disables LDP forwarding debugging.

rm
Enables or disables the Resource Manager forwarding debugging.

rsvp
Enables or disables RSVP forwarding debugging. Refer to the **debug mpls forwarding rsvp** command reference page for detailed information.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following **debug mpls forwarding all** command turns on all MPLS forwarding debugging.

```
device# debug mpls forwarding all
```

The following **show debug** command output verifies that all MPLS forwarding debugging is turned on.

```
device# show debug
MPLS Debug Settings
    MPLS Debug is ON
    Forwarding
      All
Debug message destination:  Console
```

The following **debug mpls forwarding rm** command turns on MPLS forwarding debugging for the Resource Manager.

```
device# debug mpls forwarding rm
```

The following **debug mpls forwarding error** command turns on logging of MPLS forwarding debug error messages.

```
device# debug mpls forwarding error
```

The following **show debug** command output verifies that logging of MPLS forwarding debug error messages is turned on and debugging for the Resource Manager is turned on.

```
device# show debug
MPLS Debug Settings
    MPLS Debug is ON
    Forwarding
      Error
      RM
Debug message destination:  Console
```

The following **no debug mpls forwarding rm** command turns off MPLS forwarding debugging for the Resource Manager.

```
device# no debug mpls forwarding rm
```

The following **show debug** command output verifies that logging of MPLS forwarding debug error messages is turned on, but the debugging for the Resource Manager is turned off.

```
device# show debug
MPLS Debug Settings
    MPLS Debug is ON
    Forwarding
      Error
Debug message destination:  Console
```

The following **debug mpls forwarding ldp fec-key all** command turns on MPLS forwarding debug messages related to all LDP Forwarding Equivalence Classes (FECs).

```
device# debug mpls forwarding ldp fec-key all
```

The following **debug mpls forwarding ldp fec-key prefix 1.2.3.4** command turns on debug messages related to the LDP tunnel to destination 1.2.3.4.

```
device# debug mpls forwarding ldp fec-key prefix 1.2.3.4
```

The following **debug mpls forwarding ldp fec-key vc 100** command turns on debug messages related to the LDP virtual circuit (VC) FEC (for the VLL or VPLS instance) with VC ID 100.

```
device# debug mpls forwarding ldp fec-key vc 100
```

When the **debug mpls forwarding all** command is enabled, the debug messages are displayed as shown in the following example:

```
device(config-mpls-lsp-to_30)#enable
Connecting signaled LSP to_30
Sep  6 03:18:53.611 RLDF: In attempt to recover lsp_cb, DIDNOT Recover
Sep  6 03:18:53.611 RLDF: ADD lsp_cb(0X3499EF58), lsp_xc_idx=0X00000003, lsp_sync_index=0X00000000,
sync_info: 0X00000000
Sep  6 03:18:53.611 RLDF: Add in_cb(0X3248D26C),lblsp-idx=0, in-if=0,in-lbl=0 lsp_cb 0X3499EF58
Sep  6 03:18:53.611 RLDF : Check BW
Sep  6 03:18:53.611    Path_TSpec valid: BW = 0 Kb/sec
device(config-mpls)#
Sep  6 03:18:53.630 RLDF: In attempt to recover lsp_cb, DIDNOT Recover
Sep  6 03:18:53.630 RLDF: ADD out_cb(0X3498AF48), out-s-idx=3, out-int=1, out-lbl=2048 lsp_cb=0X3499EF58
Sep  6 03:18:53.630 RLDF: ADD xc_cb(0X3498A780), in_cb=0X3248D26C, out_cb=0X3498AF48, lsp_cb=0X3499EF58
Sep  6 03:18:53.630 RLDF : Check BW
Sep  6 03:18:53.630    Path_TSpec valid: BW = 0 Kb/sec
Sep  6 03:18:53.631    Resv_TSpec valid: BW = 0 Kb/sec
Sep  6 03:18:53.631 Alloc BW[outseg idx: 3]: setup/hold priority 7/0:
Sep  6 03:18:53.631    Path_TSpec valid: BW = 0 Kb/sec
Sep  6 03:18:53.631    Resv_TSpec valid: BW = 0 Kb/sec
Sep  6 03:18:53.631 Allocated BW 0 kbps for priority 0 on e1/1
Sep  6 03:18:53.631 RLDF: Update XC: lsp_cb 0X00000000, in_cb 0X00000000, out_cb 0X3498AF48, xc_cb
0X3498A780
Sep  6 03:18:53.631 RLDF: Update XC: lsp_xc_id 3, in-lbl 0, in-if port_id 65535, out-seg_idx 3, out-if
e1/1
Sep  6 03:18:53.633 RLDF: update_tnnl_vif_nht_index: Old 65535, new 1
Sep  6 03:18:53.634 RLDF - tnl 0 goes up

Sep  6 03:18:53.634 RLDF: Ingress tunnel(to_30) VIF index (0) tunnel accounting action=0

SYSLOG: <13>Sep  6 03:18:53 DUT12 MPLS: LSP to_30 using path <NULL> is up
```

debug mpls forwarding rsvp

Displays the RSVP forwarding debugging information.

Syntax

debug mpls forwarding rsvp [**all** | **lsp session-obj** [**all** | *source-ipddress destination-ip-address tunnel-id*]]

no debug mpls forwarding rsvp [**all** | **lsp session-obj** [**all** | *source-ipddress destination-ip-address tunnel-id*]]

Parameters

all

Enables or disables all RSVP forwarding debugging.

lsp session-obj all

Enables or disables all RSVP session object debugging.

lsp session-obj *source-ipddress destination-ip-address tunnel-id*

Enables or disables RSVP session object debugging for the specified RSVP session object.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The **debug mpls forwarding rsvp all** command output resembles the following example.

```
device# 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.

```
device# debug mpls forwarding rsvp lsp sess-obj 10.110.110.1 10.130.130.1 13
device# (config)# router mpls
device#(config-mpls)# lsp t-2
device#(config-mpls-lsp-t-2)# to 10.130.130.3
device#(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 ldp

Displays the MPLS LDP-related information.

Syntax

debug mpls ldp [**all** | **adjacency** | **error** | **event** | **fec** | **gr** | **packet** | **socket** | **state** | **tcpdump** | **tunnel**]

no debug mpls ldp [**all** | **adjacency** | **error** | **event** | **fec** | **gr** | **packet** | **socket** | **state** | **tcpdump** | **tunnel**]

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays all MPLS LDP-related information.

```
device# 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

Displays information on MPLS LDP adjacencies.

Syntax

debug mpls ldp adjacency

no debug mpls ldp adjacency

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays information on MPLS LDP adjacencies.

```
device# 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

Displays the errors detected by LDP components such as LDP session keepalives timer expiration, hello adjacency timeout, and so on.

Syntax

debug mpls ldp error

no debug mpls ldp error

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays the errors detected by LDP components such as LDP session keepalives timer expiration, hello adjacency timeout, and so on.

```
device# debug mpls ldp error
debug mpls ldp error
LDP_ERR: Session KeepAlive timer to peer 10.120.120.2 has expired
```

debug mpls ldp event

Displays debug messages related to MPLS LDP events.

Syntax

debug mpls ldp event

no debug mpls ldp event

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays debug messages related to MPLS LDP events.

```
device# 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

Displays the MPLS LDP FEC information.

Syntax

debug mpls ldp fec [**all** | **lsr-id** *ip-address label-space* | **key** [**prefix** *ip-address prefix-length* | **vc** *vc-id*]]

no debug mpls ldp fec [**all** | **lsr-id** *ip-address label-space* | **key** [**prefix** *ip-address prefix-length* | **vc** *vc-id*]]

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays the MPLS LDP FEC information.

```
device# debug mpls ldp fec
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

Displays debug messages related to MPLS LDP Graceful Restart (GR) including internal GR FSM transitions.

Syntax

debug mpls ldp gr

no debug mpls ldp gr

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays debug messages related to MPLS LDP GR and FSM transitions.

```
device# 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

Displays debug messages related to MPLS LDP packets which is further filtered by direction, packet types, and LSR ID.

Syntax

debug mpls ldp packets [**all** | **detail** | **direction** | **lsr-id** | **pkt-type**]

no debug mpls ldp packets [**all** | **detail** | **direction** | **lsr-id** | **pkt-type**]

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays the debug messages related to MPLS LDP packets.

```
device# 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 direction

Limits the display of MPLS LDP packets to specific directions (send and receive).

Syntax

debug mpls ldp packets direction [**send** | **receive**]

no debug mpls ldp packets direction [**send** | **receive**]

Parameters

send

Displays information on the sent LDP packets.

receive

Displays information on the received LDP packets.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays packets limited to specific send directions.

```
device# debug mpls ldp packets direction send
```

The following example displays packets limited to specific receive directions.

```
device# debug mpls ldp packets direction receive
```

debug mpls ldp packets lsr_id

Displays MPLS LDP packets for the specified LDP LSR ID.

Syntax

debug mpls ldp packets lsr-id *ip-address label-space-id*

no debug mpls ldp packets lsr-id *ip-address label-space-id*

Parameters

ip-address
Specifies the IP address.

label-space-id
Specifies the label ID.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays MPLS LDP packets for the specified LDP LSR ID.

```
device# debug mpls ldp packets lsr-id 10.1.1.2 10
```

debug mpls ldp packets pkt_type

Displays the MPLS LDP packets of specific packet types.

Syntax

debug mpls ldp packets pkt_type [**all** | **address** | **initialization** | **label** | **notification** | **hello** | **keepalive**]

no debug mpls ldp packets pkt-type [**all** | **address** | **initialization** | **label** | **notification** | **hello** | **keepalive**]

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays information about MPLS LDP addresses, including address withdraw messages.

```
device# 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

Displays MPLS LDP addresses, including address withdraw messages.

Syntax

debug mpls ldp packets pkt_type address

no debug mpls ldp packets pkt-type address

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays MPLS LDP addresses, including address withdraw messages.

```
device# debug mpls ldp packets 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 hello

Displays information about the periodic link Hello messages sent and received.

Syntax

debug mpls ldp packets pkt_type hello

no debug mpls ldp packets pkt-type hello

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays information about the periodic link Hello messages sent and received.

```
device# 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 pkt_type initialization

Displays LDP Initialization messages.

Syntax

debug mpls ldp packets pkt_type initialization

no debug mpls ldp packets pkt-type initialization

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays LDP Initialization messages.

```
device# 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

Displays the LDP notification which is generated when an unexpected event occurs.

Syntax

debug mpls ldp packets pkt_type notification

no debug mpls ldp packets pkt-type notification

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays an LDP notification.

```
device# 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 socket

Displays debug messages related to MPLS LDP sockets.

Syntax

debug mpls ldp socket

no debug mpls ldp socket

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays debug messages related to MPLS LDP sockets.

```
device# 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

Displays the MPLS LDP states.

Syntax

debug mpls ldp state

nodebug mpls ldp state

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

The following MPLS LDP state are possible:

- 0—Unknown
- 1—No route
- 2—Route found
- 3—Path sent
- 4—Path error

Examples

The following example displays the MPLS LDP states.

```
device# 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

Displays the MPLS LDP packets as raw data.

Syntax

debug mpls ldp tcpdump

no debug mpls ldp tcpdump

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays the MPLS LDP packets as raw data.

```
device# 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

Displays debug messages related to MPLS LDP LSP interaction with other modules as virtual interfaces.

Syntax

debug mpls ldp tunnel [**all** | **prefix** *ip-address prefix-length*]

no debug mpls ldp tunnel [**all** | **prefix** *ip-address prefix-length*]

Parameters

all

Displays all MPLS LDP tunnel up and down events.

prefix *ip-address prefix-length*

Limits the display of information to specific prefixes.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays debug messages related to MPLS LDP LSP interaction with other modules as virtual interfaces.

```
device# debug mpls ldp tunnel
LDP_TUNNEL: tunnel(5) to 10.100.100.100 is up
```

debug mpls lmgr

Displays the MPLS label manager information.

Syntax

debug mpls lmgr [**all** | **error** | **ldp** | **rsvp**]

no debug mpls lmgr [**all** | **error** | **ldp** | **rsvp**]

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays the MPLS label manager information.

```
device# 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

Displays the label manager information for RSVP.

Syntax

debug mpls lmgr rsvp [**all** | **lsp** [**name** *name* | **sess_obj** *source-ip address destination-ip address tunnel-id*]]

no debug mpls lmgr rsvp [**all** | **lsp** [**name** *name* | **sess_obj** *source-ip address destination-ip address tunnel-id*]]

Parameters

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.

.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays the label manager information for RSVP.

```
device# debug mpls lmgr rsvp all
device# 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.
```

debug mpls routing

Generates the MPLS routing information.

Syntax

debug mpls routing [**all** | **error** | **interface** | **prefix**]

no debug mpls routing [**all** | **error** | **interface** | **prefix**]

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays the MPLS routing information.

```
device# 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 error

Displays debugging information related to Interior Gateway Protocol (IGP) neighbor down events.

Syntax

debug mpls routing error

no debug mpls routing error

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays the debugging information related to Interior Gateway Protocol (IGP) neighbor down events.

```
device# 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
```

debug mpls routing interface

Displays the MPLS routing-related information to the specific interfaces.

Syntax

debug mpls routing interface[**all** | **ethernet** *islot/port* | **pos** | **ve** *index*

no debug mpls routing interface[**all** | **ethernet** *islot/port* | **pos** | **ve** *index*

Parameters

all

Displays all debug messages related to MPLS routing for all the interfaces.

ethernet

Limits the messages to specific Ethernet interfaces.

slot/port

Specifies the slot and port number.

pos

Limits the messages to specific POS interfaces.

ve*index*

Limits the messages to specific virtual Ethernet interfaces.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays the MPLS routing-related information to the specific interfaces.

```
device# 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
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 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

Limits the display of MPLS routing-related information to specific prefixes.

Syntax

debug mpls routing prefix*ip-address prefixl-engath*

no debug mpls routing error*ip-address prefixl-engath*

Parameters

ip-address
Specifies the IP address.

prefixl-engath
Specifies the prefix length.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example limits the display of MPLS routing-related information to specific prefixes.

```
device# debug mpls routing prefix 10.1.1.1/23
```

debug mpls rsvp

Displays the MPLS RSVP information.

Syntax

debug mpls rsvp [**all** | **error** | **event** | **packets** | **session** | **tunnel**]

no debug mpls rsvp [**all** | **error** | **event** | **packets** | **session** | **tunnel**]

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays the MPLS RSVP information.

```
device# debug mpls rsvp all
```

debug mpls rsvp event

Displays MPLS RSVP events.

Syntax

debug mpls rsvp event

no debug mpls rsvp event

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays messages related to MPLS RSVP events.

```
device# 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

Displays the MPLS RSVP packets-related information, which is further filtered by direction, packet type, interface, and session object.

Syntax

debug mpls rsvp packets [**all** | **detail** | **count** *number* | **direction** [**send** | **receive**] | **pkt_type** [**ack** | **all** | **bundle** | **path** | **patherr** | **resv** | **resverr** | **resvtear** | **summary-fresh** | **hello**] | **interface** | **sess_obj** *source-ip-address* [*destination-ip-address* | *p2mp-id*] *tunne-id*]

no debug mpls rsvp packets [**all** | **detail** | **count** *number* | **direction** [**send** | **receive**] | **pkt_type** [**ack** | **all** | **bundle** | **path** | **patherr** | **resv** | **resverr** | **resvtear** | **summary-fresh** | **hello**] | **interface** | **sess_obj** *source-ip-address* [*destination-ip-address* | *p2mp-id*] *tunne-id*]

Parameters

all
Displays all messages related to MPLS RSVP packets.

detail
Displays debug messages related to MPLS RSVP errors.

count *number*
Displays detailed information about MPLS RSVP packets.

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 acknowledgment messages.

ack
Displays information about MPLS RSVP reservation request acknowledgment messages.

all
Turns on or off debugging of all MPLS RSVP packet types.

bundle
Displays MPLS RSVP bundle messages.

path
Displays MPLS RSVP path messages.

patherr
Displays MPLS RSVP path error messages.

pathtear
Displays MPLS RSVP path tear messages.

resv
Displays MPLS RSVP reservation request messages.

resverr
Displays MPLS RSVP reservation request error messages.

resvtear
Displays MPLS RSVP reservation tear messages.

summary-refresh
Displays MPLS RSVP summary refresh messages.

hello
Displays MPLS RSVP hello messages.

interface
Displays RSVP packets transmitted or received on an interface.

sess_obj *source-ip-address*
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, destinationIP address, and tunnel ID. P2MP session object incudes source IP address, P2MP ID, and tunnel ID.

destination-ip-address
Specifies the destination IP address.

p2mp-id
Specifies the P2MP ID.

tunnel-id
Specifies the tunnel ID.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays the MPLS RSVP packets-related information, which is further filtered by direction, packet type, interface, and session object.

```
device# 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
```

The following example displays MPLS RSVP hello messages.

```
device# debug mpls rsvp packets pkt_type hello
Feb 27 21:58:03.367
Feb 27 21:58:03.367 Send Hello message: src 10.31.31.16, dst 10.31.31.15 on e4/3
Feb 27 21:58:03.368
Feb 27 21:58:03.368 Receive Hello message: src 10.31.31.15, dst 10.31.31.16 on
e4/3
```

debug mpls rsvp packets detail

Displays the detailed information about MPLS RSVP packets.

Syntax

debug mpls rsvp packets detail

no debug mpls rsvp packets detail

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays the detailed information about MPLS RSVP packets.

```
device# debug mpls rsvp packets detail
Feb 27 22:00:09.368
Feb 27 22:00:09.368 Send Hello message: src 10.31.31.16, dst 10.31.31.15 on e4/3
Feb 27 22:00:09.368 Type 20(Hello) ver 1 flags 0x00 cksum 0xa0d0 ttl 1 len 20
Feb 27 22:00:09.368 Obj_class 22 (HELLO) ctype 1 (HELLO_REQ) length 12
Feb 27 22:00:09.368 Source instance: 0x0002ade9 , Destination instance:
0x0001d1ae
Feb 27 22:00:09.368
Feb 27 22:00:09.368 Receive Hello message: src 10.31.31.15, dst 10.31.31.16 on
e4/3
Feb 27 22:00:09.368 Type 20(Hello) ver 1 flags 0x00 cksum 0xa0cf ttl 1 len 20
Feb 27 22:00:09.369 Obj_class 22 (HELLO) ctype 2 (HELLO_ACK) length 12
Feb 27 22:00:09.369 Source instance: 0x0001d1ae , Destination instance:
0x0002ade9
Feb 27 22:00:09.369
```

debug mpls rsvp packets interface

Displays the MPLS RSVP packets for specific interfaces.

Syntax

debug mpls rsvp packets interface [**all** | **ethernet** *slot/port* | **pos** *slot/port* | **ve** *index*]

no debug mpls rsvp packets interface [**all** | **ethernet** *slot/port* | **pos** *slot/port* | **ve** *index*]

Parameters

all
Displays all debug messages related to MPLS routing for all the interfaces.

ethernet
Limits the messages to specific Ethernet interfaces.

slot/port
Specifies the slot and port number.

pos
Limits the messages to specific POS interfaces.

ve*index*
Limits the messages to specific virtual Ethernet interfaces.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following examples displays MPLS RSVP packets limited to specific interfaces.

```
device# 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

Displays information about the MPLS RSVP packets for the specified RSVP session object or Point to Multipoint (P2MP) session object.

Syntax

debug mpls rsvp packets sess_obj*source-ip-address* [*destination-ip-address* | *p2mp-Id*] *tunnel-id*

no debug mpls rsvp packets sess_obj*source-ip-address* [*destination-ip-address* | *p2mp-Id*] *tunnel-id*

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays information about the MPLS RSVP packets for the specified RSVP session object or Point to Multipoint (P2MP) session object.

```
device# 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
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

Displays the MPLS RSVP session-related information.

Syntax

debug mpls rsvp session [**all** | **detail** | **lsp**]

no debug mpls rsvp session [**all** | **detail** | **lsp**]

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following examples displays the MPLS RSVP session-related information.

```
device# 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

Displays the MPLS RSVP session-related information.

Syntax

debug mpls rsvp session lsp [**name** *name* **sess_obj** *source-ip-address* [*destination-ip-address* | *p2mp-Id*] *tunnel-id*

no debug mpls rsvp session lsp [**name** *name* **sess_obj** *source-ip-address* [*destination-ip-address* | *p2mp-Id*] *tunnel-id*

Parameters

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.

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following examples displays the MPLS RSVP session-related information.

```
device# 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

Displays the MPLS RSVP LSP tunnel interface-related information.

Syntax

debug mpls rsvp tunnel [**all** | **detail** | **lsp**]

no debug mpls rsvp tunnel [**all** | **detail** | **lsp**]

Parameters

all

Displays all messages related to an 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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following examples displays the MPLS RSVP LSP tunnel interface-related information.

```
device# 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

Displays the RSVP tunnel information for specific LSPs.

Syntax

debug mpls rsvp tunnel lsp [**name** *name* **sess_obj** *source-ip-address* [*destination-ip-address* | *p2mp-Id*] *tunnel-id*

no debug mpls rsvp tunnel lsp [**name** *name* **sess_obj** *source-ip-address* [*destination-ip-address* | *p2mp-Id*] *tunnel-id*

Parameters

name *name*
: Limits the display of information to debug messages for the specified LSP name.

sess_obj
: Limits the display of information to debug messages for the specified LSP session object.

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following examples displays the RSVP tunnel information for specific LSP PE3.

```
device# debug mpls rsvp tunnel lsp name PE3
device# debug mpls
device# 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)
```

debug mrp

Enables MRP debugging.

Syntax

debug mrp [**bpdu** | **diagnostics** | **event**]

no debug mrp [**bpdu** | **diagnostics** | **event**]

Parameters

bpdu

Displays error message whenever a bridge protocol data unit (BPDU) is lost on the master node (not shown for member nodes).

diagmostics

Displays MRP diagnostic information.

event

Displays information about MRP events.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

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

Examples

The following example displays an error message whenever a bridge protocol data unit (BPDU) is lost on the master node (not shown for member nodes).

```
device# 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
```

The following example displays MRP diagnostic information.

```
device# 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
```

The following example displays information about MRP events.

```
device# 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
```

debug mstp

Enables MSTP-related debugging.

Syntax

debug mstp [**bpdu** | **event** | **mstid** *num* [**region** *region-id*] | **port** [**ethernet** *slot/port* | **pos** *slot/port*] | **region** *region-id* | **show** | **state** | **verbose**]

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

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example generates debugging information for the configured MSTP debugging parameters.

```
device# 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
```

The following example displays MSTP BPDUs.

```
device# 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
```

The following example displays MSTP state events.

```
device# 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
```

The following example displays debugging information for a specific MSTP instance in the configured regions.

```
device# 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
```

The following example displays debugging information for a specific MSTP port in the configured regions.

```
device# debug mstp port ethernet 1/15
MSTP debugging turned on for ports ethernet 1/15
```

The following example displays debugging information related to a particular MSTP PB or PBB region.

```
device# 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
```

The following example displays the current MSTP debug parameters.

```
device# 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
```

The following example displays debugging information related to MSTP port state events.

```
device# 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
```

The following example displays debugging information for the configured MSTP debug parameters in the verbose mode.

```
device# 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
```

debug mvrp

Enables MVRP-related debugging.

Syntax

debug mvrp [**cli** | **config** | **db-event** | **error-event** | **ethernet** | **event** | **itc** | **pdu** | **reset** | **rx-event** | **show** | **sm-event** | **timer** | **tx-event** | **verbose**]

no debug mvrp [**cli** | **config** | **db-event** | **error-event** | **ethernet** | **event** | **itc** | **pdu** | **reset** | **rx-event** | **show** | **sm-event** | **timer** | **tx-event** | **verbose**]

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays the current MVRP debugging parameters.

```
device# 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
```

debug openflow

Enables logging of OpenFlow-related messages.

Syntax

debug openflow [**config** | **flow-all** | **forwarding** | **hardware** | **ipc** | **meter** | **opm** | **show**]

no debug openflow [**config** | **flow-all** | **forwarding** | **hardware** | **ipc** | **meter** | **opm** | **show**]

Parameters

config
Displays OpenFlow configuration debug messages.

flow-all
Displays all the OpenFlow flow debug messages.

forwarding
Displays OpenFlow forwarding debug messages.

hardware
Displays OpenFlow hardware debug messages.

ipc
Displays OpenFlow Interprocess Communication (IPC) debug messages.

meter
Displays OpenFlow meter related debug messages.

opm
Enables debug for the OPM task.

show
Displays OpenFlow debug flags.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays OpenFlow configuration debug messages.

```
device# 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
```

The following example displays all the OpenFlow flow debug messages.

```
device# 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
```

The following example displays OpenFlow forwarding debug messages.

```
device# 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
```

The following example displays OpenFlow hardware debug messages.

```
device# 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.

```
device# 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.

```
device# 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
```

The following example displays OpenFlow IPC debug messages.

```
device# 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
```

The following example displays OpenFlow meter related debug messages.

```
device# debug openflow meter
Apr 28 13:35:59.400 OPENFLOW METER: Adding Meter with Meter :1023
Apr 28 13:35:59.400 OPENFLOW METER: Building Meter Band for Meter :1023
Apr 28 13:35:59.400 OPENFLOW METER: openflow_fill_meter_band_entry, Meter
Band:DROP, rate:3000 burst size:1250
Apr 28 13:35:59.400 OPENFLOW METER: Building Meter Band for Meter :1023
Apr 28 13:35:59.400 OPENFLOW METER: openflow_fill_meter_band_entry, Meter
Band:DSCP REMARK, rate:1700 burst size:1250 prec_level:27
Average rate is adjusted to 1693952 bits per second.
Average excess rate is adjusted to 2996992 bits per second.
Apr 28 13:35:59.400 OPENFLOW METER: METER 1023 info is added to DB
Apr 28 13:35:59.400 Committed rate 1693952, max_burst 1250
Apr 28 13:35:59.400 Excess rate 2996992, max_burst 1250
Apr 28 13:35:59.400 OPENFLOW METER: openflow_lp_calculate_meter_param: EIR
re-adjust, cir 109, eir 85
Apr 28 13:35:59.400 OPENFLOW METER: openflow_lp_calculate_meter_param: EIR
re-adjust, cir 109, eir 85
Apr 28 13:35:59.400 OPENFLOW METER: METER 1023 is succesfully created in hardware
```

The following example enables debug for the OPM task.

```
device# debug openflow opm
OPM Trace: received OFPT_HELLO. xid 1
```

The following example displays OpenFlow debug flags.

```
device# 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 commands Q-Z

debug rstp

Enables debugging of RSTP over PBB environment.

Syntax

debug rstp [**bpdu** | **event** | **mct** | **port** [**ethernet** *slot/port* | **pos** *slot/port*] | **reset** | **show** | **verbose** | **vlan** *vlan-id* | **vpls-id** *vpls-id*]

no debug rstp [**bpdu** | **event** | **mct** | **port** [**ethernet** *slot/port* | **pos** *slot/port*] | **reset** | **show** | **verbose** | **vlan** *vlan-id* | **vpls-id** *vpls-id*]

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example generates debugging information for the configured RSTP debugging parameters.

```
device# 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
```

The following example enables debugging of RSTP BPDUs.

```
device# debug rstp bpdu
RSTP Bpdu debugging ON
```

The following example enables debugging of RSTP non-BPDU events.

```
device# debug rstp event
RSTP Event debugging ON
```

The following example enables debugging of RSTP MCT events.

```
device# debug rstp mct
RSTP MCT debugging ON
```

The following example displays debugging information for a specific RSTP instance on the ports.

```
device# debug rstp port ethernet 1/2
RSTP debugging turned on for ports ethernet 1/2
```

The following example displays the current RSTP debugging parameters.

```
device# 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
```

The following example displays debugging information for the configured RSTP debugging parameters in the verbose mode.

```
device# debug rstp verbose
RSTP debugging set to VERBOSE mode
device# debug rstp show
RSTP Debug Parameters
---------------------
RSTP debugging is OFF [Mode: Verbose]
BpduEvents are being tracked
Ports: All
VLANs:
VPLSs: 1
device# 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
```

The following example enables RSTP debugging for a specific VLAN and displays debugging information for that VLAN.

```
device# debug rstp vlan 2
RSTP debugging turned on for VLAN instance 2
```

The following example enables RSTP debugging for a specific VPLS instance and displays debugging information for that VPLS instance.

```
device# debug rstp vpls-id 1
RSTP debugging turned on for vpls instance 1
device# 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
```

debug snmp client

Enables debugging of the SNMP PDUs.

Syntax

debug snmp client [*ipv4-addr* | *ipv6-addr* | **all**]

no debug snmp client [*ipv4-addr* | *ipv6-addr* | **all**]

debug snmp client [**show**]

no debug snmp client [**show**]

Parameters

ipv4-addr

Enables SNMP debugging for the specified IPv4 client.

ipv6-addr

Enables SNMP debugging for the specified IPv6 client.

all

Removes any existing individual client related debug configuration and enables SNMP debugging for all SNMP PDUs.

show

Displays the debug configuration for SNMP.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Command Output

The **debug snmp client** command displays the following information:

Output field	Description
IP	The IP address of the client which sent the request.
Request-id	The type of the SNMP PDU being processed and it can be any one of the following: Get, GetNext, GetBulk, Set
User	The SNMP user can be any one of the following: v1, v2c, v3. This parameter will be displayed only for SNMPv3 requests.
Queue depth	The queue depth of the SNMP request. This parameter will be displayed only when the request is queued in the SNMP engine.
Queue time	The queue time of the SNMP request. This parameter will be displayed only when the request is queued in the SNMP engine.

Output field	Description
error status	The error status of the SNMP response. This parameter will be printed only on error index errors in SNMP response.
error index	The error index of the SNMP response. This parameter will be printed only on error index errors in SNMP response.

Examples

The following example enables SNMP debugging for a specific IPv4 client.

```
device# debug snmp client 172.20.1.181
SNMP debugging enabled for client 172.20.1.181
```

The following example enables SNMP debugging for a specific IPv6 client.

```
device# debug snmp client 2001:DB8::4
SNMP debugging enabled for client 2001:DB8::4`
```

The following example enables SNMP debugging for all clients.

```
device# debug snmp client all
Warning: Enabling SNMP debugging for all clients will remove any existing debug
configuration for individual clients.
SNMP debugging enabled for all clients.
```

When the SNMP engine receives a request from the debug-enabled client, a set of debug messages as shown in the following sample output will be printed at various stages of processing.

```
Sep 26 19:43:40.604 172.20.1.181: 23045: SNMP Get request with 3 varbinds : User =
v3_user : Queue depth = 10 : Queue time = 10 milliseconds
SNMP Processing varbinds ifDescr.9, ifDescr.10, sysLocation.0
Sep 26 19:43:41.577 172.20.1.181: 23045: SNMP response error status = 1, error
index =3
Sep 26 19:43:41.600 172.20.1.181: 23045: SNMP response varbinds ifDesc.9 = Port
1/9, ifDescr.10 = Port 1/10.
```

The following example displays the debug configuration for SNMP.

```
device# debug snmp client show
SNMP debugging enabled for client(s) 172.20.1.181, 2001:DB8::4
```

debug spanning-tree

generates information about all BPDUs and spanning tree events (by default) or specific events.

Syntax

debug spanning-tree [**config-bpdu** | **event** | **port** [**ethernet** *slot/port* | **pos** *slot/port*] | **reset** | **show** | **tcn-bpdu** | **verbose** | **vlan** *vlan-id*]

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

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

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.

Examples

The following example generates information about all STP activity and events.

```
device# debug spanning-tree
STP: debugging is on
STP: Sending Config BPDU - VLAN 3 Port 2/3
  0000 00 00 00 800000000001c042 00000000
  8000000000001c042 8043 0000 0000 0000 0000
STP: Sending Config BPDU - VLAN 3 Port 2/4
  0000 00 00 00 800000000001c042 00000000
  8000000000001c042 8044 0000 0000 0000 0000
STP: Sending Config BPDU - VLAN 2 Port 2/1
```

The following example generates information about STP BPDUs.

```
device# 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
```

The following example displays information about non-BPDU events, such as timer, configuration, and so on.

```
device# 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
```

The following example displays spanning tree information about a port for a specific interface.

```
device# 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
```

The following generates information about the STP debug configuration.

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

The following example displays information about TCN BPDU events.

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

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

```
device# 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
```

The following example restricts debug output to a specific VLAN.

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

debug system trace

Performs a system debugging trace.

Syntax

debug system trace

no debug system trace

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following is a sample output of the **debug system trace** command.

```
device# 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

Enables and displays the debugging traces.

Syntax

debug system upgrade

no debug system upgrade

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example enables and displays the debugging traces.

```
device# 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

Displays information about Layer 2 trace protocol events.

Syntax

debug trace-l2 events

no debug trace-l2 events

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example command displays information about Layer 2 trace protocol events.

```
device# 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
```

debug vlan tvf-lag-lb

Enable debugging of the TVF LAG load balancing.

Syntax

debug vlan tvf-lag-lb *vlan-id*

no debug vlan tvf-lag-lb *vlan-id*

Parameters

vlan-id

Displays the TVF LAG load balancing debug messages for the specified VLAN ID.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following is the sample debug output for LAG member port up and down handling.

```
device# debug vlan tvf-lag-lb 100
TVF LAG load balance debugging is now ON for vlan 100
device(config-lag-test)# disable ethernet 5/1
devuce(config-lag-test)# [vlanmgr_tvf_lag_lb_update_port]: VLAN 100 TVF LAG load
balancing update port 5/1 DOWN
[vlan_tvf_lag_lb_fid_program]: TVF group id 257 program FID group
vlanmgr_set_hw_flooding(): VLAN: 100 TVF LAG load balancing is enabled
[vlan_tvf_lag_lb_fid_program]: TVF group id 258 program FID group


device(config-lag-test)# enable ethernet 5/1
device(config-lag-test)# [vlanmgr_tvf_lag_lb_update_port]: VLAN 100 TVF LAG load
balancing update port 5/1 UP
[vlan_tvf_lag_lb_fid_program]: TVF group id 257 program FID group
vlanmgr_set_hw_flooding(): VLAN: 100 TVF LAG load balancing is enabled
[vlan_tvf_lag_lb_fid_program]: TVF group id 258 program FID group
```

debug vll

Displays debugging information related to VLL.

Syntax

debug vll [**events** | **fsm** | **ipc**]

no debug vll [**events** | **fsm** | **ipc**]

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example enables vll Finite State Machine (FSM) debugging.

```
device# debug vll fsm
MPLS: VLL FSM(Finite State Machine) debugging is on
device# debug vll events
MPLS: VLL Events debugging is on
device# debug vll ipc
MPLS: VLL IPC debugging is on
device(config)# router mpls
device(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
device(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)
device(config-mpls-vll-mct-vll-1)# vll-peer 10.0.0.1 10.0.0.2
device(config-mpls-vll-mct-vll-1)# vlan 401
device(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

Displays information about activity in local Virtual Lease Line (VLL-Local) configurations.

Syntax

debug vll-lcaol

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

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.

```
device# 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
device(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
device(config-mpls-vll-lo-10)# vlan 501
device(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
device(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
```

debug vpls

Generate MPLS VPLS information.

Syntax

debug vpls [**count** | **events** | **failover** | **forwarding** | **generic** | **statistics** | **topology** | **fsm-trace** *vpls_id vpls_peer_ IP_ address*]

no debug vpls [**count** | **events** | **failover** | **forwarding** | **generic** | **statistics** | **topology** | **fsm-trace** *vpls_id vpls_peer_ IP_ address*]

Parameters

count
Displays information about the VPLS debug print counter.

events
Displays information about VPLS control-plane events.

failover
Displays information about VPLS failover events.

forwarding
Displays information about VPLS CPU packet forwarding.

generic
Enables generic VPLS debugging.

statistics
Displays information about VPLS statistics.

topology
Displays information about VPLS topology group events. This option is available only on the Management Processor.

fsm-trace *vpls_id vpls_peer_ IP_ address*
Displays VPLS peer FSM trace history.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

The **count** option 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.

The **forwarding** option 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

The **topology** option 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.

Examples

The following example shows an MPLS uplink being disabled and then re-enabled.

```
device# debug vpls events
device(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.
device(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.
```

The following example enables debugging of VPLS failover-related events

```
device# 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
```

The following example generates some CPU packet forwarding in regard to VPLS traffic.

```
device# 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
```

The following output was generated with **debug vpls statistics** enabled, and as a result of the **show mpls statistics vpls** command.

```
device# debug vpls statistics
debug vpls statistics is enabled
device# 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
```

The following example enables debugging of the topology group-related handling in the VPLS area.

```
device# 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
```

The following example displays VPLS peer FSM trace history.

```
device
# 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           -
```

debug vpls cam

Displays information about VPLS CAM or PRAM programming.

Syntax

debug vpls cam [**additions** | **deletions** | **updates**]

no debug vpls cam [**additions** | **deletions** | **updates**]

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays information about VPLS CAM or PRAM programming.

```
device# 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 dy-sync

Monitors all VPLS dy-sync activity.

Syntax

debug vpls dy-sync [**local** | **mac** | **remote** | **tlv**]

no debug vpls dy-sync [**local** | **mac** | **remote** | **tlv**]

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

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.

Examples

The following example generates information about local VPLS dy-sync activity.

```
device# debug vpls dy-sync local
device(config-mpls-vpls-test1)# vlan 2
device(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
```

The following example indicates that an existing VPLS MAC has been deleted and then reinstalled.

```
device# debug vpls dy-sync mac
device# 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
```

The following example indicates that an existing peer was deleted and then reinstalled.

```
device# debug vpls dy-sync remote
device(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
```

The following example information about the VPLS configuration that is being synchronized (dy-sync) between management and CPU cards.

```
device# debug vpls dy-sync tlv
device(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.
device(config-mpls)# vpls 1 1
device(config-mpls-vpls-1)# vpls-peer 10.200.200.1
device(config-mpls-vpls-1)# vlan 2
device(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 filter

Displays VPLS filtering options.

Syntax

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**]

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**]

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

debug vpls mac

Generates generic information about VPLS events such as VPLS MAC table allocation failures, and VPLS MAC table deletions.

Syntax

debug vpls mac { **errors** | **group** | **local** | **remote**]

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example generates information about local MAC activity.

```
device# 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 vsrp

Generates information about VSRP activity.

Syntax

debug vsrp [**all** | **aware** | **error** | **events** | **packets** | **show** | **state** | **verbose** | **vlan** *vlan-id*]

no debug vsrp [**all** | **aware** | **error** | **events** | **packets** | **show** | **state** | **verbose** | **vlan** *vlan-id*]

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

Examples

The following example displays information about all VSRP events.

```
device# debug vsrp all
VSRP debugging is setup for all attributes
device# 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
```

show tech-support

The **show tech-support** command is useful when collecting a large amount of information about the ExtremeRouting XMR Series and ExtremeRouting MLX Series routers for troubleshooting purposes. The output of this command is used by technical support representatives when reporting a problem.

Syntax

show tech-support [**bfd** | **bgp** | **cluster** | **fib** | **ip** | **ipv6** | **interface-link** | **isis** | **l2** | **mpls** | **oam** | **openflow** | **rtm** | **rtm6** | **sfm** | **system** | **tm**]

Parameters

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.

interface-link
Generates system and debugging information related to all the data registered and the hardware information from the physical layer, medium access layer, and optics.

isis
Generates system and debugging information specific to Intermediate System to Intermediate System (IS-IS) configurations.

ls
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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

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 subcommands 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 subcommand that 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 subcommands are executed, such as INTERNAL, MP, MP-OS, LP, and LP-OS.
- TIME-STAMP—A time stamp, with millisecond granularity, helps in determining the 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
- show tm log
- show tm statistics
- show tm non
- show tm non detail
- show tm log histogram
- show cpu histogram hold noclear
- show cpu histogram wait noclear
- itc show statistics
- itc show error list

NOTE

For TM output, pruning discards on egress TM(s) can be expected without any loss in traffic due to source port suppression for broadcast, multicast and unknown unicast traffic.

Examples

The following is an example of the **show tech-support** command.

```
device# 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

Displays system and debugging information specific to BFD configurations.

Syntax

show tech-support bfd session *session-id*

Parameters

session *session-id*
Specifies the BFD session ID.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

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**

Examples

The following is an example of the **show tech-support bfd** command.

```
device# show tech-support bfd session 5
```

show tech-support bgp

Generates system and debugging information specific to BGP configurations.

Syntax

show tech-support bgp ip [**multicast** | **unicast**] **route** *ip-prefix*

show tech-support bgp ipv6 [**multicast** | **unicast**] **route** *ipv6-prefix*

show tech-support bgp [**l2vpn vpls** | **vpnv4** | **vrf**]

Parameters

ip
Displays IPv4 commands.

multicast
Displays the multicast route information for BGP routes.

unicast
Displays the unicast route information for BGP routes.

route *ip-prefix*
Specifies the IPv4 destination prefix and mask length for the BGP route.

ipv6
Displays IPv6 commands.

route *ipv6-prefix*
Specifies the IPv6 destination prefix and mask length for the BGP route.

l2vpn vpls
Displays information related to all the BGP Layer 2 Virtual Private Network (L2VPN) Virtual Private LAN Services (VPLS) routes.

vrf
Specifies the name of the Virtual Routing and Forwarding (VRF) instance for which you want to display the configuration information.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

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**

Examples

The following is an example of the **show tech-support bgp** command.

```
device# show tech-support bgp
```

show tech-support fib

Displays system and debugging information specific to FIB configurations.

Syntax

show tech-support fib ipv4 *ip-address/iprefix*

Parameters

ipv4 *ip-address/iprefix*
Specifies the destination IPv4 address and IP subnet mask length.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

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 an 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**

Examples

The following is an example of the **show tech-support fib** command.

```
device# show tech-support fib
```

show tech-support interface-link

Collects debugging information.

Syntax

show tech-support interface-link [*slot*]

Parameters

slot

Specifies the slot number.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

The optional *slot* variable specifies the slot number for which you want to collect information for the link layer.

If the **show tech-support interface-link** command is executed on the global level, it collects debugging information for all the slots.

The output of the show tech-support interface-link command includes the output of the following commands:

- **show module**
- **show logging**
- **show running-config interface**
- **show statistics**
- **show interfaces brief**
- **show media**
- **show optic**
- **show optic thresholds**
- **show local-fault**
- **show remote-fault**
- **show bip**

Examples

The following is an example of the **show tech-support interface-link** command.

```
device# show tech-support interface-link
```

show tech-support ip multicast

Collects snooping information specific to IPv4 multicast from both LP and MP.

Syntax

show tech-support ip multicast [**vlan** *vlan-id* | **vpls** *vpls-id*] [*group-address*]

Parameters

vlan *vlan-id*
Displays multicast snooping information for the specified VLAN ID.

vpls *vpls-id*
Displays multicast snooping information for the specified VPLS ID.

group-address
Displays multicast snooping information for the specified group address.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

The list of commands captured from the MP and LP are different. When you specify a group address along with the VLAN ID or VPLS ID, the **show tech-support ip multicast** command displays multicast snooping information for the specified group address along with the global and VLAN- or VPLS-specific snooping information.

On the MP module, the default output of the **show tech-support ip multicast** command includes the output of the following commands:

- **show ip multicast**
- **show ip multicast resource**
- **show ip multicast static**

When you specify either the VLAN ID or VPLS ID on the MP, the **show tech-support ip multicast** command output includes the output of the following commands:

- **show ip multicast**
- **show ip multicast resource**
- **show ip multicast static**
- **show ip multicast vlan** *vlan_id* | **vpls** *vpls_id*
- **show ip multicast vlan** *vlan_id* | **vpls** *vpls_id* **statistics**
- **show ip multicast vlan** *vlan_id* | **vpls** *vpls_id* **igmpv3**
- **show ip multicast vlan** *vlan_id* | **vpls** *vpls_id* **pim**

- **show ip multicast vlan** *vlan_id* | vpls vpls_id tracking

When you specify a group address along with the VLAN ID or VPLS ID, the **show tech-support ip multicast** command displays multicast snooping information for the specified group address along with the global and VLAN- or VPLS-specific snooping information.

On the LP module, when you specify either the VLAN ID or VPLS ID, the **show tech-support ip multicast** command output includes the output of the following commands:

- **show ip multicast**
- **show ip multicast resource**
- **show ip multicast vlan** *vlan_id* | **vpls** *vpls_id*

Examples

The following is an example of the **show tech-support ip multicast** command.

```
device# show tech-support ip multicast vlan 10
```

show tech-support ip ospf

Displays system and debugging information specific to IPv4 OSPF configurations.

Syntax

show tech-support ip ospf [**route** *ip-address* | **vrf** *vrf-name* | **vrf-all**]

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

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 ip-address**
- **show ip ospf database link-state link-state-id ip-address**

Examples

The following is an example of the **show tech-support ip ospf** command.

```
device# show tech-support ip ospf
```

show tech-support ip pim

Displays system and debugging information specific to IPv4 PIM.

Syntax

show tech-support ip pim [**vrf** *vrf-name* | **all-vrf** | **mcache** *group-address* | *source-address* | **vrf** *vrf-name* **mcache** *group-address* | *source-address*]

Parameters

vrf *vrf-name*
Specifies the name of the VRF instance for which you want to display the PIM-specific configuration information.

all-vrf
Displays the PIM-specific configuration information for all the VRF instances.

mcache *group-address*
Displays the configuration information for all the VRF instances.

mcache *group-address*
Displays all IPv4 multicast cache entries for the specified group address.

source-address
Specifies the source IP address.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

The **show tech-support ip pim** command is used on both the MP and the LP to collect system and debugging information specific to IPv4 PIM. The list of commands captured from the MP and LP will be different.

On the MP module, with the default VRF or when you specify the **vrf** *vrf-name* parameter, the **show tech-support ip pim** command output includes the output of the following commands:

- **show ip pim sparse**
- **show ip pim dense**
- **show ip pim interface**
- **show ip pim neighbor**
- **show ip pim bsr**
- **show ip pim anycast-rp**
- **show ip pim rp-candidate**
- **show ip pim rp-set**
- **show ip pim prune**

- **show ip pim nsr**
- **show ip pim traffic**
- **show ip pim resource**
- **show ip pim optimization**
- **show ip pim counters**
- **show ip pim counter nsr**
- **show ip pim counter mct**
- **show ip pim counter tbp**
- **show ip pim mcache count**
- **show ip msdp summary**
- **show task-counter mcast**
- **show task mcast**
- **show ip igmp settings**
- **show ip igmp interface**
- **show ip igmp static**
- **show ip igmp ssm-map**

On the LP module, with the default VRF or when you specify the **vrf** *vrf-name* parameter, the **show tech-support ip pim** command output includes the output of the following commands:

- show ip pim settings
- show ip pim interface
- show ip pim neighbor
- show ip pim anycast-rp
- show ip pim rp-set
- show ip pim rep-table
- show ip pim resource
- show ip pim counters

On the MP module, when you specify the *mcache group_address* | *source_address* parameter, the **show tech-support ip pim** command output includes the output of the following commands:

- show ip igmp group
- **show ip pim** *mcache group address* | *source address*

On the LP module, when you specify the *mcache group_address* | *source_address* parameter, the **show tech-support ip pim** command output includes the output of the following commands:

- **show ip igmp group**
- **show ip pim** *mcache group_address* | *source_address*

Examples

The following is an example of the **show tech-support ip pim** command.

```
device# show tech-support ip pim all-vrf
```

show tech-support ip vrrp

Displays system and debugging information specific to IPv4 VRRP configurations.

Syntax

show tech-support ip vrrp

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

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**

Examples

The following is an example of the **show tech-support ip vrrp** command.

```
device# show tech-support ip vrrp
```

show tech-support ip vrrp-extended

Displays generates system and debugging information specific to IPv4 VRRP-extended (VRRP-E) configurations.

Syntax

show tech-support ip vrrp-extended

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

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**

Examples

The following is an example of the **show tech-support ip vrrp-extended** command.

```
device# show tech-support ip vrrp-extended
```

show tech-support ipv6 multicast

Collects snooping information specific to IPv6 multicast from both LP and MP.

Syntax

show tech-support ip multicast [**vlan** *vlan-id* | **vpls** *vpls-id*] [*group-address*]

Parameters

vlan *vlan-id*

Displays multicast snooping information for the specified VLAN ID.

vpls *vpls-id*

Displays multicast snooping information for the specified VPLS ID.

group-address

Displays multicast snooping information for the specified group address.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

The **show tech-support ipv6 multicast** command is used on both the MP and the LP to collect snooping information specific to IPv6 multicast. The list of commands captured from the MP and LP will be different.

The list of commands captured from the MP and LP are different. When you specify a group address along with the VLAN ID or VPLS ID, the **show tech-support ipv6 multicast** command displays multicast snooping information for the specified group address along with the global and VLAN- or VPLS-specific snooping information.

On the MP module, the default output of the **show tech-support ipv6 multicast** command includes the output of the following commands:

- **show ipv6 multicast**
- **show ipv6 multicast resource**
- **show ipv6 multicast static**

When you specify either the VLAN ID or VPLS ID on the MP, the **show tech-support ipv6 multicast** command output includes the output of the following commands:

- **show ipv6 multicast**
- **show ipv6 multicast resource**
- **show ipv6 multicast static**
- **show ipv6 multicast vlan** *vlan_id* | **vpls** *vpls_id*
- **show ipv6 multicast vlan** *vlan_id* | **vpls** *vpls_id* **statistics**
- **show ipv6 multicast vlan** *vlan_id* | **vpls** *vpls_id* **igmpv3**

- **show ipv6 multicast vlan** *vlan_id* | **vpls** *vpls_id* **pim**
- **show ipv6 multicast vlan** *vlan_id* | vpls vpls_id tracking

When you specify a group address along with the VLAN ID or VPLS ID, the **show tech-support ipv6 multicast** command displays multicast snooping information for the specified group address along with the global and VLAN- or VPLS-specific snooping information.

On the LP module, when you specify either the VLAN ID or VPLS ID, the **show tech-support ipv6 multicast** command output includes the output of the following commands:

- **show ipv6 multicast**
- **show ipv6 multicast resource**
- **show ipv6 multicast vlan** *vlan_id* | **vpls** *vpls_id*

Examples

The following is an example of the **show tech-support ipv6 multicast** command.

```
device# show tech-support ipv6 multicast vlan 11
```

show tech-support ipv6 ospf

Displays system and debugging information specific to IPv6 OSPF configurations.

Syntax

show tech-support ip ospf [**route** *ip-v6address* | **vrf** *vrf-name* | **vrf-all**]

Parameters

route *ipv6-address*
Specifies the destination IPv6 address for an OSPF route.

vpls *vpls-id*
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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

The default output of the **show tech-support ipv6 ospf** command includes the output of the following commands:

- **show ipv6 ospf summary**
- **show ip ospf**
- **show ip ospf area**
- **show ipv6 ospf spf tree**
- **show ipv6 ospf debug 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 ip ospf route** *ipv6-prefix*
- **show ipv6 ospf database prefix** *ipv6-prefix*

Examples

The following is an example of the **show tech-support ipv6 ospf** command.

```
device# show tech-support ipv6 ospf vrf-all
```

show tech-support ipv6 pim

Displays system and debugging information specific to IPv4 PIM.

Syntax

show tech-support ipv6 pim [**vrf** *vrf-name* | **all-vrf** | **mcache** *group-address* | *source-address* | **vrf** *vrf-name* **mcache** *group-address* | *source-address*]

Parameters

vrf *vrf-name*
Specifies the name of the VRF instance for which you want to display the PIM-specific configuration information.

all-vrf
Displays the PIM-specific configuration information for all the VRF instances.

mcache *group-address*
Displays the configuration information for all the VRF instances.

mcache *group-address*
Displays all IPv6 multicast cache entries for the specified group address.

source-address
Specifies the source IP address.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

The **show tech-support ipv6 pim** command is used on both the MP and the LP to collect system and debugging information specific to IPv6 PIM. The list of commands captured from the MP and LP will be different.

On the MP module, with the default VRF or when you specify the **vrf** *vrf-name* parameter, the **show tech-support ip pim** command output includes the output of the following commands:

- **show ipv6 pim sparse**
- **show ipv6 pim dense**
- **show ipv6 pim interface**
- **show ipv6 pim neighbor**
- **show ipv6 pim bsr**
- **show ipv6 pim anycast-rp**
- **show ipv6 pim rp-candidate**
- **show ipv6 pim rp-set**
- **show ipv6 pim resource**

- **show ipv6 pim counter group**
- **show ipv6 pim traffic**
- **show ipv6 pim counter**
- **show ipv6 pim counter mct**
- **show ipv6 pim counter tbp**
- **show ipv6 pim mcache count**
- **show ip msdp summary**
- **show task-counter mcast6**
- **show task mcast6**
- **show ipv6 mld settings**
- **show ipv6 mld interface**
- **show ipv6 mld traffic**

On the LP module, with the default VRF or when you specify the **vrf** *vrf-name* parameter, the **show tech-support ipv6 pim** command output includes the output of the following commands:

- show ipv6 pim settings
- show ip pim interface
- show ipv6 pim neighbor
- show ipv6 pim anycast-rp
- show ip pim rp-set
- show ip pim counters
- show ip pim resource

On the MP module, when you specify the *mcache group_address* | *source_address* parameter, the **show tech-support ipv6 pim** command output includes the output of the following commands:

- show ipv6 mld group
- **show ipv6 pim** *mcache group address* | *source address*

On the LP module, when you specify the *mcache group_address* | *source_address* parameter, the **show tech-support ipv6 pim** command output includes the output of the following commands:

- **show ipv6 pim** *mcache group_address* | *source_address*

Examples

The following is an example of the **show tech-support ipv6 pim** command.

```
device# show tech-support ipv6 pim all-vrf
```

show tech-support ipv6 vrrp

Displays system and debugging information specific to IPv6 VRRP configurations.

Syntax

show tech-support ipv6 vrrp

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

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**

Examples

The following example is an example of the **show tech-support ipv6 vrrp** command.

```
device# show tech-support ipv6 vrrp
```

show tech-support ipv6 vrrp-extended

Generates system and debugging information specific to IPv6 VRRP-extended (VRRP-E) configurations.

Syntax

show tech-support ipv6 vrrp-extended

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

The default output of the **show tech-support ipv6 vrrp-extended** command includes the output of the following commands:

- **show ipv6 vrrp-extended brief**

Examples

The following is an example of the **show tech-support ipv6 vrrp-extended** command.

```
device# show tech-support ipv6 vrrp-extended
```

show tech-support isis

Displays the system and debugging information specific to IS-IS configurations.

Syntax

show tech-support tm route[*ip-prefix* | *ipv6-prefix*]

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

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-se**t
- **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-orde**r

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**

Examples

The following example is an example of the **show tech-support isis** command.

```
device# show tech-support isis route 2001:db8:a::123/64
```

show tech-support l4

Displays the Layer 4 configuration information.

Syntax

show tech-support ip ospf acl *acl-name* | *all*

Parameters

acl *acl-name*

Specifies the name of the Access Control List (ACL) instance for which you want to display the Layer 4 configuration information.

acl *all*

Collects Layer 4 configuration information for all the ACL instances.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

The **show tech-support l4** command is supported only on MP.

Examples

The following is an example of the **show tech-support l4** command.

```
device# show tech-support l4
```

show tech-support mpls

Displays the system and debugging information specific to MPLS configurations.

Syntax

show tech-support tm [**brief** | **detail** | **debug** | **ldp** | **rsvp** | **static-lsp**]

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

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 also use the **brief**, **detail**, and debug options with the **ldp**, **rsvp**, and **static-lisp** 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**

Examples

The following is an example of the **show tech-support mpls** command.

```
device# show tech-support mpls brief
```

show tech-support mpls label

Displays the information related to MPLS cross-connect entries and corresponding label RAM information on each packet processor.

Syntax

show tech-support tm *label*[**brief** | **detail**]

Parameters

label

Specifies the label ID.

brief

Displays the brief technical support information for the MPLS label.

detail

Displays detailed technical support information for the MPLS label.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

This command combines and displays the output of the **show tech-support** commands related to MPLS cross-connect entries and corresponding label RAM information on each packet processor.

On the MP module, the show tech-support mpls label command output includes the output of the following commands:

- **show mpls forwarding**

For each of the LPs that has the MPLS interface enabled, the following command outputs are collected and displayed:

- **show mpls lsp_xc** *in-label*
- **show nht-table**
- **dm label-ram me/N LABEL**
- **dm pram** *slot/interface pram_index* **mpls**

Examples

The following is an example of the **show tech-support mpls label** command.

```
device# show tech-support mpls label
```

show tech-support openflow

Displays system and debugging information specific to OpenFlow configurations.

Syntax

show tech-support openflow

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

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**

Examples

The following is an example of the **show tech-support openflow** command.

```
device# show tech-support openflow
```

show tech-support rtm

Displays system and debugging information specific to IPv4 RTM configurations.

Syntax

show tech-support rtm6 [**vrf** *vrf-name* [**route** *ipv-prefix*]

Parameters

vrf *vrf-name*

Specifies the name of the VRF instance for which you want to display the configuration information.

route *ipv6-prefix*

Specifies the IPv4 destination prefix and mask length for the RTM route.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

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**

Examples

The following is an example of the **show tech-support rtm** command.

```
device# show tech-support rtm
```

show tech-support rtm6

Displays system and debugging information specific to IPv6 RTM configurations..

Syntax

show tech-support rtm6 [**vrf** *vrf-name* [**route** *ipv6-prefix*]

Parameters

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.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

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 ipv6rtm api**

For a particular route, the output of the **show tech-support rtm6** command includes the output of the following commands:

- **show ipv6** *route ip-prefix* **debug**
- **show ipv6 static route** *route ip-prefix*
- **show ipv6 ospf routes** *pv6-address*
- **show isis routes** *route ip-prefix*
- **show ipv6 bgp routes** *route ip-prefix*
- **show ipv6 rip route** *route ip-prefix*
- **show ipv6 network** *route ip-prefix*

Examples

The following is an example of the **show tech-support rtm6** command.

```
device# show tech-support rtm6
```

show tech-support sfm

Displays system and debugging information specific to SFM and high-speed SFM (hSFM) configurations.

Syntax

show tech-support sfm

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

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**

Examples

The following is an example of the **show tech-support sfm** command.

```
device# show tech-support sfm
```

show tech-support system

Displays the system-related information for debugging purposes.

Syntax

show tech-support system

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

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**

- lpc 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

Examples

The following is an example of the **show tech-support system** command.

```
device# show tech-support system
```

show tech-support tm

Displays system and debugging information specific to Traffic Manager (TM) configurations.

Syntax

show tech-support tm [**slot** *islot_number*] [**critical-registers**]

Parameters

slot *slot_number*
: Collects technical support information related to the TM for a specific slot.

critical-registers
: Collects critical registers information for the TM.

Modes

Privileged EXEC mode

Usage Guidelines

Diagnostic commands are developed and intended for specialized troubleshooting. Please work closely with Extreme Networks technical support in running **debug** or **show system internal** commands and interpreting their results.

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**

NOTE
Pruning discards on egress TM(s) can be expected without any loss in traffic due to source port suppression for broadcast, multicast and unknown unicast traffic.

Examples

The following example is an example of the **show tech-support tm** command.

```
device# show tech-support tm
```